From: AAAI Technical Report FS-92-03. Copyright © 1992, AAAI (www.aaai.org). All rights reserved.

Model Synthesis in Device Redesign

Sattiraju Prabhakar
School Of Computing Sciences, University of Technology, Sydney,
PO Box 123, Broadway, Sydney, NSW 2007, Australia

Ashok Goel
College of Computing, Georgia Institute of Technology,
Atlanta, Georgia, USA

Abstract

We present a model that represents the knowledge required for designing devices innovatively and a methodology, called Performance Driven Innovation (PDI), that synthesizes new models for redesign problem solving. PDI is an extension of KRITIK system that integrates Case-based and Model-based reasoning. The knowledge required for innovative design is represented using variations of a ***Structure-Function-Behavior*** (SBF) model. This model represents both the design and the failure knowledge of a device. This failure knowledge is the designer's understanding of how the device fails in a new environment. The design knowledge of the devices is often incomplete, as the complete knowledge of all the environments, in which the device can operate, is not available. This knowledge is arrived at as an instantiation of prototypical failures which are present in a case-base. This knowledge is composed with the design knowledge. This failure knowledge is used to discover the new constraints on the device imposed by the new environment that lead to its behavioral failure. These constraints are realized by generating behaviors. These behaviors are composed with the behavior in the design knowledge which gives rise to the behavior that satisfies all the constraints.

1 Background

Over the last several years we have been developing a **model-based approach** to design in the domain of physical devices such as heat exchangers which fits within a general framework of case-based device design. In our framework, the case-based method sets up several subtasks of the design task: case retrieval, design adaptation, design verification, redesign, and case storage, etc [Goel and Chandrasekaran 1990]. Model-based methods are used to perform these subtasks. In this paper, we focus on model-based redesign [Goel and Chandrasekaran 1989] in the context of case-based device design.

The redesign task is decomposed into the subtasks of diagnosis and repair. If a proposed design fails to accomplish the intended functions, it is first diagnosed and then repaired. The designer's comprehension of the design is represented as a design-specific ***structure-behavior-function*** (SBF) model. This model explicitly represents the design structure, the intended functions of the device, and the internal causal mechanisms (or behaviors) that specify how the structure achieves the functions. The SBF model of the proposed design helps in diagnosing the design and in identifying the needed repairs. The Kritik system [Goel 1991a,1991b] demonstrates the feasibility and utility of this model-based approach to device redesign.

Kritik, however, (implicitly) makes an important assumption: the internal causal behaviors in the SBF model of the proposed design are complete. This assumption seems reasonable under two conditions: (i) all constraints on the functions desired of the design are known and static, and (ii) the desired functions differ from the functions delivered by a design stored in case memory only in the values of one or more variables. Under these conditions, the SBF model of the matching stored design can be assumed to be complete with respect to the new design problem. This completeness assumption, however, is invalid if either of the two conditions is false.

In this paper, we relax the condition about the rigidity of constraints on the design, and extend Kritik's approach to accommodate ***discovery*** of new design constraints. New constraints on the design may be discovered, for example, from the performance of the designed device in a real environment. In general, a designer might not anticipate all environments in which the device might be used, and, therefore, may not pre-enumerate the constraints imposed by the environments. Thus, in general, some constraints on the design are discovered only through observing the interactions of the device with its environment. We call this ***performance-driven innovation*** **[PDI]** [Prabhakar and Goel 1992].

Since, in general, the SBF model of the design may not be complete with respect to the newly discovered constraints, the issue becomes how to synthesize new SBF models useful for redesign problem solving. Our approach is based on a theory of incremental learning of new SBF models by revising old ones [Goel 1991a]. In

From: AAAI Technical Report FS-92-03, Copyright © 1992, AAAI (www.aaai.org). All rights reserved.

addition to past designs and their SBF models, we posit that designer's often may know of ***prototypical design failures*** and the causes for them. This idea of prototypical failures and their causes is borrowed from our earlier work on taxonomic organization of causal knowledge for diagnosis [Prabhakar, et al 1990]. Knowledge of prototypical failures and their causes can be expressed as a causal model for the failure. Given a specific type of failure of a design, the causal model for the corresponding prototypical failure can be instantiated and composed with the incomplete SBF model of the design to obtain a new and more complete SBF model useful for redesign.

