Teaching Advanced Architectural Issues Through Principles of CAAD

Henri Achten[1]

Abstract

The paper discusses the differences between teaching CAAD by using standard software ("off-the-shelf"-software) and teaching the principles of CAAD ("principles-teaching"). The paper distinguishes four kinds of application for design systems in education: social systems, professional systems, educational systems, and innovative systems. The paper furthermore proposes to distinguish between computational issues and architectural issues relative to design systems. It appears that there is not a principled distinction between software-teaching and principles-teaching when it comes to computational issues of design systems. However, when the architectural content of CAAD systems is concerned, then principles of CAAD systems seem to be more appropriate for teaching. The paper presents work on generic representations as a specific case. Generic representations can be used to teach one particular kind of architectural content of design systems. The paper ends with conclusions.

Introduction

The paper aims to clarify some positions about the differences between software-teaching and principles-teaching in courses that apply CAAD, and to show in what context they may be related. The first two sections establish a framework of terminology which can be used to highlight architectural and computational content in CAAD-courses. The section on Generic representations presents a particular approach developed in research work and teaching. It is used as an illustration of the framework.

Four different kinds of computer systems

In a recent survey considering the use of computers in education, a workgroup of the Department of Architecture, Building, and Planning distinguished on the basis of Plomp (1996) four kinds of computer systems in education: social systems, professional systems, educational systems, and innovative systems[2].

Social systems are computer tools which any one architecture student should be able to use. Examples are word processors, information retrieval systems, spreadsheets, and databases.

Professional systems are computer tools as they are used in architectural practice today. Examples are CAAD software, expert shells, calculation software, and geographic information systems.

Educational systems are dedicated computer tools (or modified existing software) to convey specific pedagogical purposes. Educational systems usually are developed at the institution where they are used in teaching. Cases at the Department of Architecture in the Eindhoven University of Technology are using AutoCAD for "negative design," and using internet browsers and rendering software for multimedia courses.

---

[1] Eindhoven University of Technology. Department of Architecture, Building, and Planning. Design Methods Group.

[2] Plomp distinguishes three 'rationales' for teaching information technology: social, professional, and innovative. In discussions it seemed proper to introduce the educational rationale as well. The survey this paper mentions is not yet published at the time of writing the paper.

Innovative systems are computer systems that reach beyond current state of the art of professional systems. Examples are automated plan recognition, virtual reality design systems (van Zutphen et al. 1996 and Smeltzer et al. 1994), and the development of constraint propagation software.

Off-the-shelf software and Home-made software

Social systems and professional systems usually are ”off-the-shelf” software, that is, software developed by regular software houses (such as AutoCAD, 3DStudio, Arkey, WordPerfect, etc.) Educational systems and innovative systems usually are ”home-made” software, that is, software developed mainly in the institution itself. Educational systems sometimes - and innovative systems always - result from research work (see Figure 1).

”Off-the-shelf” CAAD-software can be used both as a social system (providing basic required

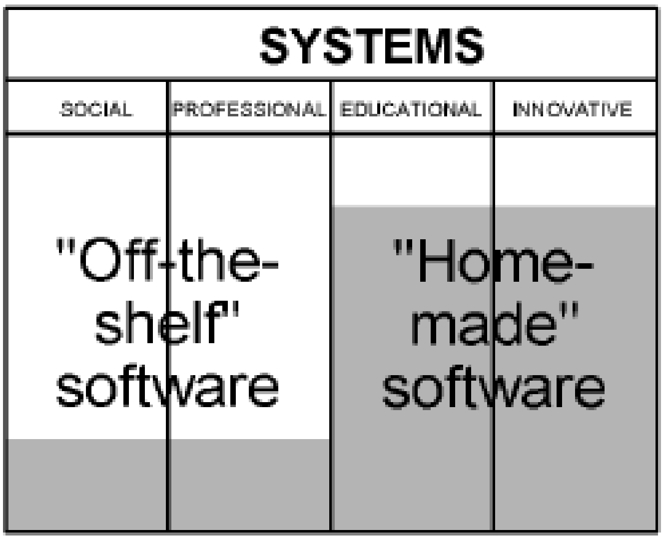


Figure A: Four kinds of uses for computer systems: Social and Professional systems usually are ”off-the-shelf” software; Educational and innovative systems usually are ”home-made”software.

knowledge of design systems) and as a professional system in teaching (providing experience of the use of CAAD systems in practice). Principles-teaching usually is provided through the use of educational systems (which has specific pedagogic aims) and innovative systems (which has specific research advancing aims).

