A Functional Theory of Design Patterns

Sambasiva R. Bhatta
NYNEX Science & Technology
500 Westchester Ave.
White Plains, NY 10604, USA.

Ashok K. Goel
College of Computing
Georgia Institute of Technology
Atlanta, GA 30332-0280, USA.

Abstract

Design patterns specify generic relations among abstract design elements. In the domain of physical devices, design patterns, called *generic teleological mechanisms* (or GTMs), specify generic functional relations and abstract causal structure of a class of devices. We describe a functional theory of acquisition, access, and use of GTMs, but focus on their use in analogical design. In this theory, GTMs are acquired by abstraction over known designs, accessed by goals of adapting a familiar design to meet new design requirements, and used by instantiation in the context of a familiar design. This account of design patterns is one part of a general theory of analogical design called *model-based analogy* (or MBA). The IDEAL system implements the MBA theory for conceptual design of physical devices and evaluates its account of design patterns.

1 Introduction

Design patterns specify generic relations among abstract design elements. The relations are generic in that they are independent of any specific design situation and the elements are abstract in that they do not refer to any specific physical structure. We focus on a specific kind of design patterns that specify generic functional relations and abstract causal structure of a class of physical devices. We call these functional and causal design patterns *generic teleological mechanisms* (or GTMs). The abstract concept of feedback in control systems is one example of a GTM; a generic mechanism for transforming translational motion into rotational motion is another example. The feedback GTM, for example, specifies both the generic function it achieves (e.g., regulation of a device output, given possible fluctuations in the device input) independent of any specific design situation, and the abstract causal structure that achieves it (e.g., transmission of information about fluctuations in the device output to a device control input) without reference to the physical structure of any particular device.

We describe a functional theory of acquisition, access and use of GTMs. In this theory, GTMs are acquired by abstraction over known designs, accessed by goals of adapting a familiar design to meet new design requirements, and used by instantiation in the context of a familiar design. In particular, we hypothesize that knowledge of structure-behavior-function (SBF) models of specific designs enables the acquisition, access, and use of GTMs. SBF model of a device specifies the internal causal behaviors of the device that explain how the device works, i.e., how the device structure delivers its functions. This account of design patterns is one part of a general normative theory of analogical design called *model-based analogy* (or MBA). The IDEAL system instantiates the MBA theory for conceptual design of physical devices and evaluates its account of design patterns. In this paper, we briefly describe the MBA theory, focusing on our hypothesis about SBF models enabling the access and use of GTMs in analogical design. [Bhatta and Goel, 1997] describes model-based learning of GTMs.

2 Model-Based Analogy

The process of MBA takes as input a specification of a target design problem in the form of the functional requirements and structural constraints on a desired design, and gives as output a solution in the form of a structure that realizes the specified function(s) and also satisfies the structural constraints. In addition, MBA gives an SBF model that explains how the structure realizes the desired function. Figure 1 illustrates a part of the MBA process that pertains to GTMs. A stored design analogue in this process specifies (i) the functions delivered by the known design, (ii) the structure of the design, and (iii) a pointer to the causal behaviors of the design (the SBF model). The design analogues are indexed both by the functions that the stored designs deliver and by the structural constraints they satisfy.

If a source analogue that exactly matches the target problem cannot be found in memory, MBA spawns reasoning goals for adapting the source design. Different types of functional differences between the target and the source lead to different types of adaptation goals, some requiring only simple modifications (such as parameter tweaks) and some others requiring more complex modifications (such as topological changes). In order to control the reasoning involved in making complex modifications, MBA requires knowledge that can encapsulate the relationships between candidate modifications and their causal effects. In device design, design patterns, and, in particular, GTMs, provide such knowledge. Therefore, MBA uses the knowledge of GTMs in modifying device topology