2 An Illustrative Example

We illustrate our aproach to model synthesis in device redesign, through an example of household automatic coffee makers. Let us suppose that a designer takes as input a set of constraints that specify the functionality desired of the coffee maker, namely, automatic brewing of coffee-decoction from coffee powder and hot water. The design that realizes this function is shown in Figure 1. This design has two containers, with container-1 positioned above the container-2. The design also has a filter in between these two containers. The filter is positioned such that it makes contact with the bottom surface of the container-1. Initially, container-1 contains coffee powder. As hot water is added to it, the coffee powder dissolves in the water, forming a mixture of coffee powder, water and coffee decoction as indicated in Figure 1A. Coffee decoction permeates through the filter and gets collected in container-2 as illustrated in Figure-1B. Note that this design for the coffee maker satisfies all the initial constraints on the given design problem.

Now suppose that the design is instantiated in a prototype coffee maker and it is used in a real setting, it is found that while the coffee maker fails to deliver required output behaviors in some operating environments. Specifically, the coffee-maker delivers luke-warm coffee-decoction in a cold environment.

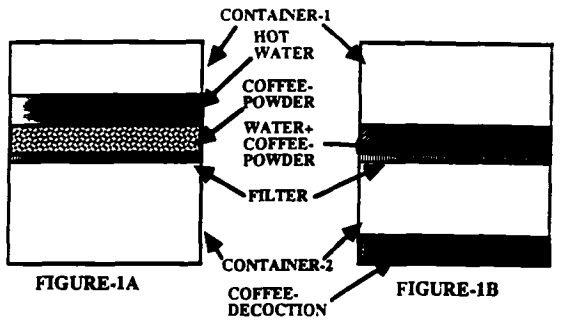

FIGURE 1: A Design for the Coffee Maker

The designer may reason about this output behavior by first forming a causal explanation for it. He might infer that the reason why the coffee decoction reaches the bottom container only lukewarm is that the coffee making process is taking a long time and during this time heat escapes from the coffee decoction in the top container to the environment. This causal explanation may lead the designer to the formulation of new constraints on the design problem. Specifically, the time taken to brew the coffee needs to be reduced significantly. Given the new design constraints, and the causal explanations for undesirable output behaviors, the designer might create a new design. He might, for example, introduce a plunger into the top container to speed up the process of coffee brewing. This, he might reason, would reduce the time the coffee decoction remains in the top container. In fact, this is precisely the design of many modern coffee makers.

Note also that the ``causal explanations" here are of a very specific kind: they relate the structure of the coffee maker to its output behaviors, where the output behaviors are a superset of the functions of the device. That is, the causal explanations specify the processes and mechanisms by which the structure produces its output behaviors. The SBF models capture precisely these internal causal behaviors of the device.

3 SBF Models of Physical Devices

The SBF model of a device explicitly represents its structure, intended functions, and internal causal behaviors. Kritik's SBF models are based on a ***component-substance ontology*** of CONSOLIDATION [Bylander and Chandrasekaran 1985]. The SBF model of a device is specified in ***behavioral representation language*** that builds on **Functional Representation Scheme** (FR Scheme) [Sembugamoorthy and Chandrasekaran 1986].

