The computational nature of routine design problems

Homero M. Schneider

Centro de Pesquisas Renato Archer – CenPRA, Campinas SP
E-mail: *homero.schneider@cenpra.gov.br*

Abstract. In this paper, we make an investigation into the structure of routine design problems. By analyzing the design problem solving process, we show that the designer acquires a special kind of design knowledge, which we called path variability knowledge. Based on this kind of knowledge, we conclude that routine design can be regarded as a class of well-structured problems. Therefore, contrary to the prevailing view that regards routine design problems as search problems, we argue that within a chosen domain of variation they may be solved by direct algorithmic methods.

1. Introduction

It has been a common practice to divide design problems into classes with different degrees of difficulty (Kota, 1994). For example, Gero (1990) identifies three classes of design problems: routine, innovative and creative, and Brown (1991) classifies design problems in reference to two orthogonal axes (routine/non-routine and conceptual/parametric), defining four major class of design problem. For the purposes of this paper, we will recognize only two classes: routine and non-routine design. The former class of design problems can be solved in reference to past experiences using only domain and design knowledge already available to the designer. On the other hand, for non-routine design problems it is necessary to acquire new domain or design knowledge and they involve various cognitive mechanisms (Kolodner and Wills, 1993; Goel, 1997).

Since the seminal work of Newel and Simon (1972), search has been an influential computational paradigm to approach design problems (as in the whole AI field). However, this view has been rejected for creative design problems on the ground that this kind of problem is too ill-defined and that it should be treated more like a design space exploration (Logan and Smithers, 1992). Nevertheless, search is still accepted as an adequate paradigm for routine design.

In this paper, it is argued that even for routine design problems search is an inadequate paradigm. The argument is that for this type of design problem there is enough design knowledge available to solve them by direct methods. This knowledge, which will called path variability knowledge, is developed as the designer makes explorations during the design process and solves a number of similar design problems. Eventually, path variability knowledge becomes the dominant knowledge in the design process, such that, similar problems can be solved using only this kind of knowledge. Moreover, when captured and formalized, path variability knowledge turns out into a procedural form of knowledge and, therefore, search is not necessary.

In the following three sections, we will define the basic framework over which routine problems will be investigated. Then, in sections 5 and 6, we consider in detailed the structural nature of the knowledge acquired during the design problem solving. In section 6 we conclude that a routine design problem is a well-structured problem and, therefore, amenable to a direct problem solving method.

2. Design as a problem solving process

As shown in Figure 1, the design activity can be conceived as a problem solving process whereby a designer transforms product requirements into detailed product descriptions. A solution to the problem is a detailed product description incorporating all the requirements. During this process, knowledge (from a knowledge base) is brought into the problem solving and new knowledge is generated through cognitive process. The new knowledge is stored in the knowledge base for future use.

In general, design is a very complex problem solving activity that lasts months and even years. Therefore, many models for the systematization of the design activity have been proposed (Pahl and Beitz, 1988). Following the model presented in (Ulrich and Eppinger, 1995), we will divide the design process into three phases: concept development, system-level design and detail design, each one associated to a different design representations of the product. However, we left out the phases of testing and refinement, and production ramp-up. These phases may give rise to design cycles for the improvement of the solution, but do not introduce new product representations.

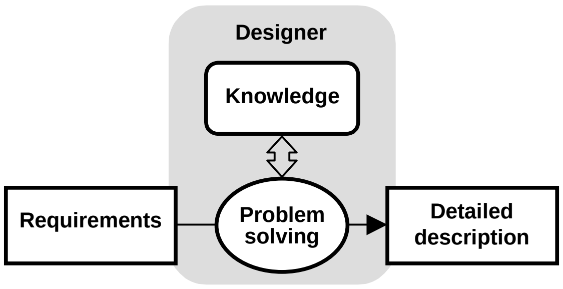

Figure 1. Main elements of the design problem solving process.