When there is an aim to integrate or incorporate CAD or CAAD in the curriculum, we may well state that teaching social systems is an unchallenged feature of any curriculum. The paper will therefore concentrate on the remaining three types; professional, educational, and innovative systems.

Computational and Architectural Issues

The distinction made between four kinds of systems does not yet help very much in judging between software-teaching versus principles-teaching. Both CAD and CAAD-systems may be inserted in any category. It seems necessary to make a further distinction that will enable to see the differences between CAD and CAAD. This distinction can be found by considering automated design systems as instruments, and by the requirement of computational systems of rigorous and consistent definition of their properties.

Design systems as instruments

In the most general meaning of the word, an instrument is an aid to improve human performance. Any instrument incorporates some knowledge of the task it is subjected to. Next to knowing the procedural aspects and inherent means, goals, and so forth of the instrument, it is also required to know how to realize the instrument itself. Instruments are not only as good as the grasp on the problem we seem to have, but also as good as the way in which we can implement and manage them. The idea that instruments incorporate knowledge can be readily demonstrated in, for example, instruments which measure and make something. These instruments reveal concepts of length, angle, straightness or roundness, weight, time, and so forth. For a ruler this implies that it must be straight and divided in equal distances in order for it to be an instrument for drawing and measuring straight lines. Clocks measure time by utilizing periodical recurring phenomena which repeat themselves accurately enough. Instruments that organize work reveal concepts of time, process, order, and causation. Design instruments reveal concepts of design problems, solution strategies, solution types, and the specific design domain. By the same token, we may state that the instrument that is a CAAD-system reveals architectural concepts.

Design systems require rigorous and consistent definitions

The second ground for using design systems to teach architectural issues lies in the requirement of design systems for rigorous and consistent definitions in order to implement them. From the perspective of advanced research fields such as Artificial Intelligence in design, cognitive science, design computation, and design research, it seems clear that a significant part of the (scientific) study of architectural issues profits from the use of computers. It comes natural to combine architectural issues and computational issues to teach principles of CAAD from the perspective of architectural theory.

The discussion above implies that CAD and CAAD systems have a common ground in so-called *computational issues* and a different ground in so-called *architectural issues*. It is the application and adherence to architectural issues that make CAAD systems differ from CAD systems. Therefore, next to the four kinds of systems, we propose to distinguish also between ‘computational issues’ and ‘architectural issues.’

We have now established a framework (see Figure 2) that will help clarify positions on software versus principles-teaching. In particular, we will discuss ongoing research work that has been applied to education. This discussion will underscore the view that principles-teaching is connected with architectural issues.

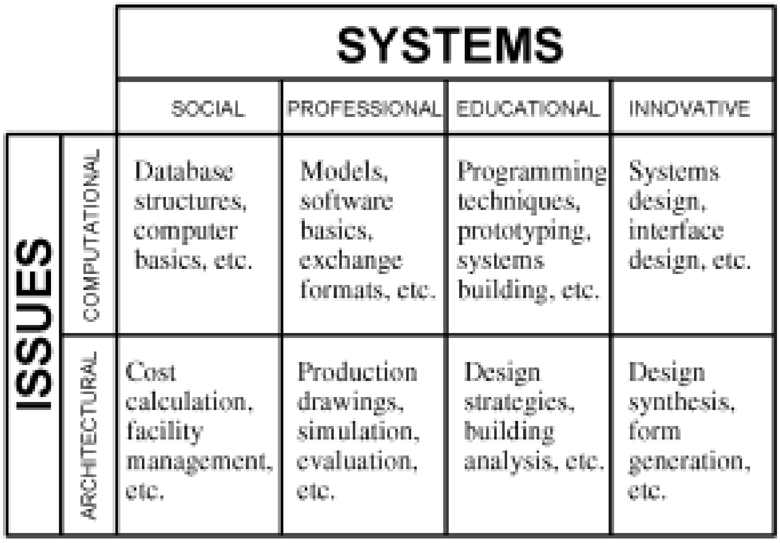


Figure B: Framework of terminology for discussing software versus principles teaching in a CAAD-curriculum.