The structure of a device in Kritik as constituted of components (e.g., battery, pipe, container), substances (e.g., water, electrical charge, coffee-powder) and relations between them (e.g., connection, containment). The substances can be abstract, e.g., heat. The components and substances can have behavioral interactions. For example, substances can be allowed to flow from one component to another if there is a certain type of structural relation between the two components, viz., the connection relation. A substance can also have behavioral interactions with other substances. This is represented as a substance has a property with-respect-to another substance. For example, coffee-powder can dissolve in water. This is represented as water has solubility-capacity of medium magnitude with-respect-to coffee-powder. The typology of primitive behavioral interactions is borrowed from the consolidation. For example, a battery ***pumps*** electrical charge and a pipe ***allows*** substances with certain properties. Knowledge of the structure of the device is organized in a structure-substructure hierarchy. A substructure is represented as a schema that specifies its functional abstractions, its modalities, parameters and

From: AAAI Technical Report FS-92-03. Copyright © 1992, AAAI (www.aaai.org). All rights reserved.

their operating ranges, its substructures and behavioral state transitions.

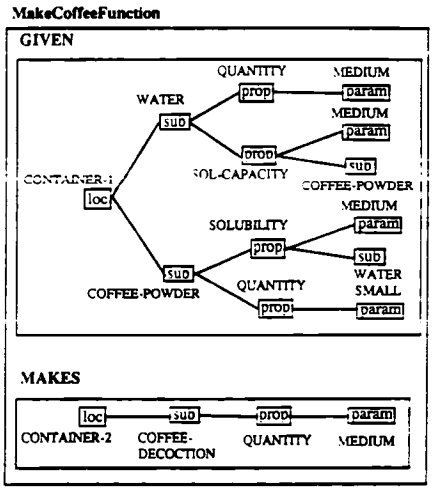

FIGURE 2. The Function of the Coffee Maker

A function of a device is viewed as a transformation from one behavioral state to another. Knowledge of a function is further represented in the form of a schema. The function schema specifies the behavioral state it takes as input and the behavioral state it gives as output. For example, the function, MakeCoffeeFunction, shown in Figure 2, has two states. The given state of this function specifies that the coffee powder and water are in container1. The makes state specifies that the coffee decoction is in container2. A function also specifies the internal causal behavior that results in the function, and the stimulus from the environment which triggers behaviors. For example, the MakeCoffeeFunction specifies the internal causal behavior, MakeCoffeeBehavior. The scheme for using functions as indices to causal behaviors is borrowed the FR scheme.

An internal causal behavior of a device is viewed as a sequence of behavioral state transitions. Knowledge of a behavior is represented as a directed acyclic graph of behavioral states and state transitions. The internal behavior of coffee maker called MakeCoffeeBehavior, is shown in Figure 3.

A state in a behavior is represented in the form of a schema. The state schema specifies the location, property, and parameters and parameter values of a substance. For example, in state1 of Figure 3, (some quantity of) Water is at the location, container1 in the device space and has the properties of temperature and solubility with values T1 and S respectively. The state schema may also specify the substances contained within a substance.

A transition in a behavior is annotated by the primitive behavioral interactions that cause the state transition to occur, e.g., the transition from one state to another is caused by a primitive function. A transition may also be annotated by the enabling conditions under which the behavioral interactions result in the transition. The enabling conditions in one behavior may point to another behavior. Finally, a transition may be annotated by domain principles underlying the transition, and the parametric relations between the state variables characterizing the preceding and succeeding states. For example, Figure 3 shows a state, representing the container1 with water, coffee-powder and coffee-decoction, makes a transition to another state, representing the container2 with coffee-decoction. This state transition has three annotations. 1. The first annotation specifies a ***contact*** relation between bottom surface of container1 and the filter. This is a restriction on the flow of substances between components which says that if a substance flows through one component in ***contact*** with another, then it has to go flow through the other one also if it ***allows*** the substance. 2. The function ***allow*** of bottom-surface of container1 specifies that it allows the

