Design Process and Knowledge
Possibilities and Limitations of Computer-Aided Design

by

Konstantinos Michael Papamichael

(Graduate of the University of Thessalonika, Greece) 1980
M.Arch. (Iowa State University) 1983

DISSERTATION
Submitted in partial satisfaction of the requirements for the
degree of
DOCTOR OF PHILOSOPHY
in
Architecture

in the
GRADUATE DIVISION
of the
UNIVERSITY OF CALIFORNIA, BERKELEY

Approved

Date

Jean-Pierre Protzen, Chairman 8-8-91

Richard Fernau 8-16-91

Arthur Rosenfeld 8-8-91
Design Process and Knowledge
Possibilities and Limitations of Computer-Aided Design

Copyright © 1991

by

Konstantinos Michael Papamichael
Design Process and Knowledge
Possibilities and Limitations of Computer-Aided Design

by
Konstantinos Michael Papamichael

Abstract

An attempt to determine how computers can be used to assist designers resulted in the development of a design theory, according to which design is "feeling and thinking while acting." Design is theorized as living through one's imagination, however being continuously affected by real life itself.

The design process is decomposed into elementary activities that are characterized with respect to the nature of knowledge requirements and the degree to which they can be specified and delegated to computers. The results are considered as criteria to determine possibilities and limitations of computer-aided design.

An integration of a variety of computer applications tools is proposed towards the design and development of a computer-based Design Support Environment (DSE), that is applicable to any design domain. The proposed DSE automates all specifiable and delegable design activities, while assisting with the nondelegable ones through appropriate user interface.

A DSE demonstration prototype is also presented in the
Appendix. This prototype addresses the design of fenestration and electric lighting systems of office spaces with respect to comfort, energy and cost.
I dedicate this dissertation to my professor,

*Horst W. J. Rittel*

who opened my eyes for me to see
and closed his before I could clearly describe what I saw,

and to my parents,

*Michael and Ourania*

who gave me the eyes to start with.
Table of Contents

List of Figures .................................................. v

Acknowledgements ............................................. vii

Chapter I. Introduction ........................................ 1
  Objective ....................................................... 3
  Background .................................................... 3
    First Generation Approach ................................. 4
    Second Generation Approach .............................. 4
  Issues ........................................................ 6
  Methodology .................................................. 7

Chapter II. Design Theory ..................................... 10
  Formulation of the ought-to-be performance .............. 14
    Determination of performance variables ............... 21
    Formulation of performance criteria ................... 23
    The relative importance of performance criteria .... 26
  Development of the will-be description .................. 29
    Creativity ................................................ 34
  Determination of the will-be performance ............... 34
  Design Expertise ............................................ 36
    Expertise and Creativity ................................ 39

Chapter III. Computer-Aided Design ........................ 40
  Formulation of the ought-to-be performance .............. 42
    Determination of performance variables names ....... 42
    Formulation of performance criteria ................... 44
  Development of the will-be description .................. 46
  Determination of the will-be performance ............... 50
    A computer-based Design Support Environment (DSE) .. 52
    Structure .................................................. 52
    Operation .................................................. 54
    Development and growth .................................. 57

Chapter V. Conclusions ........................................ 60

References ..................................................... 62

Appendix. A DSE Demonstration Prototype .................... 66
List of Figures

Figure 1. Design process flow diagram. ......................... 13

Figure 2. A time cycle of the linear process of life, showing that design involves both, feeling and thinking, being an inherent part of life. ............................................. 18

Figure 3. Life through time. Design is the equivalent of imaginary life. ................................. 20

Figure 4. The structure of the DSE................................. 53

Figure 5. The performance variables considered in the DSE demonstration prototype, presented as outcome of deliberation on the overall performance. ................................. 70

Figure 6. The control (design and context) variables considered in the DSE demonstration prototype, presented as objects with attributes (outline style indicates nominal scales and italic style indicates pseudo-ordinal scales). ...... 71

Figure 7. The main screen of the user interface of the DSE prototype. ................................. 72

Figure 8. The specification of minima and maxima acceptable values for performance variables is optional, aided by the use of an Issue-Based Information System (IBIS) that may also include positions based on codes and standards. ................................. 73

Figure 9. The specification of the desired direction for the values of performance variables. ........ 74

Figure 10. Automatic indication of the control variables that affect the values of performance variables towards the specified desired direction. ................................. 75

Figure 11. The specification of proposed values for design variables. ................................. 76
Figure 12. Automatic indication of the effects on the values of performance variables by the proposed values for design variables. ........ 77

Figure 13. The glazing database of the DSE demonstration prototype. ........................................... 78

Figure 14. The specification of the consideration of pseudo-design variables as design ones. ..... 79

Figure 15. The specification of a "new" solution through request for the determination of the value of a performance variable. ..................... 80

Figure 16. The navigation through the potential design solutions is either chronological (through the left and right pointing arrows) or following the parent/children relations. ............... 81

Figure 17. Request for explanation on performance and control variables links the user to an Electronic Handbook Information System (EHIS). ...................................................... 82

Figure 18. Performance variables may be ignored at user request. ............................................. 83
Acknowledgements

I thank my dissertation committee members for their assistance, especially Jean-Pierre Protzen who taught me the steps and let me free to dance, respectfully considering every single move I made.

I thank the members of the Building Technologies Program of the Applied Science Division at Lawrence Berkeley Laboratory for supporting my work, especially Steve Selkowitz for trusting me and bearing with my academic attitude.

I thank all my friends for bearing with my behavior, especially Panos Kliafas for providing constructive argumentation and encouragement.

I thank my son, Michael, for teaching me my ignorance.

I thank God it’s over.
Chapter I. Introduction

Continuously decreasing cost has brought computers into most architectural and engineering offices, most commonly for activities such as drafting, accounting and word processing. Computers are used less often to predict the performance of design solutions; performance simulation software packages, most of which are simplified versions of mainframe analytical tools originally developed for research purposes, are used for this purpose. However, such software packages are focusing on specific design issues according to the research needs that originated them. In addition, the data input requirements are complicated and incompatible with each other, and the output data are usually specialized and difficult to interpret. As a result, they can be seen as a more powerful approach to using calculators, that is, as the evolution of hand-calculations. It is yet to be seen how the increasing memory and processing speed of computers, the two main advantages that computers have over the human brain, can be used to assist designers through the development of integrated computer-based design-support environments.

As society's demand for better products increases, along with the available technological options emerging from research and industry competition, memory and processing speed are increasingly needed by designers. The number of design criteria exceeds by far what the average human can handle simultaneously. The same is true with the number of
the technological options that emerge as a response to human needs, and whose performance cannot be known a-priori since it depends on the context of their application. The design process is becoming increasingly demanding and complicated. The development of appropriate design tools to assist designers in managing this data-explosion situation is essential for the successful outcome of design projects.

A major effort is under way during the recent years to utilize the power of computers to assist designers directly in the design process, possibly at the early stages of design, where most of the important design decisions are made. Since performance assessment through simulation is not possible during the early design stages, heuristics are implemented, in the form of the so-called Expert or Knowledge-Based Systems, which emerged from the field of Artificial Intelligence [Rich 1983].

Example applications within the building design domain include diagnosis of problems with various types of building equipment [Haberl et al. 1989] and selection of various building components and systems [Degelman and Kim 1988; Tuluca et al. 1989]. These initial attempts have led to the identification of problems in knowledge acquisition and representation, as well as integration with existing software [Hall and Deringer 1989].

The reason for such problems is the lack of a comprehensive theory about design, which would serve as the
foundation for the development of building-related data and knowledge representation schemes. Unless such a theory is founded, the various attempts to utilize computers to directly assist in the design process will follow different modeling methods resulting in a variety of incomplete knowledge-based systems, being incompatible with each other as well as with existing drafting and simulation software. Moreover, such systems may prove inappropriate, misleading designers due to hidden subjective preferences and/or forcing them to premature judgements.

**Objective**

The objective of this research is to find out how computers can be used to assist designers, that is, find out which of the activities that designers perform and which of the knowledge they posses can be delegated to a computer and how. The term "design" covers all decision-making activities, such as planning, architectural design and engineering.

**Background**

The attempts to understand and handle design problems in a systematic way have been classified into two generations [Rittel 1972].
First Generation Approach

The first generation was initiated during the Second World War and was based on the concept of "systems analysis" dictating the handling of design and planning problems in a rational, straightforward, systematic way, following specific steps, or phases. There are several variations on what these phases are, however very similar. A system's analyst understands the problem, gathers information, analyzes it, generates solution(s), implements them, tests them, and, if necessary, modifies them. In a particular type of the first generation of systems approach, operations research, the system's analyst finds the best solution by defining the solution space, the constraints and the measure of effectiveness, the latter of which is then optimized.

Second Generation Approach

The second generation was initiated in the late 1960's and was based on the concept of "wicked" problems, dictating the handling of design and planning problems in an argumentative way. Design problems were characterized as "ill-defined," or "wicked" problems, in contrast to the "tame" problems of science and engineering. Wicked problems were characterized by a number of properties which contradict the first generation approach and limits the application of systems analysis and operations research to tame problems. Wicked problems have no
definite formulation. Every formulation corresponds to a statement of the solution and vice versa. There is no stopping rule for wicked problems and the terms "correct" and "wrong" are not applicable to them, while there are no exhaustive, enumerable list of permissible operations. Wicked problems are seen as discrepancies between a situation as is and a situation as it ought to be, and can be considered as symptoms of other wicked problems of a higher order. In addition there is neither an immediate nor an ultimate test to check the appropriateness of solutions. Wicked problems are essentially unique and their treatment is the equivalent of a one-shot operation, for which the wicked problem solver, in contrast to the scientist, has no right to be wrong [Rittel 1972].