Generic representations

Generic representations are developed to formulate and implement knowledge of building types. Building types are complex forms of knowledge that could advance CAD systems into CAAD systems. A generic representation is a schematic graphic representation denoting a particular state of the design. It consists of a specified set of graphic elements through which knowledge of the design object is encoded. A single generic representation encodes typological knowledge of the building type (that part of declarative knowledge which applies to a building type). A sequence of generic representations encodes generic knowledge of the building type (that part of procedural knowledge which defines design strategies relative to a building type).

The next three pages show a sequence of generic representations applied to the office building type. Each row in the table is a specific generic representation, such as ”Simple contour.” The first column - Icon - shows in a graphic manner through an icon the characteristics of the generic representation. The second column - Building and Representation - shows how a building design might appear as it shows the properties of a generic representation. The third column - Name and some characteristics - briefly describes the design decisions implied by the generic representation relative to the office building type.

The theory of generic representations has been taught in the context of programming and principles of knowledge-based systems. Students were taught how to analyse the office building type on its knowledge content using generic representations. Each student programmed a knowledge-based system based on the first seven generic representations. The structure of the system derived from the analysis. Results and particular approach of the course have been reported in Achten (1995a; 1995b).

Table 1: A sequence of generic representations applied to the office building type [1]

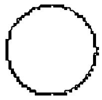
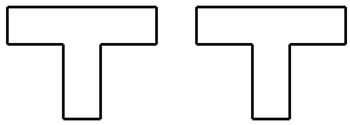
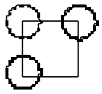
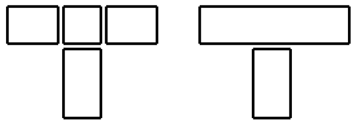
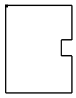
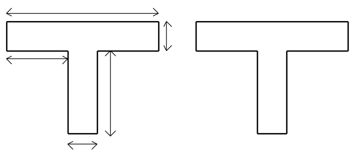
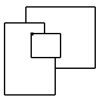
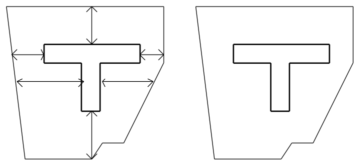
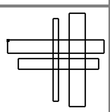
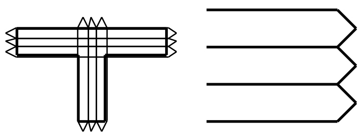
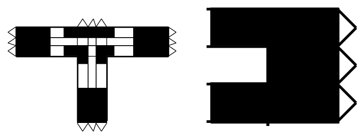
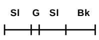
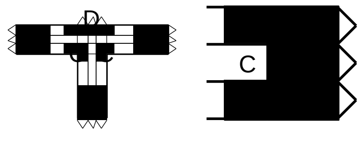
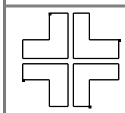
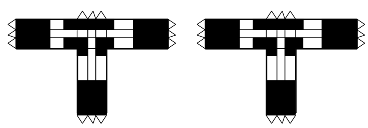
Icon	Building Representation	Name and some characteristics
		**Simple contour** Defining the outward form of the building. Establishing the T-shape; triple-winged building. Surface area. Parametrize wing-length.
		**Combination of simple contours** Composing ensemble of simple contours to establish overall shape. Define internal proportions and place of simple contours. Explore emergent forms.
		**Specified form** Establish tentative dimensions for wing length and depth, and orientation of the building.
		**Complementary contours** Establish place of building mass in site. First tentative estimate of grid active at the urban level. Relate to demands of distance from site, and other buildings.
		**Zone** Zoning structure establishes a principle of ordering the building. Establish a zoning principle for the wings, e.g. single, double, or triple zone with central circulation.
		**Schematically subdivided zone** Along a zone, establish areas that have specific qualities such as lighting, circulation, accessibility, etc. This results in an inventory of possibilities.
		**Function symbols in schematically subdivided zone** Allocate tentative functions in specific areas along a zone, relative to its qualities. This action is like an inventory of possibilities.
		**Zone in specified form** Establish the zoning system in the building form. Establish the dimensions of the zones, identify special places such as intersections, end of wing, internal/external corners, etc.