Each design phase is characterized by specific activities, methods and outcomes. In the concept design phase, the initial customer requirements are refined and the conceptual model of the product is generated, with the definition of the functional structure and the associated working principles. At the system-level design phase, the components are defined and arranged into the product architecture. Finally, at the detail design phase, the complete and detail product description showing the assemblies, part geometry, dimensioning, tolerances, materials, etc. are defined. Accordingly, we can identify four main product representation forms into which the product conception is expressed along the design process: 1) specification, expressing the product by means of a set of requirements, 2) concepts, by means of functional diagrams, 3) architecture, by means of component schemata and 4) detail design, expressing the product by means of drawings. Each of these representations can be associated to a space of all possible product expressions in that representation. Then, design problem solving can be conceived as a transformation of product information along these spaces, from requirements toward detailed description.

3. The design transformation path

Although we have ascribed a direction of transformation, design is a highly iterative process. As there is a better understanding of the design problem, the set of requirements is refined, new requirements are added and inconsistencies are removed. Furthermore, alternative conceptual models and architectures will be generated during the design process through synthesis-analysis cycles. Changes to the final detailed description are made as well, particularly during the production system development.

This state of affairs is captured in Figure 2, which shows the design process as a series of transformations (represented by arrows) between the design spaces. A sequence of transformations from a specification onto a detailed description represents a transformation path. It should be stressed that transformation paths comprises not only the outcomes associated to the design spaces, but also the methods and the decisions taken, which establish the rational linking between the outcomes during the design process. Note that the outcomes at the corresponding spaces are specific results defined by the transformations. During the design process, the spaces may be explored by the generation of alternative results, but eventually a transformation path is consolidated. This is represented in Figure 2 by the heavy arrows. Not only the consolidated transformation path, but all the outcomes from the spaces exploration becomes part of the designer's knowledge base.

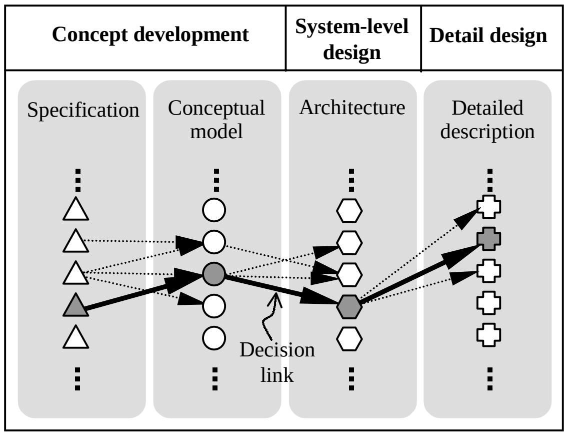

Figure 2. The design transformation paths.

Although the design transformation path is developed in connection to a particular design problem, its outcomes are available to be applied on other design problems through "prototypes" (Gero, 1990), case based reasoning (Kolodner and Wills, 1993), analogical reasoning (Goel, 1997) etc. However, for specifications that are similar (in a sense that will be made precise in the next sections) to previously solved design problems, there is also more direct ways in which the knowledge base can be used. In order to dwell on these points, first we have to define in more detail the structure of the specification spaces and its relation to the design space.

4. The space structure

A product specification is a set of requirements that has to be embodied into the design solution. For example, “operating range: -20° C to 50° C”, “material: steel or plastic”, “connection through the top”, etc. Now, let Ω be the set of all possible statements that can be made about artifacts, real or conceivable. Then, every artifact specification is a subset $R \subset \Omega$. Furthermore, let $\Psi = \wp(\Omega)$ be the set of all the subset of Ω. Then every possible artifact specification is an element of Ψ. Thus, we call Ψ the specification space. Given two specifications R, $R' \in \Psi$, if $R' \supseteq R$ then we say that R' contains additional requirements to R. The combination and the intersections of specifications is also an element of Ψ. Thus, space Ψ can be structured into a lattice of specifications by the relation of inclusion and Ω is its upper limit.