Based on the above, the second generation approach claims that the knowledge required for a design or planning problem is not concentrated in any single head. In fact, there is a symmetry of ignorance among the design problem participants, because no one knows better by virtue of her/his education or status and no one knows who knows. In every single step towards the development of a solution a judgement is made on what the case ought to be, which is not based on scientific expertise, that is, there is no scientific planning and there are no experts. As a result, the second generation approach understands the process of wicked-problem solving as an argumentative one and asks for participation by all those affected to determine what is
good and bad, rather than correct and wrong [Rittel 1972].

One of the major contributions of the second generation of systems approaches was the realization of design as an argumentative process, where designers resolve issues (i.e., answer questions) by considering alternative positions (i.e., answers) based on arguments for (advantages) and arguments against (disadvantages) them [Kunz and Rittel 1970]. This design theory has been applied towards the development of Issue-Based Information Systems (IBIS), used to record an argumentative process and organize it so that it is transparent and retraceable. This is achieved through proper relational links among author- and time-stamped IBIS entries. An IBIS may be extended to include additional entries such as references and notes, and can be organized in various ways considering its scope and various secretarial meta-issues, such as the authority to raise issues and the sequence of resolving them [Dehlinger and Protzen 1972; Grant 1977 c].

Issues

While both generation approaches seem applicable up to a point, none represents actual design practice. The claims of the second generation approach about the nature of design problems are correct. It's theoretical view, however, is only partially reflected in actual design practice.

The second generation of systems approach accepts
design as a rational activity, that is, "thinking before acting." Based on the four paradoxes of rationality design becomes impossible [Rittel 1972]. This theoretical claim, however, does not seem valid. Design is possible, since it has always been part of everyday's life.

A more careful consideration is then necessary to resolve these issues and establish a comprehensive design theory that is consistent and compatible with design practice.

Methodology

The observation that both generation approaches to systematically handle design problems do not comply with actual design practice, indicated that an analysis of design practice itself might lead to understanding what it is that designers really do when they design. Rather than trying to perceive design in a desired way, such as a systematic procedure, try to observe and describe it with an "open mind," that is, free from any modeling influence that might hide its true aspects behind assumptions that are hard to see. The same should be true for an analysis of design knowledge as it maps onto the analysis of design practice. Rather than trying to perceive design as one thinks that understands design, try to perceive it as a design product, designed by someone else, e.g., God, and describe, as much as possible, what it is rather than what
it ought to be.

A successful description of the design process and knowledge should be independent from design domain, that is, as generic as possible, representing any designer. Analyzing the design activities and knowledge of one designer and verifying the analysis findings against the design activities and knowledge of others was chosen as the method to follow. This one designer should be available to answer questions as honestly as possible. Moreover, s/he should be able to understand the questions. Since one's self best complies with such requirements, introspection was chosen as the most appropriate research method. The criterion for accepting research findings was the lack of counter-examples, that is, research findings had to be descriptive of and applicable to any designer, independent of design domain and education.

The method for the analysis of the design process called for an analysis of the definition of design into design activities, which were then further and continuously analyzed hierarchically until the very basic, elementary design activities were identified and explicitly defined.

Since the objective of the research is the use of computers to assist designers with the design process, each of the very basic, fundamental design activities are characterized with respect to wether or not it can be delegated to a computer. Moreover, the related design
knowledge is analyzed with respect to origin,
specifiability, reapplicability and openness to
argumentation.
Chapter II. Design Theory

Design has been best defined as an activity aimed at producing a plan, which, if executed, would result in a situation with specific intended properties but without unforeseen, undesired side- and/or after-effects [Rittel 1985]. The statement "if executed" may be omitted, since it can be considered as part of the "plan" concept. In addition, the term "unforeseen" seems inappropriate, since a side- or after-effect has to be foreseen in order to be considered. Moreover, if unforeseen effects are realized during or after the execution of a plan, the activity that produced the plan is still design. However, it may be considered bad design, in which case the term "unforeseen" refers to good and bad design rather than to design in general. After the above considerations, the definition of design can be simplified to: Design is an activity aimed at producing a plan which is expected to lead to a situation with specific intended properties and without undesired side- or after-effects.

Design presupposes a discrepancy between a situation as is and a situation as it ought to be. Three* types of knowledge are required during the design process:

1. Factual knowledge, to specify the as-is situation,

* Two additional types of knowledge complete this design knowledge classification system: the explanatory knowledge which specifies why something was, is, or will be the case, and the conceptual knowledge which specifies the meaning of words [Rittel 1973].

10
2. *Deontic* knowledge, to specify the ought-to-be situation, and,

3. *Instrumental* knowledge, to specify how to transform the as-is situation into the ought-to-be one.

The definition of design refers to it as one activity with *specific intend*, that can be decomposed into three distinct sub-activities:

1. The formulation of the specifications of the ought-to-be situation,

2. The generation of plans to lead from the as-is situation to the ought-to-be one, and,

3. The checking for undesired side- and after-effects.

However, the specifications of the ought-to-be situation are formulated as a set of *performance characteristics* of an ought-to-be situation, while a plan is formulated as a set of *descriptive characteristics* of a will-be situation, whose performance characteristics have to be determined and checked against those of the ought-to-be situation.

Moreover, consideration of identified undesired side- or after-effects is the equivalent of *updating* the performance specifications of the ought-to-be situation. When the will-be performance characteristics do not match the ought-to-be ones, then the will-be descriptive characteristics are *modified*, or the ought-to-be performance characteristics are *degraded*, or the designer gives up. Even if a solution without side- or after-effects has been found, the ought-to-be situation may be *improved* in hope for a *better* solution.
Based on the above considerations, the activity of design can be further decomposed into seven main activities, three initial ones and five that are performed iteratively according to five main decisions (Figure 1). All activities contribute to the development of three sets of specifications:

1. The performance of the ought-to-be situation,
2. The description of the will-be situation, and,
3. The performance of the will-be situation.

The three initial design activities are:

1. Specify the initial ought-to-be performance,
2. Specify the initial is description, and
3. Specify the initial will-be description.

The five iterative design activities are:

1. Update the ought-to-be performance and will-be description,
2. Improve the ought-to-be performance,
3. Degrade the ought-to-be performance,
4. Modify the will-be description, and
5. Determine the will-be performance.

Finally, the five decisions are:

1. Decide whether or not the will-be performance matches the ought-to-be one,
2. Decide whether or not there are undesired side- or after-effects,
3. Decide whether or not there is time / hope for solution,
Figure 1. Design process flow diagram.
4. Decide whether or not there is time / hope for improve-
ment, and

5. Decide whether or not to give up.

The five iterative design activities can be grouped into
three tasks based on their contribution to the development
of the three sets of specifications, that is, the
formulation of the ought-to-be performance, the development
of the will-be description and the determination of the will-
be performance.

**Formulation of the ought-to-be performance**

The specifications of the ought-to-be performance
characteristics is a prescription of what is good and what
is bad, that is a representation of deontic knowledge. Its
development is influenced by life itself and is the most
important and responsible design activity, since the ought-
to-be performance is used to evaluate the appropriateness of
the performance of alternative design solutions.

The performance of a design solution is described
through the use of *performance variables*, based on the
values of which the designer judges the appropriateness of

* The term "performance variable" follows the classification
  of the variables used by designers into performance, design
  and context variables [Rittel 1973]. However, the original
  definition of the term "performance variable" refers to ap-
  propriateness measured on a common scale of goodness, which
  is derived through *transformation functions* applied on the
  values of "other" variables. These "other" variables are the
  performance variables defined herein.
design solutions. The ought-to-be performance is specified as a set of performance criteria, that is conditions on the values of performance variables. The development, then, of the specifications of the ought-to-be situation requires two activities:

1. The determination of the performance variables to be considered, and

2. The formulation of the performance criteria.

The formulation of the ought-to-be performance is also a design problem of its own*. Alternative versions of the ought-to-be performance are considered and are evaluated for the determination of a final one. This metaevaluation requires the formulation of a second-order ought-to-be performance, that of the performance of the ought-to-be performance, which is in turn a design problem of its own, and so on and so forth, resulting in an endless recursion, which makes the formulation of the ought-to-be performance specifications an impossible task†. However, such theoretical claims do not seem to be valid. People start with unquestioned values and build upon them to formulate ought-to-be performance specifications. Moreover, there are usually common values that people accept and follow. The smaller and the more homogeneous a set of people, the more of a possibility for a commonly accepted ought-to-be

* This is one of the four paradoxes of rationality: a model of the design process, realized as a rational activity, needs to include itself [Rittel 1972].
† This is one of the four paradoxes of rationality: one can never start being rational [Rittel 1972].
performance. As one descents the world-continent-country-state-city-community progression, the set of common values among people increases, not only on physiology-related aspects, such as luminous and thermal comfort, but on psychology-related ones like esthetics and ethics. These commonly accepted values are part of what is meant by "common practice."