Table 2: A sequence of generic representations applied to the office building type [2]

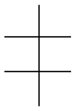
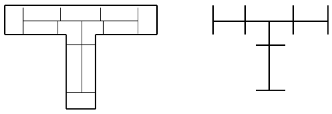
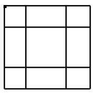
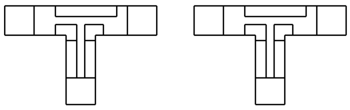
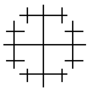
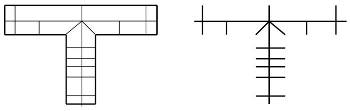
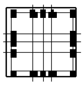
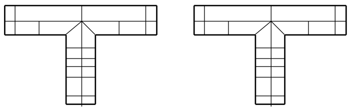
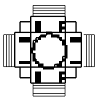
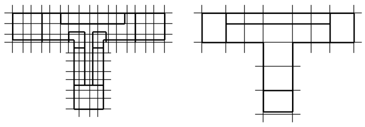
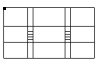
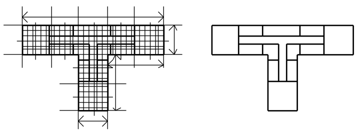
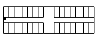
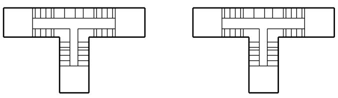
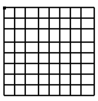
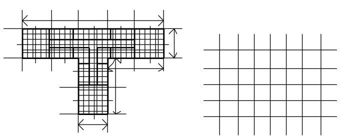
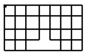
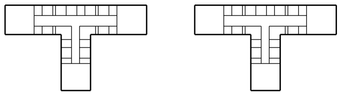
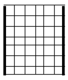
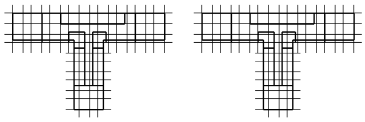
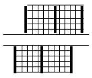
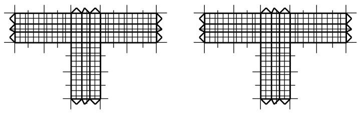
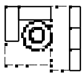
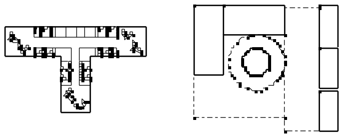
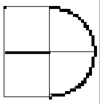
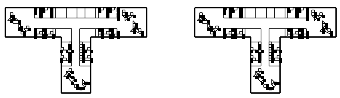
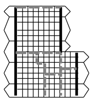
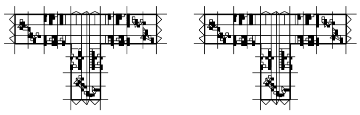
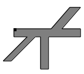
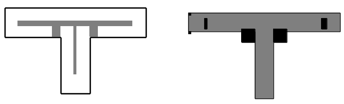
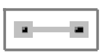
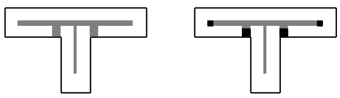
Icon	Building Representation	Name and some characteristics
		**Schematic subdivision** Divide the building into sections that can be considered quite independently from each other. For each section, establish a principle division into parts.
		**Schematic subdivision within contour** Subdivide the contour of the building according to the schematic subdivision. Identify tentative surface areas to parts of the subdivision.
		**Schematic axial system** Establish principle of axes that co-ordinate spaces. Axes of the system define lines of symmetry of spaces. Layout of major rooms and places. Organisation principle.
		**Schematic axial system superimposed on specified form** Place system of axes that define organisation of spaces in the specified form. Define general dimensions of spaces.
		**Schematic subdivision as modular zones in archetypes** Co-ordinate the schematic subdivision in the specific form along a grid or set of grids. Define major spaces within the subdivision.
		**Subdivision of specified form** Subdivision of the specified form according to the grid underlying this level. General organisation of the building layout of major spaces.
		**Partitioning system in simple contour** Principle of partitioning along which future divisions may be placed. Establish module for rooms.
		**Modular field** Establish grid of building according to modules and dimensions already available.