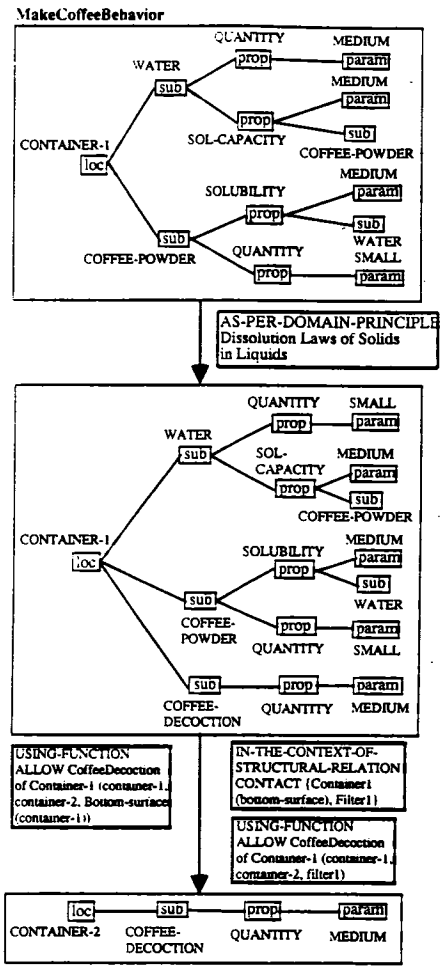

FIGURE 3. The Internal Causal Behavior of Coffee Maker

From: AAAI Technical Report FS-92-03. Copyright © 1992, AAAI (www.aaai.org). All rights reserved.

coffee-decoction to flow from container1 to container2 through the bottom-surface of container1. 3. Filter1 also has a function ***allow*** which specifies that the filter allows the coffee-decoction to flow from container1 to container2.

4 Causal Models of Prototypical Failures

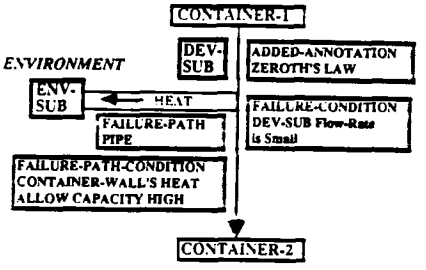

Figure 4: A Causal Model of the Prototypical Failure due to Heat Loss

Figure 4 illustrates the causal model for the prototypical failure of cooling of substances due to heat loss to the environment. This process model is based on the same component-substance ontology and described in the same behavioral representation language as the SBF model of specific devices. As illustrated in Figure 4, the cooling of substances due to heat loss to the environment is modeled in terms of generic components (container1, container2 and pipe) and generic substances (dev-sub and env-sub). The model specifies that heat flows from dev-sub to env-sub via the pipe even as the dev-sub flows from container1 to container2. The model points to the Zeroth Law of Thermodynamics as the physics principle governing the heat flow. (This law states that when two substances or components with different temperatures come in contact, heat from the substance/component at the higher temperature flows to the substance/component at the lower temperature.) In addition, the model specifies the two conditions under which the heat flow may occur: (i) the failure-path-condition pertains to the device-environment interaction and specifies that the walls of container-1 and container allow the flow with a large capacity; and (ii) the failure-condition pertains to the internal behavior of the device itself and specifies that the rate of flow of dev-sub is small.

5 Composing Design and Failure Models

The new SBF model that can explain the performance of a device is synthesized from the old SBF model of the device that explains how the device achieves its intended functions and process models of prototypical device failures stored in memory. The synthesis of the new SBF model involves two steps: ***instantiation*** of the process model of the prototypical failure in the context of the old SBF model of the device, and ***composition*** of the instantied process model with the old SBF model.

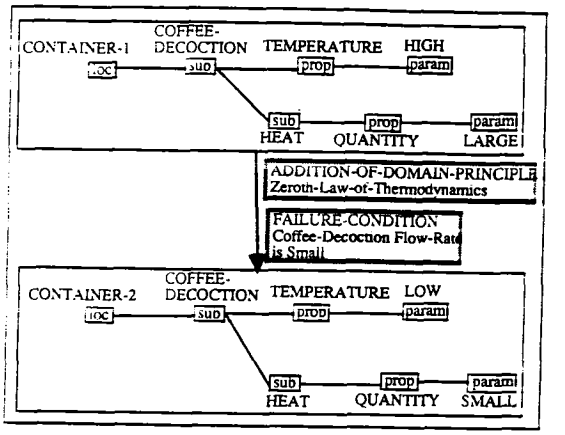

Figure 5: An Instantiation of the Causal Model of Heat Loss

Figure 5 illustrates the instantiation of the causal model for substance cooling in the context of the SBF model of coffee maker. The instantiated causal model is composed with the SBF model of the coffee maker by ***failure state*** links. The behavioral states in design and failure behaviors are linked by failure state extension links.