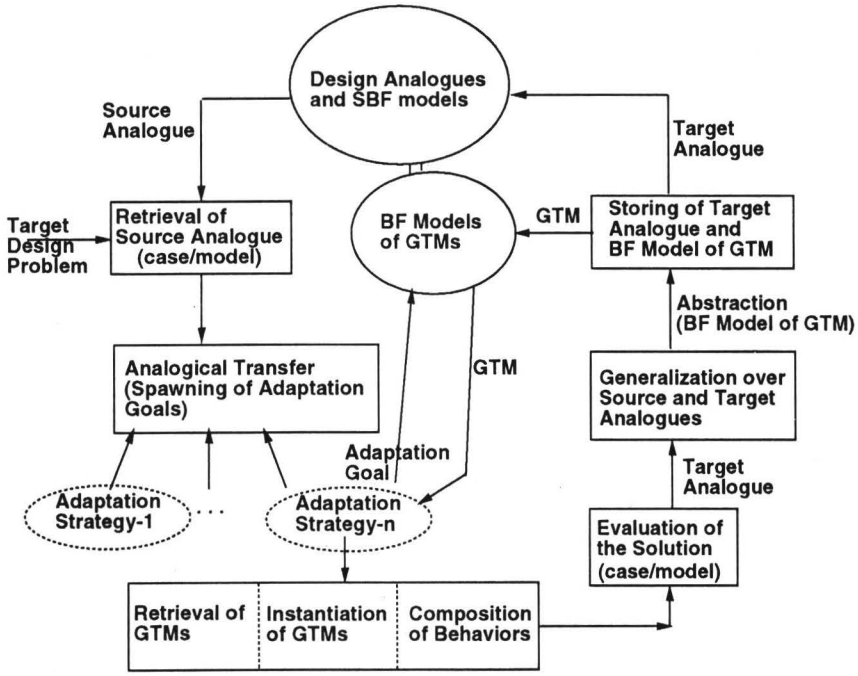

Figure 1: A Portion of the MBA Process Relating to GTMs

in the source design. MBA evaluates a modified design by qualitative simulation of its SBF model.

SBF Device Models

IDEAL represents its comprehension of specific design cases (i.e., device models) in a structure-behavior-function (SBF) language [Goel *et al.*, 1997]. This language provides conceptual primitives for representing and organizing knowledge of the structures, behaviors, and functions of a device. In this representation, the **structure** of a device is viewed as constituted of *components* and *substances*. Substances have *locations* in reference to the components in the device. They also have *behavioral properties*, such as *voltage* of *electricity*, and corresponding *parameters*, such as *1.5 volts*, *3 volts*, etc. Figure 2(a, b, c) illustrates the SBF model of a simple design of gyroscope follow-up: the structure, its function, and the behavior that achieves the function are shown.

A **function** in the SBF models is a behavioral abstraction, and is represented as a schema that specifies the behavioral state the function takes as input, the behavioral state it gives as output, and a pointer to the internal causal behavior of the design that achieves the function. The pair of states indicated by GIVEN and MAKES in Figure 2 (b) shows the function of the simple gyroscope follow-up. Both the input state and the output state are represented as *substance schemas*. Informally, the function specifies that the device takes as input angular momentum of magnitude L_i and clockwise direction at the input (gyroscope) location, and produces a proportional angular momentum of magnitude L_o and of clockwise direction at the output shaft location. L_o fluctuates over a large range, i.e., $L_o = L_{avg} \pm \Delta$, where Δ is large. Note that while the representation of a specific design may specify fluctuations in terms of quantitative tolerance limits (e.g., L_o and Δ may be numbers), the representation of a design pattern would specify fluctuations in terms of qualitative abstractions such as *small*, *medium* or *large*, independent of specific quantitative values.

The internal causal **behaviors** in the SBF model of a device explicitly specify and explain how the functions of structural elements in the device get composed into device functions. The annotations on the state transitions express the *causal*, *structural*, and *functional contexts* in which the transformation of state variables, such as substance, location, properties, and parameters, can occur. Figure 2(c) shows the causal behavior that explains how angular momentum from the input gyroscope location is transferred to the output shaft location. The functional context specified by the annotation USING-FUNCTION in $transition_{3-4}$ indicates that the transition occurs due to the primitive function "CREATE Angular Momentum" of Hydraulic-Motor.