A performance criterion is a prescription of what is good and what is bad. However, "good" and "bad" are felt. The ought-to-be performance is then more accurately defined as a prescription of what feels good and what feels bad. The more one deliberates, the more s/he attempts to feel the situation that will result from a plan, trying to increase the chances for good feelings and decrease the chances for bad ones during and after the execution of a plan.

Considering the second generation's argumentative model of the design process, issues are raised and resolved based on feelings, while positions and arguments are based on thinking. Feelings develop based on the arguments that support or negate the various positions taken for each issue. The fact that feelings develop based on arguments indicates the impact of rhetoric on the resolution of issues: the effects of an argument depend on the way it is presented.

The difference in planning, architecture, engineering,
and the rest of the problem-solving activities, down to
class- and maze-solving, is the degree of agreement with
respect to performance criteria and their importance. A
class- or maze-solver is not concerned about formulating
ought-to-be performance specifications, since these are
explicitly specified. It is not surprising that class- and
maze-solving are the most favorite problem-solving
applications of Artificial Intelligence... In contrast, a
designer has to determine what the ought-to-be performance
ought to be, a non-rational, extra-logical, and thus non-
computable procedure. The lack of common practice with
respect to what the ought-to-be performance ought to be is
increasingly evident from engineering, to architecture, city
and regional planning, and, finally, general planning.

Based on the above considerations, accepting
"rationality" as "thinking before acting," design is only
partially rational, since it involves feeling as well as
thinking. In fact, design is part of the continuous, linear
process of life (Figure 2). Our acting, as well as the
acting of others, (A), affect the is situation, whose
performance (IP) produces feelings (F) that formulate the
performance of the ought-to-be situation (OP) which directs
our thinking (F) to produce a description of the will-be
situation (WD) based on the description of the is situation
(ID), which is considered towards simulating (S) the
performance of the will-be situation (WP), which, along with
the performance of the ought-to-be situation (OP) produce
Figure 2. A time cycle of the linear process of life, showing that design involves both, feeling and thinking, being an inherent part of life.
feelings (F), which, along with the feelings (F) produced by the performance of the is situation (IP), make us decide the description of the ought-to-be situation (OD), that directs our acting (A), completing one life time-cycle.

It should be noted that the is description and the is performance include all of our past experience. All past will-be descriptions are considered as alternatives for the current ought-to-be one.

The cyclical model of the design process (Figure 1) can be derived from the linear process of life (Figure 2) assuming that the performance of the is situation (IP) does not produce feelings (F) that would make us decide about a description of the ought-to-be situation (OD) that directs our acting (A) away from a specific design problem. This is how we carry design over time, interrupting and/or continuing/terminating it according to how we feel the priorities of our life. All five design decisions of the cyclical model of the design process (Figure 1) are actually life-related decisions that are also affected by considerations that are not necessarily related to the specific design problem that we bring about*. The cyclical model of the design process (Figure 1) can, then, be derived from the linear model of life (Figure 2) by considering life through time (Figure 3) and extracting the life time-cycles

* This is not true for the decision on whether or not the will-be performance matches the ought-to-be one, unless the ought-to-be performance is not completely specified.
**Figure 3.** Life through time. Design is the equivalent of imaginary life.
that are part of bringing about a specific design problem. In fact the model of the design process is the same as that of the linear model of life. Designing is the equivalent of imaginary living, where imaginary potential will-be descriptions take the place of the actual is-description, until the actual is-performance is such that brings us back to reality. Life itself is continuous design process without beginning and end, or one that begins with our birth and ends with our death.

As a conclusion, design is not "thinking before acting," nor "feeling and thinking before acting." Rather, design is feeling and thinking while acting.

**Determination of performance variables**

Performance variables are determined initially based on consideration of the design program objectives, and then during the design process through consideration of undesired side- and after-effects of potential design solutions. A performance variable can be considered either directly, or indirectly, that is, through deliberation towards the identification of a set of new performance variables whose consideration is the equivalent to that of the deliberated performance variable [Musso and Rittel 1967]. Deliberation is actually the equivalent of resolving the issue of how a performance variable should be considered. The initial set of performance variables, for example, can be realized as the result of deliberation on the overall performance. Since
each new performance variable can also be considered through deliberation, performance variables can be realized as a hierarchical, treelike structure, where the root of the tree represents the overall performance and the branches represent the terminal performance variables. Performance variables may operate on continuous or nominal scales. They may take any values, such as numeric quantities, images, sounds, video segments, smells, as well as sets of such values.

Performance variables are determined based on common practice. The higher up on the hierarchy of the ought-to-be performance, the more the universality of a performance variable across design domains. At the top level of the hierarchy, the "overall performance" performance variable may be considered universal across all design domains. The more the deliberation the less the universality across design domains. However, any performance variable may be considered across all domains. The observed differentiation of performance variables as the design domain narrows is the result of specific performance variables considered as unimportant for a specific design domain. Also, the lower down the hierarchy the more variations on the conceptual knowledge with respect to performance variables: the meaning of performance variables varies as the design domain narrows. However, these meanings may be considered additive, that is the different performance variables within narrow design domains may be added to represent a universal tree,
applicable across all design domains. As the design domains narrow down, some performance variables on this universal tree are unimportant and, thus, not considered. As a conclusion, there is no reason to argue against the consideration of any performance variable, since its value may be ignored during the evaluation of potential design solutions.

The ought-to-be performance is not realized in this hierarchical tree structure within a specific domain. Rather, terminal performance variables are considered directly as a set of independent performance variables. This attitude is explained in the following section.

**Formulation of performance criteria**

Continuation of deliberation on the terminal performance variables results on the formulation of performance criteria, that is conditions on the values of performance variables that specify good and bad performance.

Performance criteria may or may not be specified for the initial performance variables. Design is a compromise between what is desirable and what is possible. The design process is the equivalent of exploring what is possible under the specific design context and adjusting performance criteria accordingly, since what is desirable is not usually possible (e.g., zero cost). The performance criteria are formulated throughout the design process. The ought-to-be
performance is either updated, improved, or degraded. It is updated through the identification of side- and after-effects to include new performance variables along with the associated performance criterion. It is improved when a solution has been found but there is still enough time / hope for a better solution. It is degraded when the will-be performance does not comply with the ought-to-be one and there is no time / hope for a solution. This updating, improving, and degrading of the ought-to-be performance continues throughout the design process. The final version of the ought-to-be performance is that of the final will-be one.

As for the determination of performance variables, common practice within a design domain serves as the basis to formulate performance criteria. However, the formulation of performance criteria is not delegable, since it is subjected to personal judgement. Even if a design criterion is based on law, the designer has the right to ignore it or modify it, possibly hoping to "get away with it" or get "special permission" to violate it. Since performance criteria are subjected to personal judgement, they are arguable, that is, different criteria may be formulated based on a single performance variable, each one with arguments to support or negate it.

Performance variables can be measured either on nominal or continuous scales. Performance criteria that are
formulated as conditions on performance variables that operate on continuous scales (e.g., cost) are called herein quantitative, and can be specified with respect to both, performance acceptability as well as performance improvement and degradation. Performance criteria that are formulated as conditions on performance variables that operate on nominal scales (e.g., esthetics) are called herein qualitative, and can be specified only with respect to performance acceptability. Quantitative criteria are specified as acceptable value ranges, while qualitative criteria may be specified as acceptable value sets. However, qualitative criteria are usually not specified at all, in which case the delegation of judgement is impossible.

The judgement on a qualitative criterion is actually a deliberated judgement, in which the branching criteria are not independent, that is the appropriateness of the values of the branching performance variables depends on the values of the rest of the branching performance variables. As an example, consider an image as the value of the performance variables "esthetics," or "quality of the luminous environment." An image can be seen as a luminance distribution, that is values of a set of luminance variables. Since the values of these luminance variables are considered for the evaluation of the performance with respect to esthetics, they are actually branching performance variables operating on continuous scales, thus appropriate for the development of quantitative criteria.
However, the criterion for the appropriateness of the value of a luminance variable depends on the values of the rest of the luminance variables. As a result, if the branching criteria* of a deliberated judgement are not independent, then the deliberated judgement is not delegable since it is the equivalent of a qualitative criterion, which is non-specifiable. However, the branching criteria of a deliberated judgement can never be independent. They are always linked through their relative importance.

The relative importance of performance criteria

A deliberated design criterion represents the ought-to-be performance, which, however, may or may not be possible. If it is not possible, that is if a solution that satisfies all branching criteria cannot be found, then the branching criteria have to be modified. If it is possible, and there are more than one solutions that satisfy all branching criteria, selecting one is the equivalent of further specifying the ought-to-be situation, that is modifying it. Both cases, then, result in modification of the branching criteria. Here is where the relative importance of performance criteria is considered to select which performance criteria to modify and how. In fact, since the relative importance of performance criteria is required for their modifications, its determination is part of the

* Note the reference to design criteria, that is, conditions on the values of performance variables.
formulation of the performance criteria, rather than an independent activity. The relative importance is what makes all branching criteria interdependent, resulting in all deliberated criteria being qualitative ones, that is non-specifiable. The relative importance of performance criteria is felt.

One approach to consider quantitative branching criteria of a deliberated criterion is to consider the multidimensional space defined through the determination of their acceptable value ranges. Such an approach is taken for the consideration of thermal comfort, through the definition of a "comfort zone" [Fanger 1970; ANSI/ASHRAE 1981]. However, this approach specifies performance acceptability but not performance improvement or degradation.

