9
ROUTINENESS REVISITED

David C. Brown

Artificial Intelligence Research Group
Computer Science Department
Worcester Polytechnic Institute
Worcester, MA 01609, USA.

Abstract

In the current research literature on the use of Artificial Intelligence (AI) in Design, we find many terms for types of design. In particular, the term "routine design" is often used, with a variety of definitions. The goal of this paper is to discuss Routine Design, and to contrast it with some of the other types of design. We will attempt to clarify the definition of routineness, and point out what is missing from existing definitions. We will also consider definitions of, and comments about Routine Design from other authors, as a contrast to our definition. In conclusion, we relate the notion of Class 1, 2 and 3 types of design, introduced by Brown & Chandrasekaran (1985), to ideas presented in the paper.

9.1 INTRODUCTION

In books and papers about design problem-solving we find many terms for types of design (for example, see [AAAI 1990] and [Finger & Dixon 1989]). These include: *Preliminary, Conceptual, Functional, Innovative, Creative, Routine, Embodiment, Parametric, Detailed, Redesign, Non-routine* and *Configuration*.

The goal of this paper is to discuss Routine Design, and to contrast it with some of the activities suggested by the other terms given above.

As Gero [1990, p. 34] says, "There seems to be a general acceptance of the classification of design into routine, innovative, and creative (Brown & Chandrasekaran 1985) ..." Unfortunately, many people have used the term "routine" in slightly different ways, many without understanding the key points of the original description. In this paper we will try to point to the sources of confusion, and will try to clarify the definition of the term.

9.1.1 Three Classes of Design

Let us start by considering the following passages from [Brown & Chandrasekaran 1985]:

Class 1 Design

The average designer in industry will rarely if ever do class 1 design, as we consider this to lead to major inventions or completely new products. It will often lead to the formation of a new company, division, or major marketing effort. This is extremely innovative behavior, and we suspect that very little design activity is in this class. For this class, neither the knowledge sources nor the problem-solving strategies are known in advance.

Class 2 Design

This is closer to routine, but will involve substantial innovation. This will require different types of problem-solvers in cooperation and will certainly include some planning. Class 2 design may arise during routine design when a new requirement is introduced that takes the design away from routine, requiring the use of new components and techniques. What makes this class 2 and not class 1 is that the knowledge sources can be identified in advance, but the problem-solving strategies, however, cannot.

Class 3 Design

Here a design proceeds by selecting among previously known sets of well-understood design alternatives. At each point in the design the choices may be simple, but overall the task is still too complex for it to be done merely by looking it up in a database of designs, as there are just too many possible combinations of initial requirements. The choices at each point may be simple, but that does not imply that the design process itself is simple, or that the components so designed must be simple. We feel that a significant portion of design activity falls into this class.

Class 3 Complexity

While class 3 design can be complex overall, at each stage the design alternatives are not as open-ended as they might be for class 2 or 1, thus requiring no planning during the design. In addition, all of the design goals and requirements are fully specified, subcomponents and functions already known, and knowledge sources already identified. For other classes of design this need not be the case.

9.1.2 The Key Points

Let us now discuss the key points of that definition, and add some of the refinements which appeared in that paper and in subsequent papers.

The main point (due mainly to Chandrasekaran) which is often overlooked, is summarized in the following table:

	Knowledge Sources	Problem-Solving Strategies
Class 1	Not Known	Not Known
Class 2	Known	Not Known
Class 3	Known	Known

For class 3 design, this means that everything about the design process, including the knowledge needed (i.e., knowledge sources), must be known in advance. Note that this does not mean that the specific design (i.e., the solution) is known in advance. Nor does it mean that the pattern of use of the knowledge (i.e., the design trace) is completely known in advance.

9.1.3 "Known" Knowledge

There is some ambiguity in the use of the word "Known" in the above table. We will discuss this in terms of Knowledge Sources, with obvious extension to Problem-Solving Strategies.