Design Patterns

IDEAL represents GTMs as BF (Behavior-Function) models using a subset of the SBF language as above. The SBF representation of a GTM encapsulates two types of knowledge: knowledge about the patterns of differences between the functions of known designs and desired designs that the GTM can help reduce; and knowledge about patterns of modifications to the internal causal behaviors of the known designs that are necessary to reduce the differences. That is, it specifies relationships between patterns of functional differences and patterns of behavioral modifications to reduce those functional differences. For example, Figures 3 (a) & (b) respectively show these two types of knowledge for a partial model of

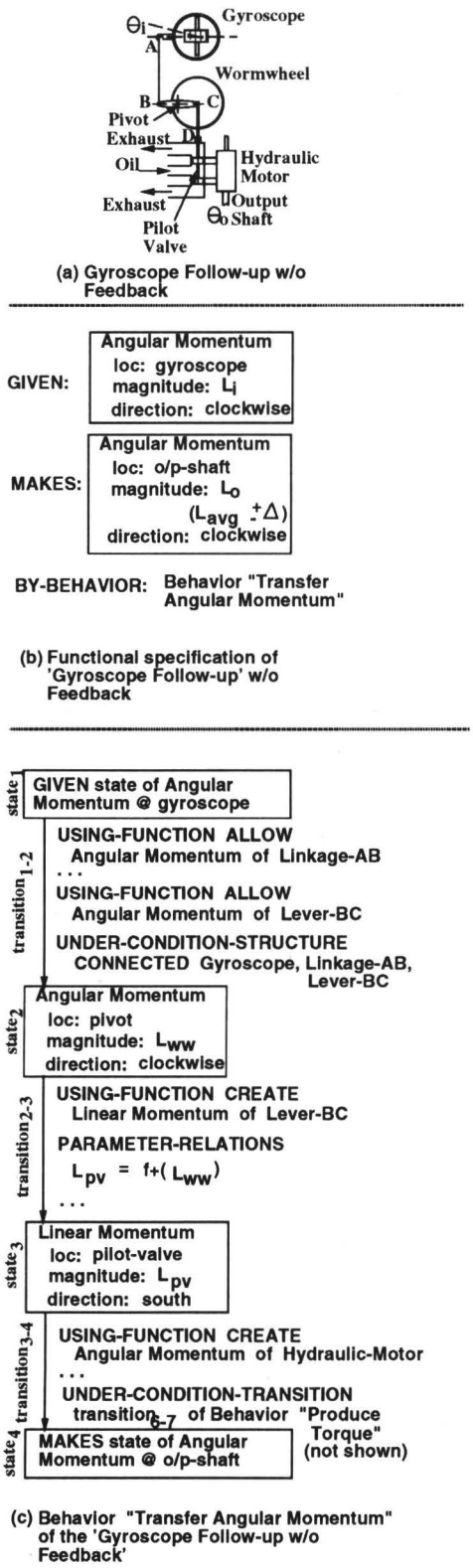

Figure 2: Simple Design of Gyroscope Follow-up (without feedback control)