On the other hand, the design space contains the detailed description of specific artifacts in a representation that closely resemble what the real artifact will be or from which it can be unambiguously produced. In what follows the design space will be represented by Φ.

Design is a problem solving process that starts with an initial specification $R \in \Psi$ which evolves during the problem solving process, as there is a better understanding of problem structure, to a final set $R' \in \Psi$. The design process ends when a solution $D \in \Phi$ is found that embodies all the requirements in R'.

If the specification R is associated to a design D by a transformation path, then a non empty subset $R' \subset R$ is also associated to D. In this case, R is a more complete set of requirements for solution D than R'. This condition lead us to the following question: Is there a set $R^2 \in \Psi$ associated to solution D which is more complete than R? And another $R^3 \in \Psi$ which is more complete then R^2? And so on. Given the lattice structure of Ψ, this process must have a limit or Ω will be associated to solution D, which is impossible since this would imply that every possible requirement, even contradictory ones, are all incorporate in D. Thus, we can introduce the following definition. A set $R^* \in \Psi$ is complete for solution D if, and only if, for every $R \in \Psi$ such that $R \supset R^*$, then the set of requirements $R - R^*$ are not embodied in D. The complete set of requirements for D will be represented by R_D.

Noting that, whenever two specifications are associated to the same solution their union is also a specification to the same solution, the following proposition (for the construction of complete specifications) can be easily proved. Let Ψ_D be the set of all specifications in Ψ associated to the design solution D. The set $R^* = \cup R$ (for all $R \in \Psi_D$) is such that $R^* = R_D$.

From the definition R_D it follows that, if $R \in \Psi$ is such that $R \not\subset R_D$, then D is not a solution for R. In particular, if $R_{D'}$ is the complete set of requirement for solution D' and $D' \neq D$, then $R_{D'} \not\subset R_D$ and $R_D \not\subset R_{D'}$. Thus, we conclude that there is a one-to-one relation between complete set of requirements in Ψ and design solutions in Φ.

If an initial specification $R \in \Psi$ is incomplete, it may be associated to any one design solution $D \in \Phi$ such that $R \subseteq R_D$. The particular solution that will be associated to R will depend on the requirements that are added to it during the design process. In practice, a complete specification is never obtained by the designers, but the options of solutions that can be associated are progressively narrowed down as new requirements are added. Of course, if R contains contradictory requirements (or if one such is added) then R is inconsistent and there is not solution associate to it.

5. The nature of the transformation paths

Making different decisions during the design process results in different transformation paths, each one ending up at a different solution in the design spaces. For example, when confronted with the choice of the material to be used in the car’s bumpers, the designer may be faced with two options: metal or plastic (reinforce with carbon fibers) bumpers. Each one of these alternatives requires a different conception of bumpers that leads to a different design solution. Moreover, decisions may affect different artifact perspectives (function, behavior, structure or attribute) and have different scopes of impact on the design solution, from a small component to several of the artifact modules.

However, decisions along the design process are not only related to the solution, but to the specification as well. If the designer is face with a decision during the design process that is related to a set of requirements in the artifact specification, he must follow the requirements and make a consistent decision. If the specification is incomplete and no requirement is related to the decision, its is up to the designer to decide between the possible alternatives. However, for every decision that the designer makes and for which there is no related requirement in the product specification, a new set of requirements corresponding to the decision made has to be added into the product specification. Furthermore, if for some reason the designer makes a decision that conflicts with the existing requirements, some of them will be changed to become consistent to the decision made. In this way, the initial specification is completed and modified as the design process progresses. However, in practice, not all the designer's decision making are made explicit and added to the product specification, though it could in principle.

From the considerations made above, we can establish a unique correspondence between all the decisions made during the design process with the set of final requirements in the specification space and, on the other hand, with the solution arrived at in the design space. Hence, every alternative decision leads to a different solution $D' \in \Phi$ which is associated to a different set of complete requirement $R_{D'} \in \Psi$. Once again we arrive at the conclusion that there is a one-to-one correspondence between complete set of requirements in Ψ and design solutions in Φ.