Another approach is to encode the relative importance of the branching performance variables through the definition of an index, thus turning the deliberated criterion into a quantitative one. Such an approach is taken for the consideration of thermal comfort, through the definition of the various thermal comfort indices [Fanger 1970; Gagge et al. 1971; Rohes et al. 1975; ASHRAE 1989] and the consideration of luminous comfort, through the definition of various visual comfort / glare indices [IES 1962; Logan and Siegel 1966; Guth 1966; Fry 1976; IES 1984]. However, the relative importance of the branching criteria is determined by the developers of the indices and not by
the designer. Moreover, they accommodate assumptions about the context of the design problem and are usually limited to small value ranges of the branching performance variables.

An equivalent to this latter approach is applicable to any deliberated performance variable. It requires the specification of transformation functions for each branching variable, that is functions that transform the values of performance variables into appropriateness, or goodness, measured on a common scale for all variables. Moreover, it requires the specification of the relative importance of the branching criteria through the assignment of weighting factors, and the specification of an aggregation function through which the appropriateness of the deliberated performance variable is then computed [Musso and Rittel 1967]. As explained, however, the relative importance is not independent from the acceptable value ranges, or the transformation functions, in this case. This approach can actually be used to demonstrate this dependency, since the weighting factors and the appropriateness values for each of the branching variables appear always together, as a product, and cannot be separated to allow for independent measurement [Rittel 1990]. Weighting factors, specified as percentages, are helpful for computation purposes, however inappropriate, because they represent a reconsideration of the relative importance of design criteria, in addition to the one that led to their formulation.
Based on the above considerations, all deliberated judgements are qualitative criteria and the aggregation of the appropriateness of the values of the terminal performance variables is not delegable and cannot be specified. As a result, designers consider only the values of the terminal performance variables and judge the overall appropriateness of design solutions essentially ignoring the intermediate branching. The evaluation of potential design solutions can be simply expressed as a how-much-of-this-for-how-much-of-that consideration.

The ought-to-be performance is, then, only partially specifiable, thus being only partially delegable. In practice, it is never explicitly specified or considered in any orderly fashion. The formulation of performance criteria is concurrent with the generation of alternative design solutions and their evaluation. The final version of the ought-to-be performance is the performance of the final solution, that is the plan that the designer decides to commit to.

**Development of the will-be description**

Deliberation on a performance variable can be seen as the identification of the variables that affect the deliberated performance variable. A deliberated performance variable can then be seen as "a function of" the branching performance variables. In the same way, the terminal performance
variables are functions of variables that describe the will-be situation. The names of these variables depend on design domain. In architectural design they are variables that describe the building and its context. In automotive design they are variables that describe the automobile and its context.* These variables are called herein control variables, because these are the variables that control the values of performance variables.

Control variables are either design, or context variables [Rittel 1973]. Design variables are those whose values are directly controlled by the designer. Context values are those that the designer decides s/he does not want to control. Since the terminal performance variables are functions of design and context variables, design can be seen as the direct control of the values of the design variables in order to indirectly control the values of performance variables. This indirect control is also partial, since the values of performance variables are affected by the values of the context variables as well.

This extended version of the hierarchical, treelike structure of the ought-to-be performance is in fact the representation of the design domain and is called herein the "design domain tree." The translation of performance variables into design and context variables is the result of

* In fact, design domains are usually specified through reference to a concept such as building, automobile, or city.
the modeling procedures for the determination of the values of performance variables, discussed in the next section. Common practice within a specific design domain refers to the names and values of the performance and control variables that are commonly used within a design domain.

In general, all control variables can be considered as design variables. The consideration of certain control variables as context ones is a design decision, and indicates the level at which the designer wants to attack the problem. The context variables are usually defined based on the design program and set the context of the design problem. As the design process progresses, an increasing number of design variables are considered as the equivalent of context ones, depending on how they influence the values of performance variables and how promising the design solution appears. A design variable that is considered as a context one is called herein pseudo-context variable. At the end, all design variables are considered as pseudo-context ones, since they represent the final design solution, when no further changes are desired. Design can then be seen as the conversion of design variables into pseudo-context ones. This progressive conversion can be seen as the equivalent of design phases, which, however, depend on the specific design problem and cannot be generalized. When a design solution is not possible, pseudo-context variables may be converted back to design ones, to explore new alternatives.
Modifications of the will-be description are made when one or more performance criteria are not met, based on the relationship of design and performance variables for the specific set of values of context variables. When a performance criterion is not met, each design variable that affects it represents an option for modifying the will-be description. However, modifying the value of a design variable does not guarantee that the required performance will be met. Determination of the value of the specific performance variable is required.

Design variables usually affect the values of more than one performance variables. Trying to improve performance with respect to one or more performance variables may, then, result in degrading performance with respect to one or more of the rest of the performance variables. The trade-offs among performance variables due to such interdependencies is what makes performance criteria difficult to meet.

Control variables are usually specified in an object-oriented fashion. The will-be situation is defined in terms of objects, which have attributes and may be children or parents of other objects, and can be seen as a hierarchical, treelike structure. The overall parent object, such as building, automobile, is usually what defines a specific design domain. Usually, the specifications of the will-be situation are initiated by assigning values to the attributes of the overall parent object and its children and
then, progressively proceed down the hierarchy towards attributes of the terminal objects. Since the will-be description is subjected to the available technologies within the design domain, not all attributes may be directly controlled by the designer. Rather, in many cases, the designer is limited to assigning values to objects, rather than their attributes (e.g., glazing type), in which case the values of the attributes of the object are predefined (e.g., glazing transmittance, U-value, shading coefficient, etc.). Objects operate on nominal scales, while attributes operate on continuous ones. However, when the attributes of an object are controlled only through specifying the object itself, then attributes seem to operate on pseudo-ordinal scales. The gradual specification of the will-be situation through the attributes and children of the top objects to the attributes of the terminal objects is the equivalent of the consideration of pseudo-context variables and, as explained, can be seen as a way of realizing design phases.

Since values of objects' attributes affect the possible values for their children attributes (e.g., window width is constrained by the parent wall width), the importance of the specifications of the will-be situation is increasing from the attributes of the terminal objects to those of the overall parent.

The development of the will-be description is then specifiable and, thus, delegable for a given design domain.
tree. The related knowledge consists of the values for
context variables, the available alternative values for
design variables, either objects or attributes, and the
relations among context, design and performance variables.

Creativity

Creativity is associated with the development of the
will-be description and can be defined as the deviation from
common practice within a design domain. Accepting the
arguments that "no knowledge is new knowledge," "new ideas
are follow-ups of fragments of old ideas" and "creativity
is the coming together of existing ideas or knowledge in
new and unforeseen ways" [Koestler 1964; Zwicky and Wilson
1967; Gordon 1971; Wilson 1987], creativity can be seen as
the assignment of values to design variables in
combinations that were never tried before, or the
introduction of new values for objects and/or attributes,
which can be considered as an application of the
morphological approach to enhancing creativity [Zwicky
1969; Grant 1977 a; Grant 1977 b]. The latter approach,
however, may affect the development of the hierarchy
since, as explained, the values of objects and attributes
affect the possibilities throughout the lower levels of
the hierarchy.

Determination of the will-be performance

Since design is aimed at producing a description of the will-
be situation, rather than actually creating it, the only means to determine the will-be performance is to simulate it. Simulation of performance can be achieved in various ways: through drawings, scale models, hand calculations, nomograms, charts, or computer-assisted calculations.

Since the will-be description is specified gradually from the attributes of the overall parent object towards the attributes of the terminal objects, the designer makes decisions before the will-be description is explicitly specified. Simulation procedures to determine the values of performance variables are then used at various degrees of detail. They range from simple, relatively effortless ones for quick estimates of the order of magnitude (e.g., sketches, crude scale models, simplified calculations), to complicated, relatively elaborate ones that are highly detailed and accurate (e.g., working drawings, detailed scale models, sophisticated calculations). Once the order of magnitude of performance appears promising, the design variables that have been addressed become pseudo-context for the specification of the values of the rest of the variables down the hierarchy of the will-be description.

Determination, then of the will-be performance is specifiable within a specific design domain, thus delegable. In fact this is what most of the design education covers, since this is what mostly designers do with to the time allocated to a design project. In many cases, however, this
determination of the will-be performance is almost concurrent with the determination of the will-be description and appears to be the latter, as is the case with sketching, drafting and drawing.

**Design Expertise**

Expertise is defined as "special knowledge derived from experience" [Webster 1970]. The common means of bringing about all three design tasks within a specific design domain is past experience and activities to gain new experience. The activities to gain new experience are confined within the past experience of others (publications, consultation). Experience then is the main source for decision-making knowledge. Since expertise and experience are actually synonyms, an expert designer is a designer who has a lot of experience within her/his design domain. Such an expert should have experience in all, or some of the design activities, within all, or some of the design tasks for her/his design domain.

The term "design domain" indicates that design problems can be classified. This is actually observed in design practice and education. Design problems are classified into design domains, such as building design, ship design, car design, etc. This classification continues within design domains in a hierarchical fashion. The building design domain, for example, is classified
into various sub-domains, such as school design, airport design, hospital design, etc., each of which is further classified in sub-domains. The school design domain, for example, is classified into sub-domains such as kindergarten design, elementary school design, high school design, etc. The more one descends the classification tree, the more the common characteristics among design problems with respect to the names and the values of the variables of the design domain tree, that is the more the available common practice.