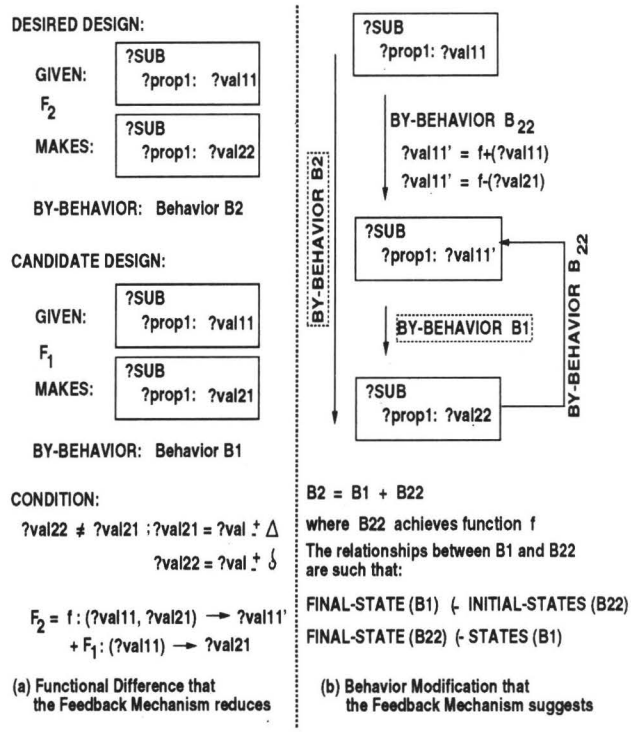

Figure 3: Feedback Mechanism IDEAL Learns

the feedback mechanism.[1] Figure 3 (a) shows the patterns of functions F_1 and F_2 respectively of a candidate design available and the desired design, and the conditions underwhich the mechanism is applicable. Because of the tasks for which they are used in MBA, the GTMs are indexed by the patterned functional differences such as shown in Figure 3 (a) (i.e., the fluctuations in the output substance property values in the candidate design function and the desired design function respectively are large and small). The model of the feedback indicates that the desired behavior (B_2) can be achieved by modifying the candidate behavior (B_1) through setting up the indicated causal relationships between the latter and the additional behaviors (that achieve the subfunctions of F_2 other than F_1 characterized in the conditions of the mechanism). In particular, the feedback mechanism suggests the addition of a causal link from a change in the output substance state to a change in an earlier state (input state or intermediate state) in the candidate behavior so that the effective input to the device is modified. Figure 3 (b) shows the relationships in the model of the feedback that IDEAL learns from two designs of amplifiers (one without feedback and the other with feedback). [Bhatta and Goel, 1996; 1997] provide a detailed account of the learning task and method.

[1] Feedback can be open loop or closed loop. The feedback mechanism described here is one type of closed-loop feedback in which the output substance, fedback substance, and the input substance are all same.

3 Access and Use of GTMs

Let us now consider a design problem in the domain of mechanical controllers presented to IDEAL. The new problem has a functional specification that given the substance angular momentum with a magnitude of L_i and clockwise direction at an input location (gyroscope), the device needs to produce the angular momentum with a magnitude L_o' proportional to the input and the same direction at a specified output location. It also specifies the constraint that the output cannot fluctuate much around an average value (i.e., $L_o' = L_{avg} \pm \delta$, where δ is small). This is the problem of designing a gyroscope follow-up [Hammond, 1958].

Let us consider the knowledge condition in which the design of a device (Figure 2) which transfers angular momentum from a gyroscope to an output shaft location is available in IDEAL's analogue memory (or is given explicitly as part of the *adaptive* design problem). Given an input angular momentum of magnitude L_i and clockwise direction at the input (gyroscope) location, this device produces a proportional angular momentum of magnitude L_o and of clockwise direction at the output shaft location; however, L_o fluctuates over a large range, i.e., $L_o = L_{avg} \pm \Delta$, where Δ is large. IDEAL retrieves (if not given explicitly) the design of gyroscope control system available in its memory because the desired function matches with the function of this design. That is, the function of this available device is similar to the function of the desired design in that the input states are identical and the output states differ in a parameter value and a constraint on that value.

Now, the task for IDEAL is to modify the available design of gyroscope control system (Figure 2) to deliver the desired function. Simple modifications such as replacing a component in the given design analogue will not result in a device that can solve the new design problem because there is no single component in the device that seems responsible for the large fluctuations and that which may be selected for modification. The issue becomes if and how IDEAL can modify the device topology using the knowledge of GTMs.

IDEAL first retrieves the relevant GTM: it uses the difference in the functions of the candidate and desired designs as a probe into its memory because it indexes the mechanisms by the functional differences and the decomposability conditions on the desired functions. It retrieves the feedback mechanism because the current functional difference, namely, the fluctuation in the output property being large vs. small (i.e., Δ *vs.* δ), matches with the difference that the feedback mechanism reduces which is specified in a device-independent manner. Then, it tries to match the decomposability condition on the desired function in the feedback mechanism (see Figure 3(a) for the condition $F_2 = f : (?val11, ?val21) \rightarrow ?val11' + F_1 : (?val11) \rightarrow ?val21)$ with the desired function in order to find the subfunctions f (or g) that need to be designed for and composed with the candidate function. By performing this match, as guided by the language of SBF models, IDEAL finds the subfunction $f{:}(L_i, L_o') \rightarrow L_{ww}'$, i.e., it needs to design for a structure that takes two inputs, angular momentum with a magnitude of L_i and angular momentum with a magnitude of L_o', and gives as output an angular momentum of L_{ww} in the opposite direction at the location of pivot in the candidate design.