Table 3: A sequence of generic representations applied to the office building type [3]

Icon	Building Representation	Name and some characteristics
		**Contour and subdivided field** Establish place of rooms and spaces according to underlying grid.
		**Contour and modular field** Superimpose the modular field (grid) on the specified form. Establish the module of the grid.
		**Zone in contour and modular field** Co-ordinate the zone structure according to the module of the grid.
		**Element vocabulary and combinatorical rules** Establish sets of furnishing for parts of the building according to functional requirements, program of demands, and suppliers.
		**Contour and element vocabulary** For parts defined according to subdivision of specified form, determine usability and functionality by interior elements (furnishing).
		**Element vocabulary in zone, contour and modular field** Determine general layout, furnishing, and zoning of the building design adhering to the Generic grid.
		**Circulation scheme** Establish circulation principle according to zoning and schematic axial principle.
		**Contour with circulation** Dimension circulation in building design according to requirements and program of demands.

The framework applied to the course

The framework established in the second section can now be applied to the knowledge-based systems course discussed above. Architectural issues covered in the educational system (see Figure 3) are *typological knowledge, generic knowledge,* and the *role of graphic representations* in design. Architectural issues covered in the context of innovative systems are *building types, generic representations*, and *intermediate structures.* Computational issues that are dealt with in the educational system are *principles of knowledge-based systems* and *systems programming.* The computational issue in the context of innovative systems was the role of *knowledge application in design systems*. We will discuss them briefly.

		SYSTEMS			
		SOCIAL	PROFESSIONAL	EDUCATIONAL	INNOVATIVE
ISSUES	COMPUTATIONAL			Principles of Knowledge Based Systems Systems Programming	Knowledge application in design systems
	ARCHITECTURAL			Typological knowledge, Generic knowledge, Graphic representations in design	Intermediate structures Generic representations Building types

Figure C: Architectural issues through principles of CAAD: typological and generic knowledge, the use of graphic representations in design, and the role of intermediate structures, generic representations, and building types for design systems. Computational issues: knowledge-based systems (programming) and knowledge application in design systems.

Architectural issues

Architectural issues that are covered through the educational use of Knowledge-based systems programming concern the role of graphic representations to encode typological and generic knowledge. We find that typological knowledge is that subset of declarative knowledge which is specific to a building type (that is, the statements that can be made to apply for example, to an office building). This body of knowledge related to a building type can be made specific partly by analysing the literature on office buildings. Students learned in this manner the relationship between knowledge in literature and building types.

In the same vein, generic knowledge is that subset of procedural knowledge which is specific to a building type (that is, those design strategies that are wise to follow when dealing with a certain type, e.g. an office building). It is proposed by the theory of generic representations that typological knowledge may be expressed in single generic representations, and that sequences of generic representations may express generic knowledge. One particular design strategy which has been followed both in the educational system (programming the knowledge-based system for the office building) and the innovative system (the research) is that of particularization. By showing this strategy to succeed in generating a knowledge-based system, the theory of generic representation has at least made plausible that it is operational. Students thus required knowledge of the importance of design strategies, and the way knowledge of building types acts in this.

The role of graphic representations to encode knowledge throughout the design process and its potential when seen as an intermediate structure[3] for developing innovative systems has been dealt with as well. In this manner, it is possible to gain insight into how an advanced architectural notion such as building type may be tackled in a CAD system, thus yielding a CAAD system. Students learned about the significance of graphic representations, and how these can aid in pulling apart the complex set of decisions involved in design.

Computational issues

When discussed from the computational issues perspective, the theory of generic representations demonstrated the role of knowledge application throughout the design process. It shows the importance of both having knowledge and knowing when and how to apply it. Both the notion of 'intermediate structures' and the procedural aspects embedded in a sequence of generic representations highlighted a designerly approach to implementing building types in a CAD system. The rudimentary programming of a knowledge-based system in an AutoCAD environment in AutoLISP began to show the principles of this kind of systems and programming them. It also indicated some of the potential of the theory of generic representations to structure, build, and interface knowledge-intensive design systems. Both structuring (analysis of the office building type through generic representations) and building (programming course) have been dealt with. It is important to note that by referring to the name and icon of the generic representation, a designer may keep track of the design process and the sequence of decisions he has gone through. In this manner, generic representations offer an interface to the knowledge applied in the design process.

Conclusions

Teaching generic representations is principles-teaching in an educational system.

Relative to the question of software versus principles-teaching, we may state that the work described here typically is a case of principles-teaching. The theory of generic representations has been taught by programming a rudimentary knowledge-based system. As can be seen from its drawbacks and underdevelopment (see Achten et al. 1995a; 1995b), it is far from being a developed and full-fledged design system. Therefore, it must be considered to fall in the class of educational systems rather than innovative systems.