Design expertise within a design domain can be classified according to the performance and control variables that define the design domain. A design expert may have experience with respect to one or more performance variables (e.g., thermal comfort, energy consumption), or one or more control variables (e.g., fenestration systems, electric lighting systems), or a combination of performance and control variables (e.g., a fenestration expert with respect to fire-safety and luminous comfort). In fact this is the classification of the various consultants available in design practice. The available expertise, then within a design domain can be mapped on the performance variables, on the control variables, and on the matrix represented by the performance and control variables, called herein the design domain matrix.

The expertise that is mapped on the performance
variables is available in the form of what is recommended for or accepted as appropriate performance. However, this expertise is arguable, because it is subjected to personal judgement. Moreover, it applies only to the identification and formulation of quantitative criteria. Expertise, then, for the formulation of the ought-to-be performance is only partially available and it is arguable. Although the formulation of the ought-to-be performance requires deontic knowledge (i.e., what the performance ought to be), the available expertise is in the form of factual knowledge (i.e., what someone claims that the performance ought to be).

The expertise that is mapped on the control variables is available in the form of the possible values for the control variables, especially for those that are usually considered as design ones through common practice. This expertise is available, however arguable with respect to accuracy. Moreover, when available, it is specifiable.

Finally, the expertise that is mapped on the matrix represented by the performance and control characters is available in two forms:

1. That of the relations among design, context and performance variables, and
2. That of determining the values of performance variables from the values of the control variables that affect them.

This expertise is available in both forms, however arguable
with respect to accuracy.

**Expertise and Creativity**

Accepting the arguments that "no knowledge is new knowledge," "new ideas are follow-ups of fragments of old ideas" and "creativity is the coming together of existing ideas or knowledge in new and unforeseen ways" [Koestler 1964; Zwicky and Wilson 1967; Gordon 1971; Wilson 1987], creativity is strongly related to experience. An experienced designer may be more creative than an inexperienced one because s/he knows alternative values for attributes and their effect on the values of performance variables. However, an experienced designer is also used to operate in certain ways when bringing about design problems, thus may be less innovative than a novice designer who is not used in specific procedures and/or considerations. The higher on the design domain tree a design problem is attacked, the more the deviation from common practice, and the more the possibilities for innovation, however failure as well, since innovation carries the risk of unknown consequences. The lower on the design domain tree a design problem is attacked, the less the deviation from common practice, and the less the possibilities for innovation and failure.
Chapter III. Computer-Aided Design

Based on the design theory presented in chapter II, the potential of computer-aided design is limited by the specifiability and delegateability of design activities. Design activities that are specifiable and delegable can be automated. However, there are also possibilities to use computers to assist with specifiable but nondelegable design activities. Integration of the delegable activities along with appropriate user interface for the nondelegable ones may contribute towards the development of a comprehensive, computer-based Design Support Environment (DSE). Proper modeling of the design process should be independent of design domain and provide means to incorporate all of the specifiable knowledge associated with any specific design domain.

It is important that a model of the design process does not become a model of "the" or "a" designer. A model of "the designer" assumes that designers operate in a specific way (for example they know the relative importance of design criteria a-priori and can even specify it using weighting factors) which is usually compatible with algorithms that offer opportunities not only for design automation, but optimization as well. A model of "a designer" assumes that one designer's ought-to-be performance is the ought-to-be performance for all designers, thus automating decisions based on "hidden"
criteria (from handbooks, standards or "experts"), which, however, may not match the designer's preferences for a specific project, or in general. Design tools that are based on such models may prove ineffective, forcing designers to premature judgements and/or misleading them due to hidden subjective preferences.

Designers assign values to design variables by monitoring the resulting values of performance variables and translating them into advantages and disadvantages of will-be descriptions. Considering the argumentative model of the design process, proposed by the second generation of systems approach, these value assignments can be seen as steps towards the resolution of an issue (the ought-to-be description), where alternative positions (the will-be descriptions) are evaluated considering the arguments that support (advantages) and negate (disadvantages) them.

Any value assignment can become an issue, for which conditional positions may be taken, that is variable positions that depend on the value (resolution) of one or more variables (predecessor issues). The conditions of a conditional position may be considered as supporting or negating arguments, depending on whether or not they were met, respectively.

Design can then be realized as argumentation towards the assignment of values to variables through delegable and nondelegable activities. It is interesting to note that the
delegable activities involve only thinking, while the nondelegable ones involve feeling as well. All design activities contribute to the specification of the ought-to-be performance, the will-be description, and the will-be performance. A computer-based model of the design process should then provide means to represent these specifications (data), automate the delegable activities (processes), and interface with designers (input/output) for the consideration of the nondelegable activities (feelings).

**Formulation of the ought-to-be performance**

The formulation of the ought-to-be performance consists of the determination of the names of performance variables and the formulation of design criteria, that is, conditions on the values of performance variables.

**Determination of performance variables names**

The names of performance variables are determine partially based on the design program and partially through the identification of side- and after-effects. Although these activities are nondelegable, performance variables are specifiable, however arguable. Performance variables result through deliberation of the "overall performance" and can be grouped in a hierarchical structure. In fact, since the aggregation of deliberated judgements is non-specifiable, only the terminal performance variables are required, which can be structured as a list. A hierarchical structure,
however, reflects the involved deliberation, allows for global enabling and disabling of terminal performance variables, and supports possible argumentation involved in deciding what the performance variables ought to be. An IBIS can be used to record and organize the various arguments that support and/or negate the consideration of a specific performance variable for a specific design problem, or in general.

Performance variables are the foundation blocks for the development of a DSE. A DSE refers to a specific design domain, which is defined by a number of performance variables and the control variables that affect them. Control variables are determined based on the methods used to determine the values of performance variables. Unless the value of a performance variable can be computed, its contribution to a computer-based DSE is minimal, that of reminding the user about its existence so that it is considered through different means.

The set of performance variables considered in a DSE is limited to those that are initially specified through the definition of the design domain that the DSE addresses. However, dynamic declaration of additional performance variables is also possible. This, however, would require a specific protocol for the specification of the computational procedures to be used for the determination of their values and the possible introduction of new control variables along
with their relations to the already declared ones. In absence of such protocols, a DSE can handle only the specific set of performance variables that represent the original design domain that the DSE was developed for.

Formulation of performance criteria.

In the beginning of the design process only few performance variables are associated with performance criteria, that is, those that describe the intended specific properties. The rest of the performance variables are associated with performance criteria at the time of their identification, that is, through checking for undesired side- and after-effects.

The formulation of performance criteria is performed throughout the design process. The initial criteria are based on the design program and represent the "intended specific properties." The rest are formulated at the time of the introduction of the associated performance variable and represent the "undesired side- and after-effects." In either case, performance criteria are continuously modified representing the improvement and degradation of the ought-to-be performance.

While the formulation of both types (quantitative and qualitative) of criteria is nondelegable, quantitative criteria are specifiable through a minimum and/or maximum acceptable value for the associated performance variable.
However, such minima and maxima are dynamic, that is they may change through the design process. As explained, the final version of the ought-to-be performance is that of the final will-be performance. Moreover, their values may depend on the design context.

If all performance criteria are quantitative with specified acceptable value ranges, then design can be automated with respect to finding one or more solutions, that is combinations of values for design variables, that result in all performance variables having values within their acceptable value ranges, as for the case of the thermal comfort zone, presented in the previous chapter. This possibility for design automation is discussed in the section that refers to the determination of the will-be description.

Although the activity of formulating design criteria cannot be delegated, it is specifiable and can be partially recorded as positions to the issues of assigning minima and maxima acceptable values for performance variables. Different performance criteria for the same performance variable would result either in conditional positions, when taken by the same designer, or in conflict of opinions, when taken by different designers, that is issues that need to be resolved among the participants of the design process. In this way, computers may be used to record and then duplicate, that is, learn, the specifiable subjective
preferences of designers. The effectiveness of such learning possibilities, however, depends on the users' willingness to specify the conditions for their positions.

Usually, common practice within a design domain dictates the consideration of certain performance criteria. Considering the above "learning" scheme, the proper alternative of "a designer" models is to record subjective preferences, such as those found in handbooks, codes, standards, and opinions of "experts," as positions taken for the related issues. In this way, designers may consider them towards the resolution of the related issues, rather than having them imposed, possibly without even being aware of it.

Development of the will-be description

The will-be description, as well as the is-description, are specified through the control variables, that is context and design variables, whose determination is based on the specification of the procedures used to determine the values of performance variables. However, additional context variables may be defined dynamically to support the consideration of conditional positions, as explained in the previous section.

The will-be description is specified through the names of design and context variables, which are either objects or objects' attributes. An object-oriented data structure can
then be used to represent the will-be description in terms of objects with attributes, which may be parents or children of other objects. The parent-children relations support the specification of constraints related to the values of control variables.

The modification of the will-be description is the result of unsatisfactory performance with respect to one or more of the performance criteria considered. When performance with respect to a quantitative performance criterion needs to be improved, the value of the particular performance variable should either increase, or decrease. Each performance variable is associated with a list of design and context variables that affect it. Increasing or decreasing the value of a performance variable can then be translated into increasing and/or decreasing the value of one or more design variables that affect it, assuming that they operate in continuous or pseudo-ordinal scales. The available value options of such design variables may then be considered as positions which are automatically supported by the performance variables that are affected positively and negated by the performance variables that may be affected negatively.

