Form–Function Synthesis in Engineering Design

U. Flemming, S. Finger,
J. Adams, C. Carlson, R. Coyne, S. Fenves,R. Ganeshan, J. Garrett,
A. Gupta, Y. Reich, D. Siewiorek, R. Sturges, D. Thomas, R. Woodbury
Engineering Design Research Center
Carnegie Mellon University
Pittsburgh, PA 15213
ujf@edrc.ri.cmu.edu sfinger@ri.cmu.edu

Abstract

In this paper, we present an overview of research on form/function synthesis for physical artifacts. This paper summarizes a lengthier technical report that characterizes the computational models underlying selected Engineering Design Research Center (EDRC) projects (Flemming et al., 1992a). This paper reviews how this problem is perceived in selected engineering disciplines, including architectural design, and classifies these models by their strategies and attempts to link these strategies to appropriate types of design problems[1] .

Introduction

This paper focuses on the synthesis of engineered artifacts. When we speak about the *form* of an artifact, we refer to the sum of its physical properties, from the shapes and materials of its components to the spatial relations that exist between its components. The *function* of an artifact denotes its intended behavior as well as its actual performance in a concrete environment and includes intended or unintended side-effects caused by its

The initial specifications for designs are usually of two types: Some specifications describe the intended function or desired behavior of the system at an abstract level, while others describe restrictions or requirements on the form of the final design. The former are usually called *functional* or *behavioral* and the latter *physical* requirements or specifications. For example, the requirements for a vibration absorber might include the behavior of the device in terms of frequency and rejection ratio and might also specify physical properties such as allowable size and weight. Taken together, the physical and functional specifications express the design objective. Physical specifications for a design may or may not be given, while some functional specifications must be given, since the functional specifications express the central design objective.

[1]This work has been supported by the Engineering Design Research Center, a NSF Engineering Research Center.

Although functional and physical specifications are independent in the functional domain, they are coupled in the physical domain. The physical and functional characteristics of individual components depend on one another, and the behavior of the whole design depends strongly on the configuration and interaction of components. Any form variable is likely to influence more than one function variable, and any function variable is likely to depend on more than one form variable. There are, in addition, dependencies between variables of the same kind. For example, maximizing the daylight in a room may have adverse effects on heat gains during the summer or heat losses during the winter.

Form or function variables can thus not be considered in isolation during the design process. However, neither humans nor computers can handle *all* form and function variables and their interactions simultaneously. The challenge is to devise a synthesis process that divides the overall task into more manageable subtasks by focusing on selected variables, while considering interactions and dependencies. In the following sections we give an overview of the form-function synthesis problem for the disciplines of architectural, mechanical, structural, and digital electronics design.

Architectural Design

Reflections about the relation between form and function in architecture are at least as old as the earliest written treatise to survive from antiquity, Vitruvius' *de Architectura* from the first century A.D. The debate continues to the present day; it cannot be reviewed here in any depth, and we restrict ourselves to a brief characterization of major positions.

At one extreme are the "functionalists" who adhere to Louis Sullivan's famous dictum that "form follows function." However, an inspection of the function of a building reveals that the criteria by which it is assessed are extremely varied; they may conflict with each other because they compete for the same limited resources (area, costs), or contradict each other for technical or other reasons such as conflicting interests of client, users or the public at large.

Furthermore, starting from a preconceived combi-

nation of functional characteristics is difficult because often it is not clear at the outset what types of components may be involved in a design. In the Kimball Art Museum in Fort Worth, Texas (designed by Louis Kahn), for example, the roof structure consists of a series of beams with a curved cross section, two of which can be combined to form a vault that lets light in from the top and diffuses it in combination with a suspended screen, which also diffuses sound. Thus two components perform structural, lighting and acoustic functions in addition to playing a major role in the overall architectural composition. We can imagine alternative solutions in which the same functions are performed by different types, even different numbers of components; e.g. light diffusion may be provided by skylights that are not part of the roof structure. Thus, mappings from function to form are generally not unique, and a straight-forward functional determinism appears not only undesirable, but impractical.