By referring to a Knowledge Source as "Known", we mean:

- that it is known in advance that the Knowledge Source will be needed to make that decision or set of decisions, and
- that the Knowledge Source is "immediately available" for use — that is, it does not have to be reasoned out or transformed from some other knowledge.

9.1.4 The Implications for Class 3 Design

The implications for class 3 design are:

- Use of a fixed set of well-understood design plans.
- No planning is required, only plan selection.
- Plan selection is fairly simple, with known criteria.
- Plans are probably not very long, or they would not be easily remembered.
- Possible problem decompositions are known in advance, while the actual

decomposition to be used is not.

- ❏ Dependencies between subproblems are known and, for the most part, can be compensated for in advance.
- ❏ Subproblems can usually be solved in a fixed order with little or no backtracking, due to the anticipated dependencies.
- ❏ All possible subcomponents of the object being designed are known in advance.
- ❏ The particular configuration of subcomponents chosen for a design in response to a given set of requirements is not known before the design activity starts. However, that configuration of subcomponents is a previously known configuration (i.e., the designer could identify it as a candidate solution for that type of design problem).
- ❏ All attributes or parameters (e.g., Length) of the design of a subcomponent are known (i.e., their names, *not* their values).
- ❏ The knowledge needed to calculate or select a value for each attribute is known in advance.
- ❏ Appropriate ranges of values are known for most attributes.
- ❏ There exist "expectations" about a typical value for an attribute in a particular design situation.
- ❏ The types of requirements given for a design problem are all known in advance.
- ❏ Many common failures during the design process will be recognizable.
- ❏ There exist suggestions about how to make changes to parameter values in order to fix failures.

9.1.5 AIR-CYL

The AIR-CYL system [Brown & Chandrasekaran 1989], that designed air cylinders, is an example of a Class 3 design system. The possible configurations are known in advance and are selected at run-time as a side-effect of plan selection, the possible plans for each subproblems are all available, the parameters to be given values are all known in advance, as is the knowledge used to produce those values.

The system, written in DSPL, a language for constructing design expert systems, also satisfies all of the other criteria in the list above. AIR-CYL is a system that does routine design. DSPL has been used to build systems for a variety of domains, such as Operational Amplifiers, Gear Pairs, Distillation Columns and Commercial Buildings.

In the following sections we will attempt to clarify the definition of routine-

ness, and point out what is missing from the presentation above. Then we will consider definitions of, and comments about Routine Design from other authors, as a contrast to the definition presented here.

9.2 A SECOND AXIS

The author's recent work [Brown 1991] addresses a form of learning known as *Compilation*, in which knowledge becomes transformed and reorganized in order to produce more efficient problem-solving.

The thesis that underlies the work in knowledge compilation during design is that design tasks become routine due to learning. This learning is brought about by repetition of similar problem-solving. That is, routineness is a direct reflection of experience. Routine designs are done more efficiently.

In order to avoid unwanted connotations, we will use the term *Non-Routine* as the opposite of *Routine*. The level of experience with a certain type of design will be reflected by a position on a Routine→Non-Routine axis — with "very experienced" at one end, and "inexperienced" at the other.

As this axis has nothing to do with *what* is being decided, this suggests the need for another axis that describes what sort of decisions are being made at various points during a design. We will use a Conceptual→Parametric axis for that. The intuition is that the axis shows the abstractness of the decisions being made, and reflects the notion that more constraints are added to the solution as the design activity progresses.

By *Conceptual* design we mean that the kind of things being decided at that point in the design are abstract (conceptual). For example, that the design requirements can be satisfied by a design which provides a particular function, or which has a particular pattern of sub-functions.

This is quite compatible with Dixon's very useful taxonomy of design problems [Dixon et al 1988]. His levels are named: *Functional*, *Phenomenological*, *Embodiment*, *Attribute* and *Parametric*. Clearly, these levels correspond to portions of the Conceptual→Parametric axis, even though this axis is less specific about the content of the decisions being made.

Dixon goes further, and states that "*conceptual* design is often used to describe the Embodiment of a design from Function" [p. 43]. He considers *preliminary* design to be an extension of conceptual design to another of his levels of specificity, i.e., to Artifact Type.