When control variables are measured on nominal scales, such as those that are objects, then increasing and decreasing their values does not make sense. In such cases, the alternative options is to either try different values at
random, or select values based on the desired direction of their attributes. Conventional databases can be used to store such control variables' value options, along with the values of their attributes. Objects can then be sorted and/or searched based on their attributes that operate on continuous or pseudo-ordinal scales. Databases can be expanded through addition of new value options, as long as values are provided for all of the required attributes. Alternative value options for control variables represent one of the non-arguable types of knowledge within a design domain, which may be considered as unquestionable expertise.

The possible values for control variables may also be limited by the modeling capabilities of the simulation procedures used to determine the will-be performance as well as by possible constraints due to the values of other control variables. Excluding creativity and assuming that the determination of the will-be performance is clearly specified through performance variable functions, it is possible to automate the development of the will-be description. This is possible only with respect to quantitative performance criteria. It becomes the problem of selecting the proper combination of values for design variables that will result in values of performance variables within the specified acceptable value ranges. This automation is useful only with respect to determining if one or more solutions are possible, that is, if one or
more combinations of values for design variables satisfy the quantitative performance criteria under the specified context. If more than one solutions satisfy the performance criteria, or if no solution can be found, then the performance criteria need to be modified. Modification of performance criteria, however, is a nondelegable activity. As result, "optimization" is impossible, since the relative importance of performance criteria is non-specifiable.

This possibility for automation is useful only when there is at least one solution that satisfies the performance criteria, the search for which may require an exhaustive enumeration of the combinations of values for design variables. However, the solution space may be impractically large, since each design variable considered increases the number of possible solutions by multiplying it by the number of its value options. Moreover, there are usually terminal qualitative criteria that require direct evaluation by the designer. Since the consideration of terminal qualitative criteria has to be excluded in an automatic search for solution(s) it is possible that solutions with very good performance with respect to the qualitative criteria are ignored. Automation, then, is theoretically possible for quantitative criteria, however of questionable practical usefulness.

The most promising possibility to use computers to
assist with the development of the will-be description is through the relations among performance, design and context variables. These are "included" in the simulation algorithms used to compute the values of performance variables. However, they may also be explicitly specified in the form proportionality relations or "rules of thumb" from "experts." The latter approaches, however, may prove to be risky. Proportionality relations are not always constant, since they may depend on the values of other control variables. Moreover, rules of thumb may incorporate assumptions that are hard to see, as the "shading of the south facade of buildings," which assumes that the building is on the North hemisphere...

When performance with respect to a performance criterion needs to be improved, its relations with the design variables can be used to indicate the possible alternative actions that will contribute to its improvement. This is another non-arguable type of knowledge within a design domain, which may be considered as unquestionable expertise.

**Determination of the will-be performance**

Assuming that all performance variables are specifiable functions of control variables, the values of which have already been determined through the specification of the will-be description, the determination of the will-be
performance appears to be a fully delegable activity. However, there may be more than one function to determine the value of one or more performance variables, raising the issue(s) of which one(s) to be used.

A performance simulation model, that is, a performance variable function, is characterized by modeling capabilities, time requirements, and accuracy, all of which depend on the model's (function's) inherent assumptions. Time requirements and accuracy are usually proportional, while modeling capabilities refer to different values of design and context objects and objects' attributes that the simulation model has been designed for.

The simulation models that are used in the initial stages of the design process are usually fast and inaccurate, since they are usually meant to provide an estimate, that is, order of magnitude, for the values of performance variables. As explained, a design solution is a compromise between what is desirable and what is possible, both of which are determined throughout the design process. These "order of magnitude" simulations are meant to initiate the process of understanding what is possible under a specific design context.

The determination of the will-be performance is, then, delegable through the specification of appropriate simulation algorithms, that is, functions of design and context variables. As explained in the previous section, it
is one of the non-arguable types of knowledge within a
design domain, which may be considered as unquestionable
expertise.

A computer-based Design Support Environment (DSE)

Based on the above considerations, it is possible to use
computers to support design within any design domain. The
currently available computer-modeling concepts are adequate
to model all design activities and represent all related
data and knowledge. However, they need to be integrated in a
coherent environment under an appropriate user interface.

Structure

The proposed DSE has six major modules that operate on
two common data structures, one used to describe the ought-
to-be and the will-be performance and one to describe the
will-be description. The six modules of the proposed DSE are
(Figure 5):

1. An Issue-Based Information System (IBIS),
2. A Rule-Based Information System (RBIS),
3. A Performance Simulation Algorithm Library (PSAL),
4. A Design Project Archival Database (DPAD),
5. A Control Value Options Database (CVOD), and

A general IBIS is implemented to record and organize
positions taken for the formulation of the ought-to-be
Figure 4. The structure of the DSE.
performance, that is, the names of performance variables and the associated acceptable value ranges. A specific IBIS is created for each design project, to record and organize the development of the will-be description, using the values of performance variables as supporting and/or negating arguments. Conditional positions for both types of IBIS (e.g., the minimum and maximum acceptable values for work-plane illuminance may depend on task) are recorded using the RBIS, which is also used to specify the relations among performance, design, and context variables, which may require accessing the PSAL. The RBIS is also used to specify constraints for the values of the control variables, that is conditions among the values of objects and/or objects' attributes (e.g., the window width should be less than or equal to the width of its parent wall).

The project-specific IBIS is also used to record the design process with respect to "who decided what, when, and why." The DPAD is actually a collection of all project-specific IBIS created through the use of the DSE. The CVOD contains data for the various objects and their attributes, that is design or context variable value options, such as product catalogs and climatic data. The DPAD and CVOD are implemented under a multimedia environment and are complemented by the EHIS. All three components are linked through the names of the performance, design and context variables considered within the specific design domain.
Operation

The operation of the DSE relies entirely on the user, that is the designer. At any point, the user can access and edit any of the data related to the various components of the DSE, whether s/he is working on a specific design project, or not. Through the general IBIS, the DPAD, the CVOD, and the EHIS, the DSE can be used as an educational tool.

When working on a specific project, the user can consider any of the performance variables that have already been declared to the DSE, or introduce new ones. New performance variables require the specification of the function(s) to be used for the determination of their value. The introduction of a new function may introduce new control variables, i.e., new attributes and/or objects for the objects that have already been declared to the DSE. Note that the introduction of a new performance variable is the equivalent of expanding the DSE's design domain. Performance variables may be considered or ignored at any time during the design process, as long as at least one is being considered.

Once a performance variable is being considered, the associated quantitative performance criteria may be specified as acceptable value ranges. However, this is not required, since it is used only for "design automation." As explained, the specification of the performance
variables to be considered and their associated performance criteria are supported by the general IBIS which includes various positions based on handbooks, standards and codes. The user can take her/his own position and argue for and/or against any of the already recorded positions.

Control variables are introduced automatically based on the performance variables considered. All control variables are considered by default as design variables. The designer may declare the context ones for which s/he has to assign values, which, however, must comply with the available VOD entries. The assignment of values to design variables may be manual or automatic.

In the manual mode the designer assign values directly, in the same way that s/he assigned values to the context variables. S/he can then ask for the determination of the values of the performance variables. After the initial specification of the will-be description, the DSE can operate in two different ways: 1) the designer may specify which performance s/he wants to improve and ask the DSE to indicate possibilities, that is which design variables to manipulate and how, and 2) the designer may propose new values for design variables and ask the DSE to indicate the resulting effects on the values of the performance variables considered.

In the automatic mode, the DSE considers the available
value options and selects a combination that satisfies the specified quantitative performance criteria. If a solution cannot be found, the performance conflicts are presented to the designer, who may degrade the ought-to-be performance, that is enlarge the acceptable value range of one or more performance variables, and ask again for a solution. If a solution is found, it is presented to the designer, who may improve the ought-to-be performance, that is narrow the acceptable value range of one or more performance variables, and ask again for a solution.

The degrading and improving of the ought-to-be performance does not need to be explicit, since there is ignorance on what is possible. Rather, the designer indicates the performance that may be degraded, or the performance that must be improved, without specifying new limits for the associated acceptable value ranges.

Finally, the designer has access to the DPAD, CVOD, and EHIS at any point. Access to these components may be specific, that is based on specific values or conditions of performance and control variables, or general, for browsing purposes.

Development and growth.

There are two ways that the DSE grows with respect to its capabilities and knowledge. One way is through further development for the consideration of additional performance
variables, which, is expected to be integrated into the BDSE's operational capabilities. The other way is through the use of the DSE.

The general IBIS can be extended at any point by entering new positions for existing issues and new arguments for existing positions. New issues are raised only through the introduction of new performance variables along with new associated control variables. Moreover, the designer's actions towards improving performance are translated into positions for the minima and maxima acceptable values of the associated performance variables. When different positions are taken by the same user, they are treated as conditional positions and the user is prompted to specify them using the values of "known" control variables, or "new" context variables. In this way, the system learns the subjective preferences of its users. Differences in opinions among design participants can be thus identified by DSE, in which case it asks the users to resolve the conflict and agree on one position for the specific design project.

The CVOD can be extended at any point by entering new values for the existing objects along with the values of their attributes. Finally, the EHIS can be extended as well, however through development efforts rather than automatically through the operation of the DSE. The existing entries of the EHIS may be linked to the DPAD and CVOD entries, as well as to the general IBIS for reference.
purposes.