Software-teaching is technology-pushed.

Teaching CAAD by software-teaching is by nature technology-pushed: what the software offers may be taught. The general theoretical part discussing the basics of CAAD has to be added by the teachers. However, this should not be considered a fundamental obstacle to principles-teaching. A CAAD system offers opportunities for discussing these issues and elaborating on ways they can be implemented. Furthermore, open CAD-systems such as AutoCAD allow substantive addition to the core of the systems software, thus resulting in educational systems.

"Off-the-shelf" systems do not oppose principles-teaching.

Since CAAD-systems are based on principles of CAAD, teaching them also provides insight into principles of CAAD. However, this will be achieved indirectly to say the least, and incomplete in any case since each system exhibits specific implementation issues.

---

[3] An intermediate structure is a manipulable graphic structure which represents a building design. It is structured in the sense that it has specific regularities which define architectural properties. It constraints and supports design actions by designers. It facilitates the interchange of actions by the computer system and the designer (Achten 1996a; 1996b).

The pedagogical perspective determines software-teaching versus principles-teaching.

Ultimately, choosing between principles-teaching and software-teaching depends on the pedagogical perspective, rather than on the fact if one uses ”off-the-shelf” software or ”home-made” software. A perspective from practice will favour professional and educational systems, making not much distinction between computational and architectural issues. Since this perspective will tend to use ”off-the-shelf” software, it will assume that architectural issues have been computationally realized in the systems software. A perspective from research will favour innovative and educational systems. Since there is a drive to go beyond what is available, work from this perspective will aim to advance either computational issues (such as genetic algorithms or constraint propagation), architectural issues (such as multimedia or designing in virtual environments), or both (such as generic representations, knowledge-based systems programming or shape grammars).

Acknowledgements

The research described in the paper is an ongoing PhD.-research performed in the Design Methods Group at the Faculty of Architecture, Building, and Planning. The course described in the paper has been given jointly by the Building Information Technology Group and the Design Methods Group. The author would like to thank Prof.dr. M.F.Th. Bax and Prof.dr.arch. R.M. Oxman for supervising the research and commenting on the paper, ir. R. van Zutphen for commenting on the paper and providing the distinction between four kinds of computer system applications in education, and Ing. J. Dijkstra for embedding the research work in the knowledge-based systems programming class.

References

Achten, H.H. and Dijkstra, J. and Oxman, R. and Bax, M.F.Th. (1995a). Knowledge-based systems Programming for Knowledge Intensive Teaching. From: Multimedia and Architectural Disciplines. Colajanni, Benedetto and Pellitteri, Giuseppe (eds.), 1995, Selected Proceedings of the 13th European Conference on Multimedia and Architectural Disciplines, Palermo

Achten, H.H. and Dijkstra, J. and Oxman, R. and Bax, M.F.Th. (1995b). Knowledge-based systems Programming for Knowledge Intensive Teaching. BIT-Note Publication 1995/3, Eindhoven. Extended text and included source-code of Achten (1995a).

Achten, H.H. (1996a). Generic Representations: Intermediate Structures in Computer Aided Architectural Composition. From: Approaches to Computer Aided Architectural Composition. Asanowicz, A. and Jakimowicz, A. (eds.), 1996, TU Bialystok, Bialystok (forthcoming). ISBN 83-905377-1-0

Achten, H.H. and Bax, M.F.Th. and Oxman, R.M. (1996b). Generic Representations and the Generic grid: Knowledge Interface, Organisation and Support of the (early) Design Process. From: Proceedings of the 3rd Design and Decision Support Systems in Architecture and Urban Planning Conference. Timmermans, H. (ed.), August 18-21 1996, Spa, Belgium

Plomp, T. (1996). IT in het onderwijs. From: Wijsgerig Perspectief op Maatschappij en Wetenschap, 36, 1995-1996, p. 19 e.v.

Smeltzer, G.T.A. and Mantelers, J.M.M. and Roelen, W.A.H. (1994). The Application of Virtual Reality Systems in Architectural Design Processes. BIT Note Publication 95/1, Building Information Technology, Eindhoven

Zutphen, R. van and Mantelers, J.M.M. (1996). Computational Design: Simulation in Virtual Environments, Proceedings of the 3rd Design and Decision Support Systems in Architecture and Urban Planning Conference, August 18-21, 1996, Spa, Belgium