By *Parametric* design we mean that the things being decided are values for a prespecified set of attributes, and that providing values for these attributes fully specifies the design. In Dixon's terms, the design goes from Artifact Type level to the Artifact Instance level.

For many design problems, the Conceptual→Parametric axis represents the

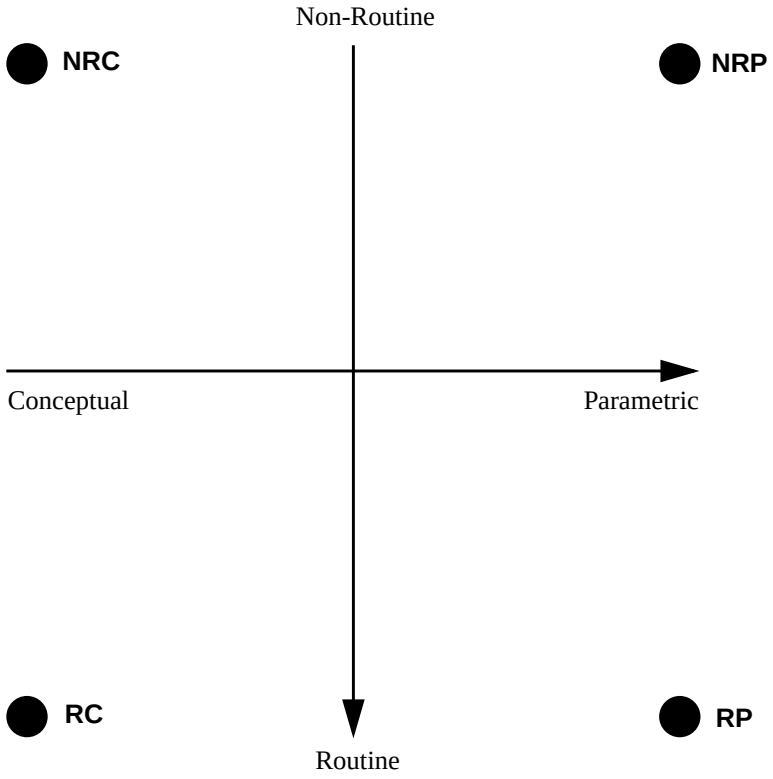

Figure 1: Orthogonal Axes

flow of time during the design activity, with earlier decisions falling towards the left and later decisions falling towards the right.

However, not all design problems have to begin with vague functional requirements and conclude with a fully specified design. For example, Dixon et al [1988] point out that a design activity can start at any level of abstraction and finish at any one of the more specific levels.

9.2.1 Four Categories of Design Activity

We consider the Routine→Non-Routine and Conceptual→Parametric axes to be orthogonal {see Figure 1}.

The space produced is naturally divided into four categories of design activity. They are represented by the four extreme points at the limits of the axes:

RC — Routine, Conceptual design,

RP — Routine, Parametric design,

NRC — Non-Routine, Conceptual design, and

NRP — Non-Routine, Parametric.

These will each be discussed below, in sections 9.3 and 9.4.

9.2.2 Concerns about the Analysis

At this point it is appropriate to discuss several concerns about this two axis analysis.

Relative Measure: As already stated, the routineness of a particular design problem depends on the experience of the problem-solver. Therefore, routineness is a relative measure. What is routine for one designer is not routine for another. What is routine for a designer today, may not have been two years ago. Routineness is in the brain of the beholder. It is an *individual's standard*.

In addition, there is also a *community standard*. The professional engineering design community may consider a design problem routine — meaning that there is an expectation that the problem will be routine for each member of the community. This may be because the specific knowledge and problem-solving for that problem is taught in college.

This community standard is probably easier to see at the Non-Routine end of the axis. Suppose we associate Non-Routine design activity with "innovation". It is easy to see that the community standard for a particular design problem is represented by the existing design solutions. Thus, a design can be innovative relative to that pool of existing designs.