The computer-based DSE described in this section is partially illustrated in the Appendix through the presentation of the structure and operation, of a demonstration prototype for a limited design domain that addresses the design of fenestration and electric lighting systems of office spaces considering comfort, energy requirements and cost.
Chapter V. Conclusions

The shortcoming of both generations of systems approach was the attempt to model design as a rational activity. The first generation assumed it without even recognizing it. The second generation recognized it, identified the four paradoxes of rationality, however, still assuming that design is a rational activity. The conclusion of this dissertation is that design is only partially rational, since it involves feeling, as well as thinking. Design is the equivalent of imaginary life, however affected by real life, which in turn can be seen as a continuous design process, where our thinking influences our behavior through continuous breaks of instinct behavior.

Design is modeled as an argumentative process through which designers think of possible future situations towards a desired performance, which is continuously adjusted according to what appears possible. When more than one performance criteria are considered for the evaluation of alternative design solutions, the delegation of judgement is not possible, since the relative importance of performance criteria is not specifiable.

Computers can "remember" and "think," but not "feel." They can be used to store the specifiable design knowledge, automate the delegable design activities, and, through appropriate user interface, assist designers to bring about
the nondelegable design activities. Inappropriate modeling of the design process may result in ineffective design tools that force designers to premature judgements and/or mislead them due to hidden subjective preferences.

Computers offer possibilities to record the specific preferences of designers. Moreover, they can be used to maintain an archival record of the design process with respect to who decided what, when, and why.

Non-arguable design knowledge appears only in the form of possible alternative descriptive characteristics, which can be used to specify future situations and simulate their performance. Through appropriate user interface, computers can be used to indicate alternative courses of action that will modify a specific description towards one that will meet specific quantitative criteria.

All of the above possibilities can be integrated into a computer-based design support environment for a specific design domain, that can grow through use, continuously adjusting to the specific preferences of its users.
References

ANSI/ASHRAE 1981


ASHRAE 1989


Degelman and Kim 1988


Dehlinger and Protzen 1972


Fanger 1970


Fanger and Valbjorn 1979


Fry 1976


Gagge et al. 1971

Gordon 1971


Grant 1977 a


Grant 1977 b


Grant 1977 c

Grant, D. "How to use the IBIS as a procedure for deliberation and argument in environmental design and planning." Design Methods and Theories, Vol. 11, No. 4, October-December 1977, pp. 185-220.

Guth 1966


Haberl et al. 1989


Hall and Deringer 1989


IES 1962


IES 1984


Kunz and Rittel 1970

Kunz, W. and Rittel, H. W. J. "Issues as elements of information systems." Working Paper 131, Institute of

Logan and Siegel 1966


Musso and Rittel 1967


Rich 1983


Rittel 1972


Rittel 1973


Rittel 1985

Rittel, H. W. J. Unpublished lectures in Design Theories and Methods, Department of Architecture, University of California, Berkeley, California, 1985.

Rittel 1990


Rohes et al. 1975


Ruberg and Cornick 1988

Selkowitz et al. 1982

Selkowitz et al. 1986

Shuman et al. 1988

Sullivan et al. 1988

Tuluca et al. 1989

Webster 1970

Wilson 1987
Wilson, F. "How we CRE." Lighting Design + Application, February 1987.

Zwicky 1969

Zwicky and Wilson 1967
Appendix. A DSE Demonstration Prototype

Simplified versions of all DSE components have been integrated in a DSE prototype whose purpose is to validate and demonstrate the proposed theory and modeling concepts, as well as to contribute to further theoretical and applied research.

The DSE prototype addresses the design of fenestration and lighting systems of office spaces, considering comfort, energy and cost, through eight performance and twenty-one design/context variables. The eight performance variables are (Figure 5):

1. Glare Index (dimensionless).
2. Thermal Comfort Index (dimensionless).
3. Annual Fuel Requirements (therms).
4. Annual Electricity Requirements (kwh).
5. Electricity Peak Demand (w).
6. First Cost ($/m²).
7. Operating Cost ($/m²).
8. Life Cycle Cost ($/m²).

The twenty one design/context variables are properties of seven "objects," as follows, (pseudo-design variables are in italics) (Figure 6):

Site:

1. Name (Madison WI, Lake Charles LA).
2. Gas cost ($/therm).
3. Electricity cost ($/kwh).

4. Interest rate (%).

Building:

5. Type (office space).

6. Payback period (yrs).

Room:

7. Width (m).

Lighting System:

8. Power density (w/m²).

9. Illuminance (lux).

10. Daylight controls (Y,N).

11. Cost ($/m²).

Window Wall:


Window:

13. Width (m).

14. Height (m).

15. Shading (Y,N).

16. Cost ($/m²).

Glazing:

17. Type (single, double, triple; clear, absorbing, reflecting, low-e).

18. U-value (w/m²/°C).

19. Shading coefficient (%).

20. Visible transmittance (%).

21. Cost ($/m²).
The performance simulation procedures are based on simplified algorithms (regression expressions) that were derived from statistical analyses of the results of multiple parametric hourly simulations of the operation of an office building prototype [Sullivan et al. 1988].

The user interface is designed to allow appropriate interaction for the bringing about of the nondelegable design activities. Moreover, it is also meant to allow for the demonstration prototype to be used as a research tool, providing means of monitoring its operation through a visible performance/context/design matrix (Figure 7).

Each performance variable is associated with a minimum and a maximum acceptable values, as well as a current value that corresponds to the current values of the control variables. The minima and maxima values can be set directly, or be selected through an IBIS that contains conditional positions based on standards and codes (Figure 8). When a minimum or maximum value is set directly, then the user may optionally be informed for the existence of negating arguments. Finally, each performance variable is associated with a goal field, through which s/he can specify the desired direction for improvement (Figure 9). When a goal is indicated, the matrix shows the available options, that is, it indicates which design variables may be altered and in what direction (Figure 10).

Each control variable is associated with a current
value and a proposed value. Proposed values can be directly specified only for design variables (not for context, nor for pseudo-design ones), using popup menus (Figure 11). When a value is proposed, the matrix shows the effects of the change for each performance variable affected (Figure 12). The values of objects' properties are stored in databases which can be reviewed and sorted at any time (Figure 13). Pseudo-design variables can be declared as design ones, for the exploration of "new alternatives" (Figure 14).

A new solution is declared when at least one proposed value has been specified and the performance with respect to at least one performance variable is requested (Figure 15). Solutions are stored chronologically (date and time stamp) as sets of values for all control variables, and are associated with the name of the user that declared them. A new solution can be initiated from any of the already known solutions. As a result, each solution has one parent solution and may have any number of children solutions. In addition to chronologically moving through solutions, the user may go directly to the parent or to any of the children of the currently displayed solution (Figure 16).

All performance and control variables are linked to an electronic handbook information system, which serves the purpose of providing information about the particular variable (Figure 17). Moreover, performance variables may be ignored at user request (Figure 18).
Upon completion, each project is stored in a design project archival database, from where they can be recalled for viewing or further modifications.

**Figure 5.** The performance variables considered in the DSE demonstration prototype, presented as outcome of deliberation on the overall performance.
<table>
<thead>
<tr>
<th>Site</th>
<th>Name</th>
<th>Fuel Cost</th>
<th>Electricity Cost</th>
<th>Interest Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Building</td>
<td>Width</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Room</td>
<td>Type</td>
<td>Payback Period</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lighting System</td>
<td>Power Density</td>
<td>Work Plane Illuminance</td>
<td>Controls</td>
<td>Cost</td>
</tr>
<tr>
<td>Window Wall</td>
<td>Orientation</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Window</td>
<td>Orientation</td>
<td>Width</td>
<td>Height</td>
<td>Shading</td>
</tr>
<tr>
<td>Glazing</td>
<td>Type</td>
<td>U-Value</td>
<td>Shading Coefficient</td>
<td>Visible Transmittance</td>
</tr>
</tbody>
</table>

**Figure 6.** The control (design and context) variables considered in the DSE demonstration prototype, presented as objects with attributes (outline style indicates nominal scales and italic style indicates pseudo-ordinal scales).
<table>
<thead>
<tr>
<th>Performance Variables</th>
<th>Minima</th>
<th>Current Values</th>
<th>Maxima</th>
<th>Goals</th>
<th>Solutions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Glare Index</td>
<td>1</td>
<td>11.550085</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Thermal Comfort Index</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Annual Fuel Requirements [therms]</td>
<td>-19.054268</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Annual Electricity Requirements [kwh]</td>
<td>1244.379119</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Peak Electricity Demand [w]</td>
<td>575.008794</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>First Cost [$/sqm]</td>
<td>18.555</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Operating Cost [$/sqm]</td>
<td>80</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Life-Cycle Cost [$/sqm]</td>
<td>820.474171</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Objects</th>
<th>Properties</th>
<th>Current Values</th>
<th>Proposed Values</th>
<th>Options / Effects</th>
</tr>
</thead>
<tbody>
<tr>
<td>Site</td>
<td>Name</td>
<td>Madison WI</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Building</td>
<td>Type</td>
<td>Office Building</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Room</td>
<td>Payback Period [yrs]</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lighting System</td>
<td>Power Density [w/sqm]</td>
<td>7.535</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Illuminance [lx]</td>
<td>300</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Window Wall</td>
<td>Orientation</td>
<td>North</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Window</td>
<td>Width [m]</td>
<td>5.040</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Glazing</td>
<td>Height [m]</td>
<td>2.19456</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cost [$/sqm]</td>
<td>10.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Type</td>
<td>Single Clear</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>U-Value [w/sqm/°C]</td>
<td>0.957</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Shading Coefficient [%]</td>
<td>0.949</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Visible Transmittance [%]</td>
<td>0.084</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cost [$/sqm]</td>
<td>1.0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 7. The main screen of the user interface of the DSE prototype.
### Performance Variables