6 Constraint Discovery

The point in synthesizing the new SBF model is that it helps to formulate new design constraints. Once formulated, the redesign process can attempt to accommodate the new constraints. The coffee maker illustrated in Figure 1 is constrained with old constraints. The goal is to create a design that is satisfies the old and new constraints.

Figure 6 shows a process model for discovery of new constraints and (re)design of a completely constrained design. The key idea is that new constraints are to violate the ***failure-condition*** and ***failure-path-condition*** in the causal model of prototypical failure shown in Figure 4. This is because the old design failed due to these conditions, and, therefore, if these conditions can be negated, then the causes for the design failure can be corrected. The violation or negation of these conditions generates new functional constraints on the design.

In the coffee maker example, for instance, the violation of the ***failure-condition*** of Figure 4 leads to the formulation of a new function illustrated in Figure 7. The ***failure-condition***, shown as an initial state in Figure 7, specifies that the coffee-decoction is at the location in between container-1 and container-2 and its flow-rate is small. The violation of this condition generates the final state of Figure 7, which specifies that the flow-rate of

From: AAAI Technical Report FS-92-03. Copyright © 1992, AAAI (www.aaai.org). All rights reserved.

coffee-decoction be large. This forms a new functional constraint for the design to satisfy.

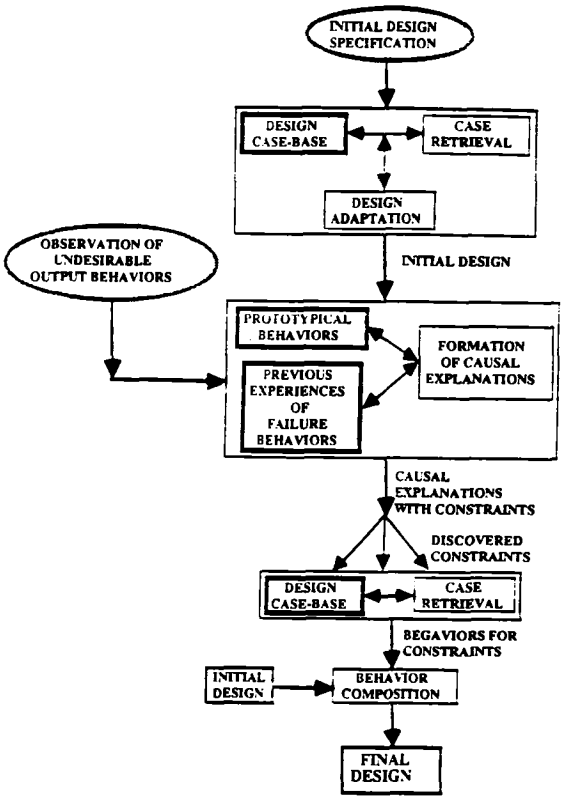

FIGURE 6. A Process Model of Constraint Discovery and Redesign

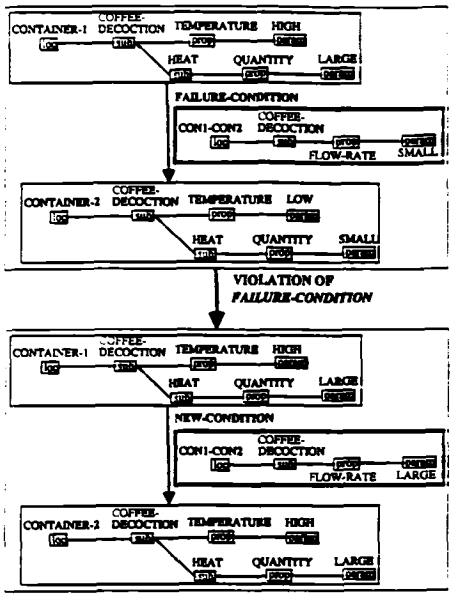

FIGURE 7. Formulation of New Constraint

7 Case Composition

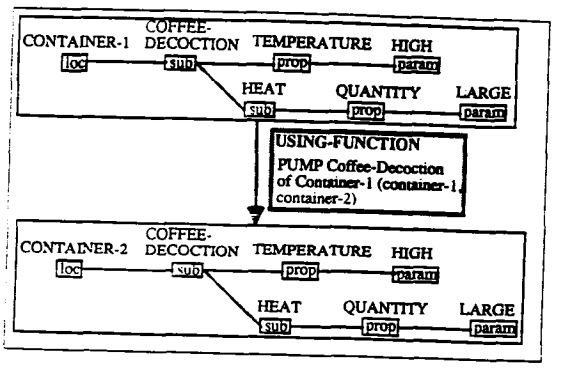

FIGURE 8. The PUMP function of Plunger Retrieved from Memory

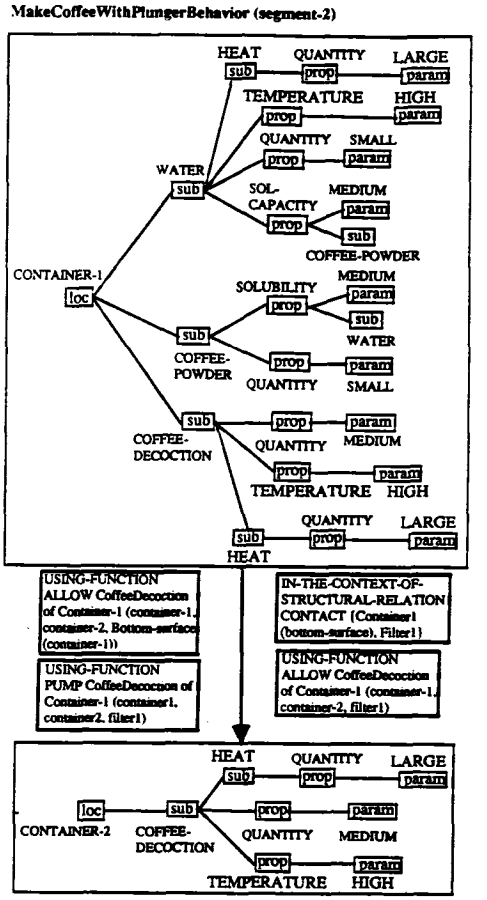

FIGURE 9. New Design for Coffee Maker with Plunger

As indicated in Figure 6, once the new functional constraints are gnerated, the old design is redesigned by retrieving additional design cases from memory and composing them with the old design. To return to the coffee maker example, the new functional constraint shown in the bottom half of Figure 7 acts as a probe into the case memory. The case memory is organized functionally as in Kritik [Goel 1991b]. That is, the

From: AAAI Technical Report FS-92-03. Copyright © 1992, AAAI (www.aaai.org). All rights reserved.

designs cases are indexed by the functions they deliver, where the design functions are specified in the language illustrated in Figure 2.

Let us suppose (for the sake of completing this argument) that the case memory contains a design for Plunger whose functional specification is shown in Figure 8. When the new functional constraint is sent to probe the case memory, the redesigner retrieves the design for the plunger and adds it to the design of the coffee maker. The new design for the coffee maker is shown in Figure 9. Note that although we have not discussed the ***failure-path-condition*** in Figure 4 in detail, it too can be accommodated in a similar manner.