The other extreme in the form/function debate is occupied by architects for whom the form of a building is its only interesting aspect and who "build against function", as the architect Philip Johnson put it. This position may appear incomprehensible to observers outside the circles in which these debates take place; but the very fact that it can be formulated at all points out that the relation between a building and its function may be rather loose indeed. This is demonstrated by the high degree of adaptability to varying uses that buildings have shown over time.

Attempts to systematize and formalize architectural design in connection with computer-aided design tend to favor some form of generate-and-test; that is, they assume that a design must be specified to a certain level of completeness before it can be evaluated. This solves some of the problems created by the complex interactions between form and function: It is generally easier to evaluate a design according to multiple criteria and to discover conflicts than to develop a design directly from multiple and conflicting criteria.

Even if inferences from function to form are generally difficult in architectural design, certain research directions appear underexplored. Not every design problem requires the invention or novel assembly of components. For certain recurring combinations of functional requirements, prototypical component configurations are known. They could be retrieved from a properly constructed database and parametrically adjusted in accordance with a specific situation, including the systematic variation of form variables via the variation of function variables (Finger & Rinderle, 1990).

Mechanical Engineering

The designer of mechanical parts often moves directly from a statement of functional requirements to sketches of the artifact that will meet those requirements. Experienced designers can answer the question: "Given the functional requirements, what should the artifact look like?" Indeed, most mechanical design CAD systems are based on the assumption that the design proceeds from geometry. That is, CAD systems enable designers to create and manipulate the geometry of a design and then to analyze it for the required behaviors.

To create a mechanical part, a designer transforms an abstract functional description for a device into a physical description that satisfies the functional requirements. In this sense, design is a transformation from the functional domain to the physical domain; however, the criteria for selecting appropriate transformations and methods for accomplishing transformations are not well understood. Design transformations in circuits, software and some architectural applications lead to a degree and type of modularity not well suited to mechanical devices (Rinderle, 1986).

Good mechanical designs are often composed of highly-integrated, tightly-coupled components, where the interactions among the components are essential to the behavior and economy of the design. This assertion runs counter to methodologies in engineering fields such as software design and circuit design that result in designs in which each component fulfills a single function with minimal interaction. Because of the geometry, weight, and cost of mechanical components, converting a single behavioral requirement into a single component is often both impractical and infeasible. Each component may contribute to several required behaviors, and a single required behavior may involve many components. In fact, most mechanical components perform not only the desired behavior, but also many additional, unintended behaviors. In a good designs, these additional behaviors are often exploited.

Because of the difficulties of representing behaviors and their interdependencies, most mechanical designers begin by generating the geometric form of the artifact and then analyzing, for example, its structural, thermal, and kinematic behaviors. This observation can be verified by examining the available mechanical CAD packages. In all of them, the designer synthesizes the artifact by interactively creating the geometry and then invokes analysis tools to check whether the artifact meets its requirements; if it does not, the designer changes the geometry and repeats the cycle.

Finding useful representations for function and behavior for mechanical artifacts is difficult because of the coupling of behavior and form in mechanical parts. Only a few domains within mechanical engineering, such as kinematics, have standard representations that are used in design synthesis. For example, the Transformation Project presented in (Finger & Rinderle, 1990) begins with a bond graph representation for the behavioral requirements of a gear train. The bond graph representation captures only the requirements for power flow; other requirements must be stated in terms of constraints. Many of the requirements can be analyzed only after the gear train has been synthesized.

Structural Engineering

The role of structural engineering is, typically, to take the overall form of an artifact determined by other considerations and elaborate it into the detailed form of a structural configuration and its components to satisfy functional requirements of strength, safety, serviceability, etc. The overall form may be derived from formal or spatial considerations (e.g., the massing of a building or shape of an automobile), from functional considerations (e.g., the geometric layout of a crossing to be bridged dictated by traffic flow), or a combination of the two (e.g., the layout of a ship or aircraft dictated by both the spatial layout of the cargo and the fluid flow around it).

Structural design deals with the determination of the detailed form of the structural load-resisting system. The determination of the overall form is affected by the incorporation of structural functionality considerations. Thus, the form/function "debate" occurs at two levels. One level concerns the overall form of the structure: Is it driven by spatial considerations or by its load-carrying function? An elegant example of the latter is Eiffel's design for his 300-meter tower: the slope of the legs at any level is such that the axial forces acting along the legs intersect the resultant of the wind loads above that level; thus, there is no bending anywhere, resulting in a very efficient structure. Weaker forms of function-driven form are Fazlur Khan's Hancock building in Chicago, hyperbolic cooling towers, and most long-span bridges. In most cases, however, function is subordinate to form: Architects, plant designers, or highway layout engineers define the overall form, and structural engineers provide the best function they can within the spatial constraints.