Of course, it is perfectly possible for a design to be innovative relative to the individual's standard, but not innovative relative to the community standard.

Design Problems by themselves are not innovative, only in context. This demonstrates some of the danger in using the term Innovative Design.

The Routineness Axis: First, as routineness is expressed on an axis, with the possibility of different degrees of routineness, one should not assume that there are only four categories of design activity (i.e., it is closer to being continuous than discrete). Routine and Non-Routine are the extremes. We will try to restrict our focus to the extremes of both axes, in order to simplify the analysis.

What was learned: The Routine→Non-Routine axis is supposed to reflect the level of experience with a particular type of design. The more routine a problem is, the more knowledge is already "known" and is ready for immediate use. In the table at the start of the paper we separated what could be known in advance of carrying out a design into Knowledge Sources and Problem-Solving Strategies. The Routine→Non-Routine axis is concerned with how much is known, but does not distinguish between these two types of knowledge. A more refined analysis probably should make this distinction.

Subproblem Type: Up to this point we have assumed that all subproblems of a design problem are of the same class. This is not always realistic. In complicated problems some subproblems will be quite new, and will be non-routine, whereas other subproblems will lead to very well known components needing routine design, or even merely selection from a catalog. Clearly, this makes any model of design more complex.

Nonlinear progress: The reader should not assume that the nice, linear progress through a design problem which is 'suggested' by the Conceptual→Parametric axis is correct. Different subproblems can be at different points on the axis at any point in time. Problem-solving can jump from one point on the axis to another — for example, when a decision about using a certain type of component suggests a simplification of the functional design (perhaps through function sharing). Also, failures during design, due perhaps to incompatible choices, can lead to redesign (making changes to something already designed) or to re-design (doing whole portions of the design again from scratch). Nonlinear progress should not affect the arguments presented in this paper.

Other Axes: This two axis analysis ignores other dimensions. Several people, such as [Chandrasekaran 1990], [Brown 1992] and [Hayes-Roth 1990], have discussed the need for multiple mechanisms, or methods, for design tasks. For example, the use of constraint satisfaction or case-based reasoning to produce a design candidate. Our analysis does not reflect that dimension, and does not require it. The analysis also ignores the effect of the Domain (e.g., mechanical versus electrical) on the design activity (for example, see [Brown 1990] or [Waldron 1990]).

In the next two sections we will examine the four extreme categories of design activity (RC, RP, NRC and NRP), giving examples of each.

9.3 ROUTINE DESIGN

In this section we will examine two of the four extreme points defined by the two axes. They are at the Routine end of the Routine→Non-Routine axis.

9.3.1 Routine, Parametric Design

At the RP point the designer is deciding values for parameters (parametric), and has well-formed methods for deciding them (routine).

This is a typical routine situation, where a designer uses well-known methods to decide values for parameters. Several existing knowledge-based systems are capable of doing this category of design activity, such as AIR-CYL [Brown & Chandrasekaran 1989], and PRIDE [Mittal et al 1986].

9.3.2 Routine, Conceptual Design

At the RC point the designer is making very abstract decisions (conceptual), and has well-formed methods for deciding them (routine).

This category of design is done by a designer who often designs complex things given a rich but fixed set of requirements. For example, the designer of low-cost office buildings needs to decide which of a standard set of designs to use, and what type of structural system to use, given the type of equipment and numbers of people to be placed in the building. She also needs to consider the geological information about the site, as well as other factors such as the weather. The decisions made are not final values of parameters, but rather attributes of the design which will allow a list of parameters to be formed, so that parametric design can be done.

The best known knowledge-based system which is close to this kind of design is HI-RISE [Maher & Fenves 1985]. HI-RISE acts as an assistant to a designer for the preliminary structural design of high rise buildings. It generates “feasible alternatives for two functional systems”, in the form of structural systems. As these functional systems are known in advance, and the methods for selecting and checking the compatibility of the structural systems are also known in advance, then the system is doing routine design. As many of its decisions are fairly abstract, such as “braced frame” versus “shear wall” construction, the system belongs towards the conceptual end of the Conceptual→Parametric axis.