Now, let us suppose that at a decision point p_k during the design process the designer is confronted with set of decision alternatives $\{a_1, a_2,..., a_n\}$, from which a_i is chosen. Then, along the way there is another decision point p_l, associated to the set of alternatives $\{b_1, b_2,..., b_m\}$, from which b_j is chosen, and so on, until a solution D is developed. In this case, there is a transformation path from the specification space to the design space that includes the decisions $p_k = a_i$ and $p_l = b_j$. However, what would happen to the transformation path if $p_k \neq a_i$? As we already considered above, we would have a different transformation path and D would be no more the associated solution. Moreover, it may happen that the decision point p_l is no more along the way in the new transformation path. In this sense, we can say that a design process leaves a trace of decisions from the specification towards the correspondent solution. Actually, this is a network of decisions, since decisions are also interconnected. Let us return to the bumpers example to illustrate this last point. We have seen that each material alternative have different implications to the bumpers' design. In particular, the plastic bumper will require a reinforcement beam and may use energy absorbing foam, each of these solutions eliciting additional decisions, for example, the density of the foam. Furthermore, let us suppose that the designer is considering two alternatives for the bumpers finishing: painting the bumpers with the same color as the car's body or to chrome plated them. However, he is told by the production that plastic cannot be satisfactorily chrome plated. Thus, the only option left for plastic bumpers is to paint the bumpers. Hence, each choice of material for the bumpers leads to a different transformation path involving different decisions along the way. Note also that in this example the decision on the bumpers finishing depends on the material chosen (or vice versa). Hence, decisions along the transformation path can be coupled by a chain of decisions, and many of them can be part of more than one transformation path, superposition different transformation paths.

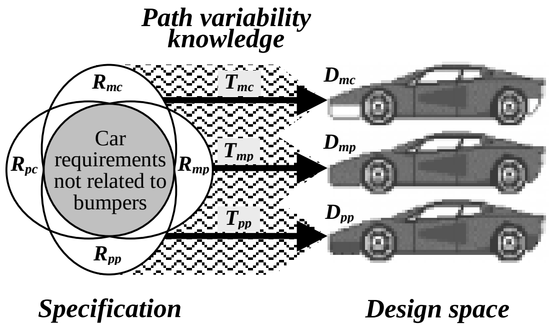

Figure 4. Transformation paths and the associated path variability knowledge.

Taking into account the decision alternatives related to the material and the finishing of the bumpers, there are three different transformation paths T_{mp}, T_{mc} and T_{pp}, as depicted in Figure 4. Each path defines a different solution in the design space: a car with painted metal bumpers (D_{mp}), a car with chromed metal bumpers (D_{mc}) and a car with painted plastic bumpers (D_{pp}). Accordingly, at the specification space there are three associated complete sets of requirements R_{mp}, R_{mc} and R_{pp}, one differing from the other at least with respect to the type of material and/or the type of finishing of the bumpers. However, note that the specification R_{pc}, which includes the requirements "plastic bumpers" and "chrome plate bumpers", is also a possible product specification but one for which there is no solution associated (as long as the technological problem with plastic chrome plating is not solved). Hence, this example also shows that not all combination of decision alternative may be a valid transformation path.

6. The variation model

In Figure 4, each of the transformation paths shown are composed of specific combinations of decisions with their outcomes at each of the design spaces. Collectively, they can be regarded as the transformation paths of distinct design problems (with similar specifications) or as alternative transformation paths generated during a single design problem solving process. In any case, the design explorations, the correlations observed, rules derived etc., are all part of the designer experience and are available to be used next time the designer is confronted with a design problem. This knowledge is about how the trasformation paths can be modified and combined, therefore, it will be called the path variability knowledge.