The second level concerns the physical form of the structure: Which function determines the form? Typically, a structural system is an assembly of discrete components, each providing multiple functionalities. The strength of a wall may be dictated by gravity and load resistance functions and its stiffness by drift limitation (comfort) function. That same wall may, furthermore, serve as an architectural separation, as a thermal and noise barrier, and as a conduit for electrical and mechanical utilities. At this level, structural design is closely related to mechanical and architectural design, in that each component must satisfy several functional requirements, and each functional requirement is satisfied by several components.

Form-function synthesis in structural design is inherently a multi-level process, where each level contributes to some elaboration or refinement of the physical structure satisfying (or optimizing) a variety of functional requirements. The components may be concrete (e.g., a beam or wall) or abstract (e.g., a 3-D frame), and the functional requirements may be intrinsic to the load-carrying function (e.g., strength and stiffness) or extrinsic to it (e.g., spatial demands). In the synthesis of a structural system, which invariably interacts with other formal or behavioral functionalities of the overall artifact, function-to-form transformations are as likely to occur as form-to-function transformations.

Digital Electronics Design

In digital electronics design, the designed artifact is composed of an interconnection of pins on various components such as transistors, gates, macrocells, chips, or boards. The form of the artifact is considered at two distinct levels of detail: *logical form*, which does not include geometric information, and *physical form*, which includes geometric information. Digital electronics designers have three well-accepted perspectives of the design: function or behavior, structure, and geometry. The mapping from function to structure is called *logical design*, and the mapping from structure to geometry is *physical design*. Some form variables (e.g. board size and the set of available components) are provided as constraints to the synthesis process.

Physical design consists of two subtasks: placement and routing. In placement, a library of geometric information is used to dimension components, which are positioned on an area of specified dimensions to satisfy a number of possibly conflicting goals. These goals include increasing routability of the interconnections, reducing signal cross talk, and reducing heat dissipation levels; usually the primary goal is minimizing the total interconnection distance. The problems of placement and routing are intimately related; however, historically they have been treated separately due to the inherent computational complexity of the problem.

The synthesis of form in logical design occurs at three levels: logic synthesis, behavioral synthesis, and system-level synthesis. In logic synthesis, a set of Boolean expressions and finite state machines is mapped into an interconnection of gates and memory devices. Generating a structure that satisfies the specified function is a relatively straightforward procedure. The difficulty is in optimizing the form for a set of criteria (e.g. least size or least time delays).

In behavioral synthesis, an algorithmic description of a portion of a computer, such as a CPU, is mapped into an interconnection of register transfer elements, such as ALUs, registers, multiplexors, and buses. Finding feasible solutions to subtasks is relatively simple, but the search for optimal solutions is NP-complete. Even though the components are modular and fulfill a single function, a good design makes maximal use of each component. For example, one component can perform the same function on different operands. Heuristic algorithms that generate near-optimal solutions have been developed. The SAW system encapsulates a set of these algorithms (Thomas et al., 1990).

In system-level synthesis, a set of high-level functional requirements (e.g. instruction set or clock speed) is transformed into an interconnection of chips (e.g. processors, memory, peripheral chips, transceivers, and latches). At this level, generating a feasible solution is

difficult because the mapping from functional requirements to an interconnection of chips is not well understood. The MICON system (Birmingham et al., 1992) uses a knowledge-based approach in which the mapping is provided incrementally by a human trainer.

While the goals of these synthesis tasks are different, they have several common characteristics. In each, the design space is large, ruling out brute-force generate-and-test approaches. In addition, no good evaluators exist for partially designed artifacts; thus, incremental generate-and-test is not suitable. None of the steps has a single objective; rather, each pursues several competing objectives so that many pareto-optimal solutions exist. Furthermore, the design space is discrete and discontinuous due to, for example, step functions in memory sizes.