A Correction: In section 9.1.4 we presented a list of implications of the earlier definition of Class 3 design. Unfortunately, a few of the points refer to “subcomponents”. The RC category of design activity need not decide subcomponents. Consequently, those points should be changed to include more abstract decisions, such as “subfunctions”.

9.4 NON-ROUTINE DESIGN

In this section we will examine the other two of the four extreme points defined by the two axes. They are at the Non-Routine end of the Routine→Non-Routine axis.

9.4.1 Non-Routine, Conceptual Design

At the NRC point the designer is making very abstract decisions (conceptual), and does not have any well-formed approach to making them (non-routine).

This is the aspect of design about which we know the least. It is easy to think of the most abstract decisions being non-routine. A typical early design task might be deciding the full functionality of the object to be designed given the requirements. Those aspects of design that we normally consider to be most creative are precisely those in the NRC category. This is the sort of activity we associate with the initial stages of architectural design, for example.

There have been some attempts to produce design systems in this category (see [Gero & Maher 1989] and [Joskowicz et al 1992]). For example, Ulrich & Seering [1989] describe a system which generates graphs of functional elements (i.e., schematic descriptions) which produce a required relationship between a given input and a given output. They call this process *Schematic Synthesis*. The descriptions consist of idealized elements, such as pumps, or springs, which contain no information about geometry or materials. Descriptions can then be used to generate a physical description.

9.4.2 Non-Routine, Parametric Design

At the NRP point the designer is deciding values for parameters (parametric), and does not have any well-formed approach to making them (non-routine).

One can easily imagine a new designer in Industry being given the final step of a design project, where the rest had been completed by a senior designer. It is clearly possible for the naive designer to know all of the parameters to be decided, but not know how to go about deciding them. This would result in non-routine behavior, such as analyzing the dependencies between the parameters in order to determine an appropriate order in which to decide them, or searching textbooks for appropriate methods or equations.

Another more complex example can be found in the task of designing hulls for racing yachts, as described by Gelsey in [Joskowicz et al 1992, p. 44]. The hull's shape can be described by a grid of planar panels. The problem can be viewed as that of finding sizes for those panels, i.e., finding values for parameters. In actual fact, the grid may need to be changed, in order to improve expected performance. That sort of change will produce a *new* set of parameters. Producing this new set is *not* a parametric design task.

The statement in section 9.2 that “routineness is a direct reflection of experience” is not meant to imply that all problems can produce the same degree of routineness with experience. For example, some problems have a dependency structure which is much too complex to be properly analyzed by the designer. Even if the dependencies were known *a priori*, the ordering of the tasks may not be, as in the tasks Balkany et al [1991] label as Type 2.

Such complexity might lead to use of the Iterative Refinement approach to parametric design [Orelup et al 1988] [Ramachandran et al 1988]. One can argue that this is not Routine design, as it is not possible to anticipate the order of decisions, therefore preformed plans cannot exist and cannot be used. In addition, an Iterative Refinement approach will decide the value of some parameters more than once, nudging them gradually to their final acceptable position.

9.5 RELATED WORK

In this section we will consider some recent definitions of, and comments about, Routine Design from other authors. This is by no means an exhaustive review. It does not concentrate on definitions which we consider to be totally wrong. It is merely intended to show the range of variation in the literature. Many authors use the term “routine” with no associated definition.

Gero [1990] proposes a model of design based on the retrieval and instantiation of “Design Prototypes” which bring “all the requisite knowledge appropriate to the design situation together in one schema”. This knowledge includes Function, Behavior, Structure, Relational, Qualitative, Computational, Constraints, and Context.

“Routine design can be defined as the design that proceeds within a well-defined state space of potentials designs. That is, all the variables and their applicable ranges, as well as the knowledge to compute their values, are all directly instantiable from existing design prototypes” [p. 34].