8 Related Research

Our work differs from earlier work on redesign, e.g. [Marcus, Stout and McDermott 1988], in three major ways: our work discovers new constraints, uses causal models of device-independent prototypical failures, and reuses past designs in designing and redesigning new ones. Our work on model-based design has much in common with other research work such as [Williams 1991]. Both Williams and we use teleological and causal models of physical devices and processes. We differ, in the sense that the device designs and models, in our method, are created by retrieving, adapting and composing past designs and models stored in memory for reuse. PDI assumes that constraints are dynamic, not static. Our work in case-based reasoning is similar to work by Navinchandra [Navinchandra 1991]. We differ in the sense while Navinchandra's approach focuses on exploring the case memory, our approach emphasizes the grounding of case-based method in a deeper understanding of domain principles and processes. Finally, our work is related to recent research on innovative and creative design. The PDI introduces new variables into the design knowledge. This is defined as creative design [Gero 1990].

REFERENCES

[Bylander and Chandrasekaran 1985] T. Bylander and B. Chandrasekaran. "Understanding Behavior Using Consolidation".*Proc. Ninth International Joint Conference on Artificial Intelligence*, pp. 450-454, 1985.

[Gero 1990] J. Gero. "Design Prototypes: A Knowledge Representation Schema for Design". AI Magazine, 11(4): 26-36, Winer 1990.

[Goel 1991a] A. Goel. Model Revision: A Theory of Incremental Model Learning. In *Proceedings of the Eighth International Conference on Machine Learning*, Chicago, June 27-29, 1991, pp. 605-609, Los Altos, CA: Morgan Kaufmann.

[Goel 1991b] A. Goel. Representation of Design Functions in Experience-Based Design. In *Proceedings of the International Federation for Information Processing WG 5.2 Conference on Intelligent Computer-Aided Design*, Columbus, Ohio, 1991, pp. 269-292.

[Goel and Chandrasekaran 1989] A. Goel and B. Chandrasekaran. "Functional Representation of Design and Redesign Problem Solving". *Proc. Eleventh International Joint Conference on Artificial Intelligence*, Detroit, Michigan, pp. 1388-1394, 1989.

[Goel and Chandrasekaran 1990] A. Goel and B. Chandrasekaran. A Task Structure for Case-Based Design. In *Proceedings of the 1990 IEEE International Conference on Systems, Man, and Cybernetics*, Los Angeles, California, November 4-7 1990, pp. 587-592.

[Goel and Prabhakar 1991] A. Goel and S. Prabhakar. "A Control Architecture for Model-based Redesign Problem Solving". *Artificial Intelligence in Design Workshop, Proc. Twelfth International Joint Conference on Artificial Intelligence*, pp.121-136, Sydney, Australia, Aug. 1991.

[Marcus, Stout and McDermott 1988] S. Marcus, J. Stout, and J. McDermott. "VT: An Expert Elevator Designer that Uses Knowledge-Based Backtracking". *AI Magazine*, 9(1):95-114, 1988.

[Navinchandra 1991] D. Navinchandra. Exploration and Innovation in Design. New York: Springer-Verlag, 1991.

[Prabhakar and Goel 1992] S. Prabhakar and A. Goel. Using diagnostic experiences in experience-based innovative design. In *Proceedings of Applications of Artificial Intelligence X: Knowledge-Based Systems*, pp. 420-434, 1992.

[Prabhakar, Murthy and Patnaik 1990] S. Prabhakar, I.S.N. Murthy and L.M. Patnaik. "Diagnosing Multiple Faults using a Taxonomic Organization of Deep Level Causal Knowledge". *AAAI Spring Symposium Series: AI in Medicine*, Stanford University, 1990.

[Sembogamoorthy and Chandrasekaran 1986] V. Sembugamoorthy and B. Chandrasekaran. "Functional Representation of Devices and Compilation of Diagnostic Problem Solving Systems". In Experience, Memory and Reasoning, J. Kolodner and C. Riesbeck (editors), pp. 47-73. Hillsdale, New Jersey: Erlbaum, 1986.

[Williams 1991] B. Williams. "Interaction-based Design: Constructing Novel Devices from First Principles". In *Proceedings of IFIP International CAD Conference*, Columbus, Ohio, 1991.