Discussion

These disciplines share a core of concerns that center around three related sets of issues: 1) Functional specifications normally do not map uniquely into component configurations. A large space of alternatives is available, and the structure of this space is often ill--understood and precludes the employment of standard search strategies. 2) The search for alternatives is complicated by the multifunctionality of components or by the fact that single-function components can be used in multiple ways. This makes it difficult to concentrate at any given time on a single functional characteristic or behavior aspect and thus eliminates some obvious problem decompositions. 3) Functional characteristics may conflict with each other. Therefore, tradeoffs must be taken into account, complicating the evaluations performed during synthesis and making optimization difficult to integrate into this process.

Among the disciplines reviewed here, digital electronics design has come closest to an accepted decomposition of an overall design problem into more manageable subproblems. Mechanical engineering appears to occupy the other extreme: it has a particular need for developing a theoretical base for formal specifications of behavior and function/form transformations.

Classification

The approaches underlying the EDRC projects fall naturally into three broad classes. The first of these encompasses *top-down* or *refinement strategies*, in which an overall description of an artifact or design objective is elaborated through several levels of abstraction until a sufficiently detailed specification has been achieved.

A second and contrasting class is formed by *bottom-up strategies*, which are also sometimes called *constructive strategies*. These strategies construct the physical structure of an artifact in a step-wise fashion at the same level of abstraction. The refinement strategies attempt to generate structures that are complete relative to the granularity associated with that level; the bottom-up strategies, in contrast, work more or less at the same level of granularity and produce a complete structure only at the end.

A third class uses strategies that start with a highly-structured description and perform transformations on this structure until a sufficiently detailed physical specification has been found. The transformations themselves may include refinement or constructive operations. We call these approaches *middle-out strategies*, although our use of this term is not entirely the same as that in the AI planning literature.

We can add additional stratifications to this classification by observing that within each strategy, one of two contrasting inference mechanisms may be at work: from function to form or from form to function, which correspond roughly to goal- and data-driven processes. In the following sections, we characterize these strategies and emphasize their prerequisites and the characteristics of the design problems for which they show promise.

Top-Down or Refinement Strategies

Function-Driven Strategies This strategy is demonstrated by the model underlying the Conceptual Design project (Sturges & Kilani, 1990), which proceeds by decomposing an initial overall goal or function statement recursively into conjunctions of goals or objectives until physical components can be selected directly to satisfy a subgoal. Alternative refinements may exist at any level, and the resulting hierarchy need not be strict; e.g. a component may satisfy more than one objective. This type of goal refinement is also represented by the recording mechanism underlying the Design Intent project (Ganeshan et al., 1991), which uses a refinement strategy that generalizes the approach used by (Garrett & Fenves, 1986). MICON also performs function-driven refinement; however, it refines parts not functions or objectives. We include it in the present class of strategies because the refinements are clearly function- or specification-driven.

An important prerequisite for top-down function-driven strategies is that knowledge about possible refinements be accessible. The first two projects rely on interactions with designers for this purpose and provide an interactive recording mechanism that reflects a particular *formal* model of the design process. The benefit is that neither project depends on pre-stored knowledge, that is, on routine situations or repetitive design problems. MICON also provides a mechanism to elicit refinement knowledge when needed from experts through its tutoring component. Thus, a knowledge base for automated refinements is built gradually over many problem runs. Any refinement hierarchy represents explicitly the couplings between goals and subgoals or between goals and components. Global interactions across goals or components must be handled by evaluations after selections have been made. We call such global evaluation mechanisms that are invoked *after* decisions have been made *critics*.

Form-Driven Strategies Form-driven refinement strategies implement part refinement through levels of abstraction. In contrast to function-driven strategies, the refinement is driven more by given part decompositions than by behavior specifications, although the latter may be important for selecting refinements. Global interactions again are handled by critics. This strategy is typically employed when the problem specification consists mainly of *physical specifications* that suggest at least good guesses about appropriate classes of component configurations. An example are the physical specifications from which BRIDGER (Reich, 1992) starts (required length, traffic volume, site characteristics). Each part can be viewed as defining a goal or objective: that of finding a refinement into less abstract subparts that satisfy the specifications associated with it. The Design Intent approach therefore can also work for form-driven refinement strategies.