On the surface this appears to be quite compatible with our definition. However, he also states that “instances are refined in two ways. The first way is by pruning the set of variables to the applicable set through a specification of applicable functions, structures, or behaviors and propagating this specification. The second way is by determining the values of the applicable set of variables using the available knowledge”.

This “second way” is clearly routine, but the “first way” implies that the set of variables to be given values needs to be determined via propagation. This would mean that neither the variables nor the methods for giving them values are “Known”, in the sense already defined. Consequently, there is some conflict here with our view.

Tomiyama [1990] lists classes of design as “Creative, New, Combinatory, Routine, Parametric, and Redesign”. Although he avoids the confusion between

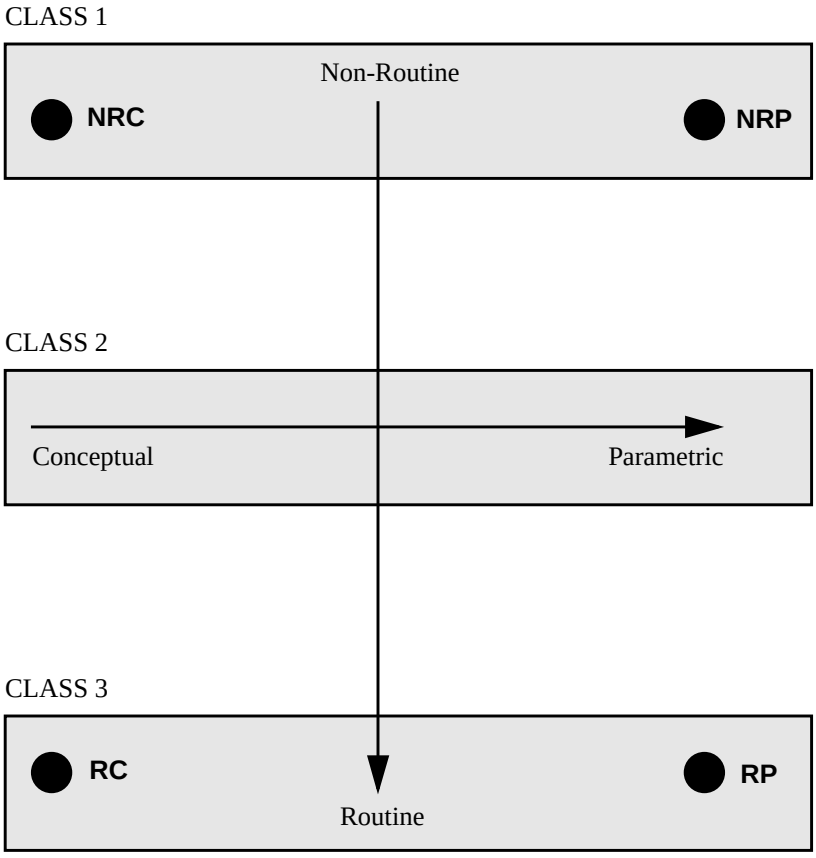

Figure 2: Three Classes of Design Activity

Routine and Parametric, the relationship between Routine and the other types is unclear (especially for "New").

Tomiyama also presents "Attribute Modeling", where: design objects only have attributes/parameters, design objects do not change their structure, and "Well Formalized" design processes act on attributes. This, he claims is used to deal with "Routine-Type Design". However, "constraint solving" is allowed as a design process. If this were to be done by the usual constraint satisfaction methods this would not be routine given our definition, as plans are not available and some search is required.

Agogino, in her presentation at a recent AAAI workshop [Joskowicz et al 1992], argued that routine design introduced "no new variables", while in non-routine design "new variables are created". This definition is in terms of the Conceptual→Parametric axis, and not strictly in terms of routineness.

Snavely et al [1990] present four "mutually exclusive types of design", called: Invention, Innovation, Routine and Procedural. For them, routine design "is the process of filling the slots of a fixed topology (or predetermined set of fixed topologies) with catalog entries". A "catalog" is a database with multiple levels of abstraction, with the lowest being typical Part Catalog entries (e.g., a particular spring). As they allow a catalog entry to be a "dimension" this appears to overlap Parametric design. Although one could argue that the topology is "fixed" because it is "known" to be the solution, that does not appear to be their claim. Their main criterion for distinguishing between their types is the variability of the topology — the higher the variability, the more inventive.