Now, let R_t = {R **:** $R \subseteq R_D$ where R_D is the complete set of requirements of some transformation path in the designer's knowledge base}. It its clear from the definition of R_D that, if $R \subseteq R_D$ then D is also a solution for R. In practice, all the designer has to do is to identify that R is consistent to some D which is part of his experience or, equivalently, that $R \subseteq R_D$ for some R_D. Note that, actually the designer does not have to know R_D for any design that is part of his experience, it suffices to recover any $R' \in R_t$ for which $R \subseteq R'$. Hence, R_t is the set of all specifications in Ψ to which the designer can, in principle, ascribe a solution based directly on the transformation paths that are part of his experience.

However, even if $R \notin R_t$ it may happen that the designer can still associate a design to it. Let us illustrate this point. After having explored the behavior of the bumpers system for various combinations of the design parameters, eventually the designer learns how these parameters are correlated and, as a consequence, he is capable of setting these parameters for a range of impact requirements (for example, between 4 and 8 km/h). Thus, if the designer is presented with a new bumper specification containing an impact requirement within that range, but one for which he has never developed a solution, he is still capable of providing a solution for it in a straightforward way. This is typically part of the path variability knowledge for the bumpers transformation paths.

Hence, because of this kind of knowledge, the set R_t can be extended to form a set of specifications R_e, such that, $R_e \supset R_t$. If presented with a new specification $R \notin R_t$, but within R_e, the designer will still be capable of defining a solution that meets the requirements of R with the help of variability path knowledge. Moreover, he can do it without having to go all along a transformation path. In what follows, we will work out this point in more details.

Let $P = \{p_1, p_2,..., p_n\}$ be a set of decision points associated to a set of transformation paths and their path variability knowledge, and A_i their correspondent set of alternative decisions. Alternatives in A_i may be an actual decision taken as part of some transformation path or be just an extrapolation from the insight gained during the design explorations, for example, a range of possible values for the foam density. Although, for each alternative in A_i (i = 1, 2,..., n) there is an associated design solution, this may not be true for any combination of alternative decisions of an arbitrary subset of P. As we have seen in the bumper example, this is due to a hierarchical structure and the dependencies that exist between the decision points. However, since the decision points are identified over the transformation paths and their variations, this is the knowledge source from which the organization of the decision points must be defined. In particular, the choices between alternative decisions can be explained based on that knowledge. Hence, based on the organizational structure of the decision points, instead of a blind decision making process, this can be done according to some inherent systematic. Moreover, at each $p_n \in P$ it is possible to elaborate procedures or methods for choosing the proper alternative in view of the requirements and the previous decisions made.

Based on the above argumentation we will make the following statement. Let $P = \{p_1, p_2,..., p_n\}$be a set of decision points identified over a set of transformation paths and their variations and let $A = \{A_1, A_2,..., A_n\}$be their associated set of alternative decisions. There is a procedural model $V(P, A)$ that is capable to transforming specifications into detailed design descriptions. This will be referred to as the variation model V associated to the decision points P and their correspondent alternative decisions A.

In order to delimit the set of requirements in the specification space over which $V(P, A)$ is defined, we begin noting that for every A_i (i = 1, 2,..., n) there is a correspondent set R_i in the specification space, such that, there is a one-to-one correspondence between alternative decisions in A_i and requirements in R_i. Therefore, the set $\Re = \{\{ r_1, r_2,..., r_n \}: r_i \in R_i$ $(i = 1, 2,..., n)\}$, defines the limits of variation within the space of specifications for the model $M(P, A)$. However, if $V(P, A)$ does not allow all combination of decisions, not all elements in $\Re$ are valid specification for this model. For every valid specification $V(P, A)$ will associate a solution to it.