Grammar-based approaches are applicable to form-driven refinement when the refinements are context--directed, where context is the current state of the design. This is particularly easy when the grammars use a represenation that is appropriate for any abstraction level. For example, a rectangular solid representing a partition at one level of abstraction can be refined into an internal frame covered by plaster board, where each refined component is again represented as a geometric solid. Some of the grammars that have been written in GENESIS can be viewed in this way (Heisserman, 1991); another example are the decompositions employed by ABLOOS (Coyne & Flemming, 1990). These processes reflect the progression through *scales* that characterizes e.g. building design.

The difficulties that arise with this strategy are similar to those mentioned in the preceding section. Knowledge about appropriate refinements must be accessible and pre-stored, which restricts applications to routine situations or assumes interactions with the designer. BRIDGER is an interesting third possibility: a system able to learn from its own experience.

Bottom-Up or Constructive Strategies

Function-Driven Strategies Bottom-up or constructive strategies build the description of an artifact through a sequence of decisions which occur at the lowest, most detailed level. The WRIGHT project demonstrates that a bottom-up approach can be function-driven (Baykan & Fox, 1991). WRIGHT solves layout problems by examining the constraints to be satisfied and builds a solution by satisfying one constraint at a time. This requires that all constraints be expressed uniformly as equations or inequalities in the form variables (in this case, the coordinates of the objects to be allocated) and that the structural alternatives that exist for solving any constraint be explicitly stored as a disjunct of constraints. The problem formulation generated by WRIGHT is equivalent to a mixed integer non-linear programming (MINLP) problem. This suggests that MINLP approaches generally can be subsumed under the present strategy, a connection that we cannot pursue further in the present paper.

The coupling between form and function variables is explicit and tighter than in any other strategy. One of the advantages of the approach is that interactions are taken into account automatically by the constraint satisfaction technique. A disadvantage is that it requires an explicit listing of all possible ways of satisfying the constraints in advance. Another problem is generic to the bottom-up strategies: the combinatorial explosion of the search space as the problems involve large numbers of objects or constraints.

Form-Driven Strategies Form-driven strategies include the classical hierarchical or incremental generate-and-test approaches as used by LOOS and other grammar-based systems. A prerequisite is that solutions can be constructed incrementally and that intermediate evaluations can be carried out with some certainty. Another, more fundamental, prerequisite is that the constructions guarantee that every promising solution can, in principle, be generated; this may require a considerable amount of theory formation. Since these mechanisms can be developed independent of the tests to be executed, the coupling between form and function variables can remain loose, which is one reason for the generality or flexibility of the approach. On the other hand, the search space tends to explode with the number of objects and forces functional concerns to be considered as early and effectively as possible. In (Flemming et al., 1992b), we present a more detailed comparison between WRIGHT and LOOS.

Middle-Out Strategies

Function-driven Strategies These strategies become possible when a highly-structured description of behavior is given that can be transformed into a logical or physical design by associating physical components with the elements in the behavior specification. "Highly-structured" means that the behavior specification contains an explicit representation of the interactions between behavioral components, which allows the transformations to be governed mainly by local constraints or specifications and eliminates, in the ideal case, the need for global critics. Both SAW and the Transformation Project use a graph representation: a Verilog description in the first case and a bond graph in the second case. This strategy depends for its success on two rather stringent prerequisites. An appropriate behavior description must be given at the outset. Equally important is the availability of behavior-preserving transformations, a decidedly non-trivial requirement as demonstrated by the Transformation Project.

Form-Driven Strategies Form-driven middle-out strategies are conceivable, albeit not represented by

any of the projects here. Instead of starting with a highly-structured behavior description, they start with a highly-structured *form* description. Many of the prototype refinement strategies discussed in the literature fit this description. They retrieve solution prototypes from a database and adjust them to a given context or problem specification by "parameter tweaking"; that is, the structure of the prototype is preserved in contrast to the refinement strategies that gradually expand a structure through levels of abstraction and the bottom-up approaches that construct it from its constituent elements. ARCHPLAN, a component of the IBDE system developed at the EDRC, implements aspects of this approach (Schmitt, 1988).