Sriram & Tong [1990] provide a formal definition of design as {S, C, A, K, Δ}, where S is the set of solutions, C is the set of constraints that need to be satisfied, A is the set of artifacts, K is the knowledge used to develop S, and Δ is the set of transformation operators. They list "design activities" as: Creative, Innovative and Routine. They distinguish between them by which of the ingredients of the formal definition are known. Thus their definition is one of the few that have activities arranged solely along the Routine→Non-Routine axis.

9.6 SUMMARY AND CONFUSION

Where do the three Classes presented in section 9.1.1 fit into this new two axis analysis? As the classes are concerned with how much is already *known*, as opposed to what is decided, it is clear that they are positioned along the Routine→Non-Routine axis {see Figure 2}.

The figure is *not* intended to be taken too literally. The rectangles representing the classes are put in representative places. Class 3 covers all the routine cases. Class 1 covers all the non-routine cases. Class 2 is between them. Where exactly are the boundaries? This isn't clear. Does class 2 cover the rest of the space? Yes, if these are really supposed to cover *all* the possibilities.

One could even propose a new class, perhaps Class 2a, with Knowledge Sources "Not Known" and Problem-Solving Strategies "Known" — where the approach to solving the problem was known in advance but what knowledge to be used wasn't. It isn't clear that this situation would often occur. As exactly *what* (i.e., Knowledge vs. Strategy) is known is yet another dimension, class 2 and 2a would occupy approximately the same position in Figure 2, as in both cases only one of the two types of knowledge is known.

9.6.1 Confusion

At the start of this paper it was pointed out that there has been some confusion about routine design. The most common confusion is that routine design equals parametric design. One major reason for this is that it is very likely that parametric design problems are routine. This is not merely because they usually represent the most "automatic" aspects of design (i.e., the use of equations to produce values). Another reason is due to the following argument.

As stated earlier: "For many design problems, the Conceptual→Parametric axis represents the flow of time during the design activity, with earlier decisions falling towards the left and later decisions falling towards the right." Because of the impact of experience, repetition of the design problem with similar but different requirements will cause the amount of reasoning needed to be reduced.

Thus, conceptual design effort will gradually be reduced, and will become unnecessary, as it will be the same or similar for every design with similar requirements. Eventually, all that will be required is parametric design, and that will become routine.

Thus, in this situation, design problems will gradually take less time, and will require less reasoning. Consequently, it is natural to associate routineness with parametric design, just as it is natural to associate non-routineness with conceptual design. This is the source of the confusion.

9.6.2 Summary

In this paper we have examined routineness, have provided a cleaner definition, have introduced four extreme categories of design using two orthogonal axes, have related this analysis to the three Classes of design, and have explained a source of confusion.

Acknowledgments: This work was supported in part by NSF grant DDM-8719960. The author would like to acknowledge the support and contributions of past and present members of the WPI AI in Design Research Group, as well as the WPI AI Research Group.

References

AAAI (1990) *AI Magazine*, Special Issue on Design, Winter, Vol. 11, No. 4, American Association for Artificial Intelligence.

A. Balkany, W. P. Birmingham & I. D. Tommelein (1991) A Knowledge-level Analysis of Several Design Tools. *Artificial Intelligence in Design '91*, J. Gero (Ed.), Butterworth Heinemann, pp. 921-940.

D. C. Brown (1990) Research into Knowledge-Based Design at WPI. *Applications of Artificial Intelligence in Engineering, Vol. 1, Design*. (Ed.) J. S. Gero, Computational Mechanics Publications & Springer-Verlag.

D. C. Brown (1991) Compilation: The Hidden Dimension of Design Systems. *Intelligent CAD, III*, H. Yoshikawa & F. Arbab (Eds.), North-Holland.