Next IDEAL instantiates the retrieved GTM in the context of the target problem. The algorithm for IDEAL's process of instantiating the GTMs is shown in Figure 4. When the abstractions are GTMs, this process involves designing for the subfunction(s) determined by matching the applicability conditions of the mechanism (in steps 2 & 3 of the algorithm) and composing the new sub-behavior(s) with the behavior of the candidate design as per the relationships specified in the retrieved mechanism (in step 5). Let us walk through the algorithm as it applies to the current example. Step 1 is to select the behavior relevant for the function of the available design. Since the function in an SBF model of a device directly points to the behavior relevant for that function, this step is trivial. In the current example, B_1 is the behavior shown in Figure 2(c). Step 2 is to identify bindings for variables in the retrieved GTM, in particular, in the subfunctions to be designed for. Some of the bindings for the state variables are obtained while doing the matching for the retrieval of the GTM itself. As described above, in the current example, IDEAL finds the subfunction f to be $(L_i, L_o') \rightarrow ?val11'$ because ?val11 is the value of the property (whose output values in the desired and retrieved functions are different) in the initial state of B_1 and ?val21 is the value in the final state of B_1. Like in this example, even after step 2, some other variables such as $?val11'$ still need to be bound with specific values from the behavior of the available design. Step 3 is exactly for doing that: the idea is to trace the relevant behavior of the available design, B_1, backwards from the final state to the initial state, and identify the intermediate states that are possible candidates for the states of the subfunctions. In the current example, IDEAL needs to find a candidate state from B_1 that could be the output state of the subfunction f. As it traces back the behavior shown in Figure 2, the first state to be considered is $state_3$. But since it describes a substance (linear momentum) different from what the substance (angular momentum) is from the bindings in step 2, this state cannot be a candidate. Next, it considers $state_2$ which is the only state left and which is a candidate for the output state of f. If $state_3$ were to describe angular momentum, it would also have been a candidate. In such a case, IDEAL would have chosen $state_3$, the state nearest to the final state of B_1. The rationale in this is that the modification selected should cause as minimal disturbance in the candidate behavior as possible, which means modifying the state as near to the final state as possible in order to solve the problem. Since $state_2$ is selected in the current example, IDEAL gets the binding for $?val11'$ from this state, and it has all parts of the subfunction specified.

Since the subfunction has multiple states, step 4 is relevant. In the current design scenario, the subfunction IDEAL needs to design really has two parts (as it takes two inputs and produces one output): one that specifies the need for transferring angular momentum from the input location to the pivot location, and the other for transferring angular momentum from the output shaft location to the pivot location. Applying step 4, we can find that $transition_{1-2}$ really covers the transfor-

Input: • M_1, the SBF Model of the Design Analogue, and its Function F_1.
• F_2, the desired function.
• G, a GTM (retrieved by matching $F_2 \sim F_1$).

Output: • M_2, the SBF Model of the new device that achieves F_2.

Procedure:

begin

(1) Select the behavior B_1 in M_1 relevant to F_1.

(2) Bind the initial & final states of B_1 to the appropriate GIVEN and MAKES states of the subfunctions f and g in G.

(3) **if** $\exists$ an unbound state variable in f or g
then backtrace B_1 to find states in B_1 that may be modified, considering the bindings from step **2**.
(3.1) **if** $\exists$ multiple candidate states for modification
then Select the state that is nearest to the final state in B_1.
(3.2) Compute values of unbound state variables in f and g based on the selected state, $(F_2 \sim F_1)$, and PARAMETER-RELATIONS in B_1.

(4) **if** $\exists$ multiple GIVEN or MAKES states in f or g
then Check if $\exists\ b \in B_1$ that achieves the transformation from any of the GIVEN states to any of the MAKES states in f or g.
(4.1) **if** *yes*
then f' = rest of the transformation in f.
(i.e., < (GIVEN-states(f) - initial-state(b)), (MAKES-states(f) - final-state(b)) >.)
g' = rest of the transformation in g.
(i.e., < (GIVEN-states(g) - initial-state(b)), (MAKES-states(g) - final-state(b)) >.)

(5) Retrieve subdesigns for f' and g'.
(5.1) **if** $\exists$ no subdesigns for f' or g' **then** FAIL.
(5.2) **else**
(5.2.1) Adapt the retrieved subdesigns for f' and g' (if necessary).
(5.2.2) Compose $B_{f'}$, the behavior for f', and $B_{g'}$, the behavior for g', with B_1 as per the relationships in G.
(5.2.3) Propagate the resulting changes in state variables forward in B_1 and in the dependent behaviors in M_1.

end.