<table>
<thead>
<tr>
<th>Variable</th>
<th>Minima</th>
<th>Current Values</th>
<th>Maxima</th>
<th>Goals</th>
<th>Solutions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Glare Index</td>
<td>IES</td>
<td>15</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Thermal Comfort Index</td>
<td>ASHRAE</td>
<td>97</td>
<td>1.19</td>
<td>94</td>
<td></td>
</tr>
<tr>
<td>Annual Fuel Requirements (therms)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Annual Electricity Requirements (kwh)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Peak Electricity Demand (w)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>First Cost ($/sqm)</td>
<td>18.555</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Operating Cost ($/sqm)</td>
<td>0.80</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Life-Cycle Cost ($/sqm)</td>
<td>620.474171</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Objects Properties

<table>
<thead>
<tr>
<th>Site</th>
<th>Madison WI</th>
<th>Current Values</th>
<th>Proposed Values</th>
<th>Options / Effects</th>
</tr>
</thead>
<tbody>
<tr>
<td>Building</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Office Building</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Payback Period (yrs)</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Width (m)</td>
<td>6.096</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lighting System</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Power Density (w/sqm)</td>
<td>7.535</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Illuminance (lx)</td>
<td>500</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Daylight Controls</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cost ($/sqm)</td>
<td>20</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Window Val</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Orientation</td>
<td>North</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Width (m)</td>
<td>3.048</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Height (m)</td>
<td>2.19456</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Shading</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cost ($/sqm)</td>
<td>10.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Glazing</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Single Clear</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>U-Value ($/sft2/°F)</td>
<td>0.957</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Shading Coefficient (%)</td>
<td>0.949</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Visible Transmittance (%)</td>
<td>0.884</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cost ($/sqm)</td>
<td>1.0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Figure 8.** The specification of minima and maxima acceptable values for performance variables is optional, aided by the use of an Issue-Based Information System (IBIS) that may also include positions based on codes and standards.
Figure 9. The specification of the desired direction for the values of performance variables.
<table>
<thead>
<tr>
<th>Performance Variables</th>
<th>Minima</th>
<th>Current Values</th>
<th>Maxima</th>
<th>Goals</th>
<th>Solutions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Glare Index</td>
<td>11.550085</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Thermal Comfort Index</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Annual Fuel Requirements [therms]</td>
<td>-19.654260</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Annual Electricity Requirements [kwh]</td>
<td>1214.579119</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Peak Electricity Demand [W]</td>
<td>575,000,794</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>First Cost [$/sqm]</td>
<td>18.555</td>
<td></td>
<td></td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td>Operating Cost [$/sqm]</td>
<td>00</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Life-Cycle Cost [$/sqm]</td>
<td>620,474,171</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Objects</th>
<th>Properties</th>
<th>Current Values</th>
<th>Proposed Values</th>
<th>Options / Effects</th>
</tr>
</thead>
<tbody>
<tr>
<td>Site</td>
<td>Name</td>
<td>Madison WI</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Building</td>
<td>Type</td>
<td>Office Building</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Room</td>
<td>Payback Period [yrs]</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lighting System</td>
<td>Width [m]</td>
<td>6.096</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Window Wall</td>
<td>Power Density [w/sqm]</td>
<td>7.535</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Window</td>
<td>Illuminance [lux]</td>
<td>500</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Window</td>
<td>Daylight Controls</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Window</td>
<td>Cost [$/sqm]</td>
<td>20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Window</td>
<td>Orientation</td>
<td>North</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Window</td>
<td>Width [m]</td>
<td>5.040</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Window</td>
<td>Height [m]</td>
<td>2.19456</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Glazing</td>
<td>Shading</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Glazing</td>
<td>Cost [$/sqm]</td>
<td>10.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Glazing</td>
<td>Type</td>
<td>Single Clear</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Glazing</td>
<td>U-Value [w/sqm/K]</td>
<td>0.957</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Glazing</td>
<td>Shading Coefficient [%]</td>
<td>0.949</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Glazing</td>
<td>Visible Transmittance [%]</td>
<td>0.684</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Glazing</td>
<td>Cost [$/sqm]</td>
<td>1.0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Figure 10.** Automatic indication of the control variables that affect the values of performance variables towards the specified desired direction.
**Figure 11.** The specification of proposed values for design variables.
**Figure 12.** Automatic indication of the effects on the values of performance variables by the proposed values for design variables.
<table>
<thead>
<tr>
<th>Glazing Type</th>
<th>Visible Transmittance [%]</th>
<th>U-Value [W/sqm/°C]</th>
<th>Shading Coefficient [%]</th>
<th>Cost [$/sqm]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single Clear</td>
<td>0.884</td>
<td>0.957</td>
<td>0.949</td>
<td>3.23</td>
</tr>
<tr>
<td>Single Absorbing</td>
<td>0.536</td>
<td>0.963</td>
<td>0.715</td>
<td>3.23</td>
</tr>
<tr>
<td>Single Reflective</td>
<td>0.297</td>
<td>0.651</td>
<td>0.412</td>
<td>6.46</td>
</tr>
<tr>
<td>Single Low-e</td>
<td>0.832</td>
<td>0.644</td>
<td>0.852</td>
<td>8.61</td>
</tr>
<tr>
<td>Double Clear</td>
<td>0.786</td>
<td>0.526</td>
<td>0.815</td>
<td>12.92</td>
</tr>
<tr>
<td>Double Absorbing</td>
<td>0.477</td>
<td>0.533</td>
<td>0.572</td>
<td>12.92</td>
</tr>
<tr>
<td>Double Reflective</td>
<td>0.271</td>
<td>0.419</td>
<td>0.320</td>
<td>16.15</td>
</tr>
<tr>
<td>Double Low-e</td>
<td>0.741</td>
<td>0.340</td>
<td>0.665</td>
<td>19.38</td>
</tr>
<tr>
<td>Triple Clear</td>
<td>0.701</td>
<td>0.359</td>
<td>0.710</td>
<td>21.53</td>
</tr>
<tr>
<td>Triple Absorbing</td>
<td>0.424</td>
<td>0.362</td>
<td>0.486</td>
<td>21.53</td>
</tr>
<tr>
<td>Triple Reflective</td>
<td>0.244</td>
<td>0.308</td>
<td>0.270</td>
<td>24.76</td>
</tr>
<tr>
<td>Triple Low-e</td>
<td>0.666</td>
<td>0.258</td>
<td>0.578</td>
<td>27.99</td>
</tr>
</tbody>
</table>

**Figure 13.** The glazing database of the DSE demonstration prototype.
**Figure 14.** The specification of the consideration of pseudo-design variables as design ones.
**Figure 15.** The specification of a "new" solution through request for the determination of the value of a performance variable.
<table>
<thead>
<tr>
<th>Performance Variables</th>
<th>Minima</th>
<th>Current Values</th>
<th>Maxima</th>
<th>Goals</th>
<th>Solutions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Glare index</td>
<td>1.0</td>
<td>11.550885</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Thermal Comfort Index</td>
<td>1.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Annual Fuel Requirements (therms)</td>
<td>-19.854268</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Annual Electricity Requirements (kwth)</td>
<td>1214.379119</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Peak Electricity Demand [w]</td>
<td>575.000794</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>First Cost [$/sqm]</td>
<td>18.555</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Operating Cost [$/sqm]</td>
<td>90</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Life-Cycle Cost [$/sqm]</td>
<td>820.474171</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Objects</th>
<th>Properties</th>
<th>Current Values</th>
<th>Parent</th>
<th>Children</th>
</tr>
</thead>
<tbody>
<tr>
<td>Site</td>
<td>Name</td>
<td>Madison WI</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Building</td>
<td>Type</td>
<td>Office Building</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Room Payback Period [yrs]</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Width [m]</td>
<td>6.096</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lighting System</td>
<td>Power Density [w/sqm]</td>
<td>7.535</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Illuminance [lux]</td>
<td>300</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Daylight Controls</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cost [$/sqm]</td>
<td>20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Window Wall</td>
<td>Orientation</td>
<td>North</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Window</td>
<td>Width [m]</td>
<td>3.048</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Height [m]</td>
<td>2.19456</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Glazing</td>
<td>Shading</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cost [$/sqm]</td>
<td>10.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Type</td>
<td>Single Clear</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>U-Value [w/sqm/°C]</td>
<td>0.957</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Shading Coefficient (%)</td>
<td>0.949</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Visible Transmittance (%)</td>
<td>0.884</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cost [$/sqm]</td>
<td>1.0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Figure 16.** The navigation through the potential design solutions is either chronological (through the left and right pointing arrows) or following the parent/children relations.
**Figure 17.** Request for explanation on performance and control variables links the user to an Electronic Handbook Information System (EHIS).
**Figure 18.** Performance variables may be ignored at user request.