Challenges and Future Work

Aside from specific challenges faced by individual projects, several general challenges emerge from our survey. Foremost is the challenge to test a model's applicability for realistic problems. This is less a challenge for the models developed for digital electronics design, which have for the most part passed that test. But the majority of projects described in this paper are still in a highly experimental stage and must be tested in the context of more realistic problems. Another interesting test would be to apply particular models across domains,

These two types of tests or experiments are likely to increase our understanding of the salient characteristics of the underlying models and their relations to the characteristics of generic design problems. The overview presented in the preceding section should only be seen as a first step towards a more detailed and substantive classification.

Issues of usability and integration of the capabilities offered by successful models into general design-support environment will have to be addressed: What are the roles of the human and computational agents involved and how can they evolve through use and time? What are the desired types of interactions? Is there a generic set of interaction types, or are they idiosyncratic to a particular computational model, or will a mixture of both be needed?

References

Baykan, C. A. and Fox, M. S., "Constraint Satisfaction Techniques for Spatial Planning", in: *Intelligent CAD Systems III Practical Experience and Evaluation*, edited by P. ten Hagen and P. Veerkamp, Springer-Verlag, Berlin, 1991, pp. 187–204.

Birmingham, W. P., Gupta, A. P., and Siewiorek., D. P., *Automating the Design of Computer Systems – The MICON Project*, Jones and Bartlett, 1992.

Coyne, R. F. and Flemming, U., "Planning in Design Synthesis - Abstraction-Based LOOS", in: *Artificial Intelligence in Engineering V. Vol 1: Design*, edited by J. Gero, Springer, York, 1990, pp. 91–111.

Finger, S. and Rinderle, J. R., "Transforming Behavioral and Physical Representations of Mechanical Designs", in: *Proceedings of the First International Workshop on Formal Methods in Engineering Design, Manufacturing, and Assembly*, Colorado State University, January 15-17, 1990, pp. 133–151.

Flemming, U., Adams, J., Carlson, C., Coyne, R., Fenves, S., Finger, S., Ganeshan, R., Garrett, J., Gupta, A., Reich, Y., Siewiorek, D., Sturges, R., Thomas, D., and Woodbury, R., "Computational Models for Form-Function Synthesis in Engineering Design", Technical Report, no. EDRC 12-52-92, EDRC, Carnegie Mellon, Pittsburgh, PA, March 1992.

Flemming, U., Baykan, C., Coyne, R. F., and Fox, M., "Hierarchical Generate-and-Test vs. Constraint-Directed Search. A Comparison in the Context of Layout Synthesis", in: *Artificial Intelligence in Design'92*, edited by J. Gero, Kluwer Academic Publishers, 1992.

Ganeshan, R., Finger, S., and Garrett, J., "Representing and Reasoning with Design Intent", in: *Artificial Intelligence in Design'91*, Butterworth-Heinemann, 1991.

Garrett, J. H. and Fenves, S. J., "A Knowledge-Based Standards Processor for Structural Component Design", no. R-85-157, Department of Civil Engineering, Carnegie Mellon, Pittsburgh, PA, 1986.

Heisserman, J., *Generative Geometric Design and Boundary Solid Grammars*, Department of Architecture, Carnegie Mellon, Pittsburgh, PA, 1991.

Reich, Y., "The Acquisition and Utilization of Basic Aesthetic Criteria in Design", Technical Report, no. EDRC 12-39-91, EDRC, Carnegie Mellon, Pittsburgh, PA, 1991.

Rinderle, J. R., "Implications of Function-Form-Fabrication Relations on Design Decomposition Strategies", in: *Proc. Computers in Engineering*, ASME, Chicago, 1986, pp. 193–198.

Schmitt, G., "ARCHPLAN - An Architectural Planning Front End to Engineering Design Expert Systems", in: *Expert Systems for Engineering Design*, edited by M. D. Rychener, Academic Press, New York, 1988, pp. 257–278.

Sturges, R. H. and Kilani, M. I., "A Function Logic and Allocation Design Environment", in: *Proceedings for ESD Fourth Annual Expert Systems Conference and Exposition*, Detroit, April 3-5, 1990.

Thomas, D. E., Lagnese, E. D., Walker, R. A., Nestor, J. A., Rajan, J. V., and Blackburn, R. L., *Algorithmic and Register-Transfer Level Synthesis: The System Architect's Workbench*, Kluwer Academic Publishers, 1990.