Figure 4: IDEAL's Method for Instantiating A GTM

mation $?val11 \rightarrow ?val11'$ and the remaining transformation (f') in f is $?val21 \rightarrow ?val11'$. That is, the first part is already designed for in the candidate design as the behavior segment $state_1 \rightarrow state_2$ (Figure 2(c)) achieves it. Therefore, in successfully instantiating the mechanism in the candidate design of gyroscope follow-up, IDEAL only needs to find a behavior (and a structure) that accomplishes the second part of the subfunction (f') given the context of the first transformation.

Let us consider the knowledge condition in which IDEAL has the knowledge of a component (called *worm*) whose function is to transfer an input angular momentum to an output location with the magnitude proportional to the output component and the direction dependent on the direction of threading on the worm. This component reverses the direction of the input angular momentum. In step 5, given the subfunction f', IDEAL retrieves that component because the desired part of the subfunction matches with the component's function. It substitutes the appropriate parameters in the behavior of the retrieved design (i.e., worm) to generate a behavior for the desired subfunction. Then it composes that behavior (i.e., B_{22}) with the behavior of the candidate design (i.e., B_1) as per the specification of the causal relationships in the feedback mechanism (as in Figure 3(b)) to propose a behavior (shown in Figure 5(b)) for achieving the desired function. Note that the resulting modification is non-local in that it modifies the device topology (see the structure of the desired device in Figure 5(a)). It finally propagates the changes in states resulting from composing the subdesign's behavior with B_1 forward to the final state or until a state is revisited.

4 Evaluation

IDEAL provides a testbed for experimenting with the MBA theory. We conducted several kinds of experiments with IDEAL that evaluate the MBA theory for its acquisition, access, and use of GTMs. One kind of experiment contained two steps. The first step involved giving IDEAL a pair of designs, one without any instance of GTM and the other with an instance of a GTM, and testing IDEAL's ability to learn a BF representation of the GTM instantiated in one of the two input designs. In the second step, IDEAL is given a design problem, from a different domain in some cases, such that it would need to access and use a previously learned GTM in order to solve the given problem. We verified if it can autonomously recognize the applicability of a GTM and successfully access and use it to solve the given problem. We conducted 12 such experiments with different combinations of design sources and target problems from 4 different design domains involving 28 distinct designs. The largest design had about 10 structural elements and 10 structural relations, and 3 inter-dependent behaviors in its SBF model. We tested IDEAL's learning of 6 different GTMs and its use of 3 of them. In all these cases, IDEAL was successful in learning GTMs, and in accessing and using them in solving design problems. The behavior of IDEAL in these experiments led us to conclude the following four results:

(1) Computational feasibility and efficacy: IDEAL successfully addresses the multiple tasks in the MBA theory, for

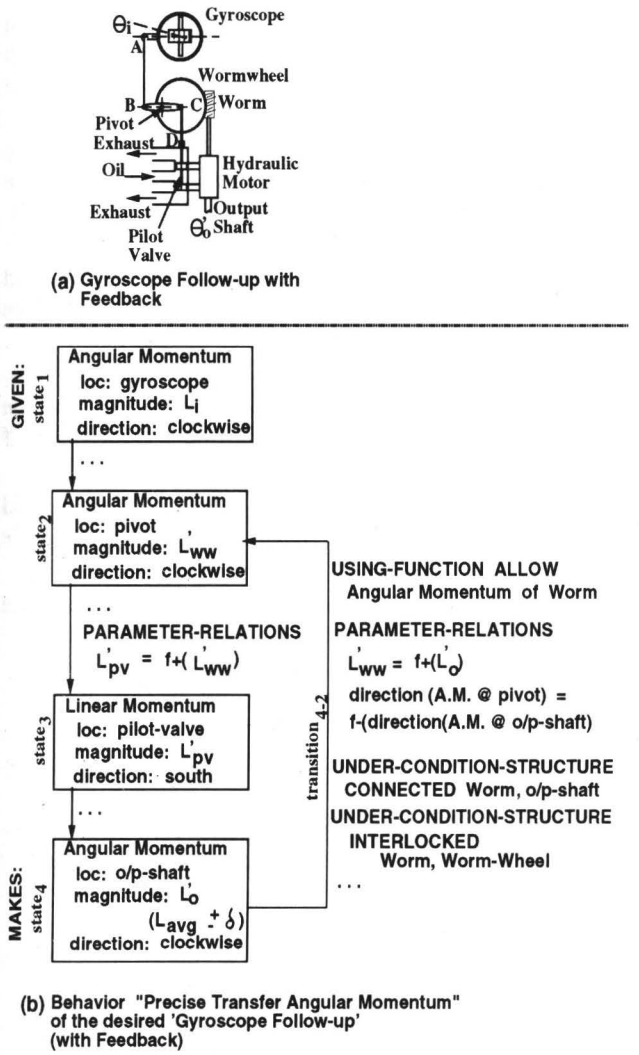

Figure 5: The Design of Gyroscope Follow-up Instantiating the Feedback Mechanism

example, the tasks of learning, accessing, transferring and using GTMs.