D. C. Brown (1992) Design. *Encyclopedia of Artificial Intelligence, 2nd Edn.*, S. C. Shapiro (Ed.), J. Wiley, pp. 331-339.

D. C. Brown & B. Chandrasekaran (1985) Expert Systems for a Class of Mechanical Design Activity. *Knowledge Engineering in Computer-Aided Design*, J. S. Gero (Ed.), North Holland, pp. 259-282.

D. C. Brown & B. Chandrasekaran (1989) *Design Problem Solving: Knowledge Structures and Control Strategies*. Research Notes in Artificial Intelligence Series, Morgan Kaufmann Publishers, Inc.

J. R. Dixon, M. R. Duffey, R. K. Irani, K. L. Meunier & M. F. Orelup (1988) A Proposed Taxonomy of Mechanical Design Problems. *Proceedings of the ASME Computers in Engineering Conference*, San Francisco, CA, Vol. 1, p. 41.

S. Finger & J. R. Dixon (1989) A Review of Research in Mechanical Engineering Design, Part I: Descriptive, Prescriptive, and Computer-Based Models of Design. *Research in Eng. Design*, Vol. 1, No. 1, Springer International, p. 51.

J. S. Gero (1990) Design Prototypes: A Knowledge Representation Schema for Design. *AI Magazine*, Special Issue on Design, Winter, Vol. 11, No. 4, American Association for Artificial Intelligence, p. 26.

J. S. Gero & M. L. Maher (Eds.) (1989) *Proc. of the Workshop on Modeling Creativity and Knowledge-Based Creative Design*. University of Sydney.

B. Hayes-Roth (1990) Three Topics for Discussion. *Working Notes for the Workshop on Creating a Scientific Community at the Interface between Engineering Design and AI.* Engineering Design Research Center, Carnegie-Mellon University, Pittsburgh, PA, p. 12.

L. Joskowicz, B. Williams, J. Cagan & T. Dean (Eds.) (1992) Symposium: Design from Physical Principles, Working Notes, AAAI Fall Symposium, October, Cambridge, MA.

M. L. Maher & S. J. Fenves (1985) HI-RISE: A Knowledge-Based Expert System for the Preliminary Structural Design of High Rise Buildings. Report No. R-

85-146, Dept. of Civil Engineering, Carnegie-Mellon University, Pittsburgh, PA.

S. Mittal, C. L. Dym & M. Morjaria (1986) PRIDE: An Expert System for the Design of Paper Handling Systems. *IEEE Computer Magazine*, Special Issue on Expert Systems for Engineering Problems, Se June Hong (Ed.).

M. F. Orelup, J. R. Dixon, P. R. Cohen & M. K. Simmons (1988) Dominic II: Meta-Level Control in Iterative Redesign. *Proc. 7th National Conf. on Artificial Intelligence*, AAAI, St. Paul, MN.

N. Ramachandran, A. Shah & N. A. Langrana (1988) Expert System Approach in Design of Mechanical Components. *Proc. ASME Int. Computers in Engineering Conf.*, San Francisco, CA.

G. L. Snavely, L. P. Pomrehn & P. Y. Papalambros (1990) Toward a Vocabulary for Classifying Research in Mechanical Design Automation. *First International Workshop on Formal Methods in Engineering Design, Manufacturing and Assembly.*

D. Sriram & C. Tong (1990) AI and Engineering Design. Notes for Tutorial WP2, *AAAI-90: the 8th Nat. Conf. on Artificial Intelligence*, Boston, MA.

T. Tomiyama (1990) Intelligent CAD Systems. Notes for Tutorial, *Eurographics '90*, Montreux, France.

K. T. Ulrich & W. P. Seering (1989) Synthesis of Schematic Descriptions in Mechanical Design. *Research in Engineering Design*, Academic Press, Vol. 1, No. 1, p. 3.

M. B. Waldron (1990) Understanding Design. *Intelligent CAD, II,* H. Yoshikawa & T. Holden (Eds.), North-Holland, pp. 73-87.