In reference to Figure 4, if bumpers are all that will be allowed to be changed in the car's design and if material and finishing are all the decisions points that will be allowed to be varied (with their alternative decisions presented above) then there is an associated variation model V({material, finishing}, {{steel, plastic}, {chrome, paint}}) that can be used to design customized cars. This is a trivial case in which only the set specifications {R_{mp}, R_{mc}, R_{pp}} forms the range of specifications to which the model can be applied. By the inclusion of dimensional decision points in this model, the set of specification over which the model is defined could be extended beyond the specifications explicitly associated to the consolidated transformation paths.

Note that the specifications discriminated in Figure 4 have a core set of requirements that are common to all of them. This is the set of car requirements not related to the bumpers. Since they are fixed, these requirements need not to be made explicit but they will be taken into account in the solution given by the model. Hence, the specification for a variational model is composed of a set of fixed and implicit requirements and a set of variable requirements. However, because the implicit part will be taken for granted, it must not be referred to explicitly as part of the specifications.

6. Conclusion

While non-routine design problems are typified as space explorations, routine design problems are still conceived as a space search (Logan and Smithers, 1992; Gero, 1994).

For a problem solving process to qualify as a search, it is necessary that the problem space, the initial state, the solution criteria and the operators that transform states within the space be all defined. Indeed, a routine design problem can be cast as a search process, for example, by representing the decision points as variables and the set of alternative decisions as values for the correspondent variable. The solution to the problem is a value attribution to the variables (a decision making) such that none of the constraints that exist between them is violated. This is how configuration problems have been treated within the constraint satisfaction problem approach (for example, Sabin and Freuder, 1996).

However, as it has been argued in sections 4 and 5, within a given range of variability, as defined by the set of decision points and their correspondent sets of alternative decisions, there is enough design knowledge associated to a routine design problem in order to establish the dependency between the decision points and to elaborate procedures to automate the decision making at those points.

Therefore, we conclude that although the routine design problems can be treated within the search paradigm, they can be further structured up to the point that, within a delimited set of specifications, they become direct problem solving process.

Bibliography

Brown, D. (1996) Routineness revisited, *in Mechanical Design: Theory and Methodology*, M Waldron and K Waldron (eds.), Springer-Verlag, Berlin, pp. 195-208.

Gero, J. (1990) Design prototypes: a knowledge representation schema for design, *AI Magazine*, **11**(4), pp 26-36.

Gero, J. S. (1994). Towards a model of exploration in computer-aided design, *in Formal Design Methods for CAD*, J. S. Gero and E. Tyugu (eds), North-Holland, Amsterdam, pp. 315-336.

Goel, A. K. (1997) Design, analogy and creativity, IEEE Expert Intelligent Systems and Their Applications, **12**(3), pp. 62–70.

Kolodner, J. L. and Wills, L. M. (1993), Case-based creative design, AAAI Spring Symposium on AI, Stanford, CA.

Kota, S. (1994) Computational models for routine design, in *Intelligent Systems in Design and Manufacturing*, C. Dagli and A. Kusiak (eds.), ASME Press, New York.

Logan, B. and Smithers, T. (1992) Creativity and design as exploration, *in Modelling Creativity and Knowledge Based Creative Design*, J. S. Gero and M. L. Maher (eds.), Lawrence Earlbaum.

Newel, A. and Simon, H. (1972) *Human Problem Solving*. Prentice Hall, Enlewood Cliffs, New Jersey.

Pahl, G. and Beitz, W. (1988) *Engineering Design*, Springer-Verlag.

Pine, J. B. II. (1993) *Mass Customization: The new Frontier in Business Competition*. Boston: Harvard Business School Press.

Sabin, D. e Freuder, E. C. (1996). Configuration as composite constraint satisfaction, *Configuration —Papers from the 1996 Fall Symposium*, AAAI Technical Report FS-96-03.

Sabin, D. e Freuder, E. C., (1998) Product configuration frameworks – a survey, *IEEE Intelligent Systems and Applications*, **13**(4), 42–49.

Ulrich, K. T. and Eppinger, S. D. (1995) *Product Design and Development*, McGraw-Hill, New York.