(2) Uniformity of representations: The different tasks in the MBA theory impose constraints on one another. They also impose different constraints on the design knowledge representations. IDEAL uses the same SBF language for addressing the different tasks. The GTMs, for example, are represented in a BF language that is a subset of the SBF language. The design analogues too are indexed in the SBF vocabulary.

(3) Generality of domains: As mentioned above, IDEAL presently contains about thirty design analogues from four different device domains, namely, the domains of simple electric circuits, heat exchangers, electronic circuits, and complex mechanical devices (such as momentum controllers and velocity controllers). This includes the design problem used as an illustrative example in this paper, which was taken from a classical textbook on mechanical design [Hammond, 1958].

(4) Generality in terms of different GTMs: IDEAL presently covers six different GTMs: cascading, four different types of feedback, and one type of feedforward.

5 Related Research

The notion of design patterns can be traced at least as far back as Christopher Alexander's [1964] *Notes on the Synthesis of Form* in which he provides a conceptual analysis of evolutionary design of village centers in rural India in terms of topological design patterns. Recently Gamma *et al.* [1995] have analyzed designs of object-oriented programs in terms of patterns of data and control flow, and described a library of twenty three reusable patterns for supporting interactive object-oriented programming. But insofar as we have been able to determine, at present there is no computational theory (other than MBA) of automated acquisition, access and use of design patterns. The MBA theory not only identifies GTMs as a useful class of design patterns in the domain of physical devices, but also shows that knowledge of SBF device models is sufficient for enabling automated acquisition, access and use of GTMs.

Our theory of design patterns is also related to Gero's theory of design prototypes [1990]. Like our design patterns, his design prototypes too specify functional relations and causal structures in a class of devices, but, unlike a design pattern, a design prototype also specifies the generic physical structure of the device class. While a design prototype is a generalization over design cases such that a case is an instance of a prototype, a design pattern is an abstraction over design prototypes such that a design prototype is a subclass of a design pattern. IDEAL too contains design prototypes as an intermediate abstraction between design cases and design patterns, but uses only the latter for analogical transfer. If the SBF model of the retrieved design case indicates that a local (i.e., parametric or componential) modification would be sufficient for meeting the new design requirements, then the system simply adapts the retrieved design. But if the SBF model does not identify a local modification, then the system accesses and instantiates a relevant design pattern to make a "non-local" (i.e., topological) modification. Furthermore, Gero's theory of analogical design uses a process similar to that of the structure-mapping engine (SME) [Falkenhainer *et al.*, 1989] to abstract causal behaviors at transfer time. In contrast, in MBA, design patterns are abstracted at storage time and acquired for potential reuse, and the process is different.

PHINEAS [Falkenhainer, 1989], which evolves from SME, also uses high-level abstractions for establishing correspondence between the source and the target situations. But it provides neither any content account of the high-level abstractions nor a process account of their acquisition. MBA provides both a content account of generic abstractions in relation to device design, and also a process account of the acquisition, access, and use of the abstractions. As in [Kedar-Cabelli, 1988], the generic abstractions in MBA are purpose-directed. In particular, in MBA the design patterns are indexed by the problem-solving goals stated in terms of functional differ-

ences between two design situations.

Case-based theories of design mostly involve direct transfer of the structure of familiar designs to new design situations. That is, the transfer is not mediated by high-level abstractions. In some case-based theories (e.g., Shinn, 1988), however, high-level abstractions do enable case reminding, but still play little role in analogical transfer. Also, the high-level abstractions in these case-based theories are generalizations over features of a problem, and do not specify relations that characterize a problem and its solution. Finally, design adaptations in case-based design are in general limited to local (typically parametric) design modifications.

The MBA theory evolves from an earlier theory of case-based design called *adaptive modeling* [Goel 1991a; 1991b]. The adaptive-modeling theory described case-specific structure-behavior-function (SBF) models. It showed how case-specific SBF models enable local (i.e., parametric or componential) modifications to source designs for solving target design problems. It also showed how case-specific SBF models of new designs can be acquired by adapting the models of known designs. Stroulia and Goel [1992] described how case-independent generic models enable topological modifications to source designs in the same domain as that of target problems. Bhatta and Goel [1996; 1997] showed how generic models can be acquired by abstraction over case-specific SBF models. The MBA theory completes the circle by showing how generic models mediate analogical transfer of design knowledge from the source domain to a target domain (e.g., from amplifiers to gyroscopes).

6 Conclusions

Design patterns in general specify generic relations among abstract design elements. GTMs, that specify generic functional relations and abstract causal structure of a class of devices, are a kind of design patterns useful for conceptual device design. In particular, GTMs mediate analogical transfer of design knowledge from one device domain to another, and enable topological modifications to familiar designs for meeting new functional requirements. SBF model of a specific device specifies the internal causal behaviors of the device which explain the functioning of the device. The SBF ontology provides a language for representing GTMs and a vocabulary for indexing them. In addition, knowledge of SBF models of familiar designs appears sufficient for acquisition, access, and use of GTMs.

Acknowledgments

This paper has benefited from numerous discussions with members of the Intelligence and Design research group at Georgia Tech. This work has been supported in part by research grants from NSF (IRI-92-10925 and DMI-94-20405) and ONR (research contract N00014-92-J-1234).

References

[Alexander, 1964] C. Alexander. *Notes on the Synthesis of Form*. Harvard University Press, 1964.

[Bhatta and Goel, 1996] S. Bhatta and A. Goel. From design experiences to generic mechanisms: Model-based learning in analogical design. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 10:131–136, 1996.

[Bhatta and Goel, 1997] S. Bhatta and A. Goel. Learning generic mechanisms for innovative strategies in adaptive design. *The Journal of the Learning Sciences*, 1997. Forthcoming.

[Falkenhainer *et al.*, 1989] B. Falkenhainer, K. Forbus, and D. Gentner. The structure-mapping engine: Algorithm and examples. *Artificial Intelligence*, 41:1–63, 1989.

[Falkenhainer, 1989] B. Falkenhainer. *Learning from Physical Analogies: A Study in Analogy and the Explanation Process*. PhD thesis, University of Illinois, Department of Computer Science, Urbana, IL, 1989.

[Gamma *et al.*, 1995] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, 1995.

[Gero, 1990] J.S. Gero. Design prototypes: A knowledge representation schema for design. *AI Magazine*, 11(4):26–36, 1990.

[Goel *et al.*, 1997] A. Goel, S. Bhatta, and E. Stroulia. Kritik: An early case-based design system. In Mary Lou Maher and Pearl Pu, editors, *Issues in Case-Based Design*. Erlbaum, Hillsdale, NJ, 1997.

[Goel, 1991a] A. Goel. A model-based approach to case adaptation. In *Proc. of the Thirteenth Annual Conf. of the Cog. Sci. Soc.*, pages 143–148, Chicago, August 1991.

[Goel, 1991b] A. Goel. Model revision: A theory of incremental model learning. In *Proc. of the Eighth Intl. Conf. on Machine Learning*, pages 605–609, Chicago, June 1991.

[Hammond, 1958] P. H. Hammond. *Feedback Theory and Its Applications*. The English Univ. Press Ltd., London, UK, 1958.

[Kedar-Cabelli, 1988] S.T. Kedar-Cabelli. Toward a computational model of purpose-directed analogy. In R.S. Michalski, J.G. Carbonell, and T.M. Mitchell, editors, *Machine Learning II: An Artificial Intelligence Approach*, pages 284–290. Morgan Kaufmann, Los Altos, CA, 1988.

[Shinn, 1988] H. S. Shinn. Abstractional analogy: A model of analogical reasoning. In Janet Kolodner, editor, *Proc. of the DARPA Workshop on Case-Based Reasoning*, pages 370–387, Clearwater Beach, FL, May 1988.

[Stroulia and Goel, 1992] E. Stroulia and A. Goel. Generic teleological mechanisms and their use in case adaptation. In *Proc. of the Fourteenth Annual Conf. of the Cog. Sci. Soc.*, pages 319–324, Bloomington, IN, August 1992.