Design as Building and Reusing Artifact Theories: Understanding and Supporting Growth of Design Knowledge

Jayachandra M. Reddy[1], Susan Finger[2], Suresh Konda[2], Eswaran Subrahmanian[2]

[1]Rockwell Science Center
Palo Alto, CA 94301 USA

[2]Engineering Design Research Center
Carnegie Mellon University
Pittsburgh, PA 15213 USA

Abstract

As artifacts are designed, knowledge is accumulated gradually and — as this knowledge is organized and reused — designs and design processes are continually refined. An understanding of the nature and growth of design knowledge and its reuse is essential for implementing better design systems and effective design practices. To develop such an understanding, we introduce *artifact theory* as an interdisciplinary theory about an artifact that is essential for designing that artifact. This theory encapsulates various types of synthetic, analytic and process knowledge and reconciles many disciplinary theories in the context of the artifact. We argue that it is necessarily a contextual theory and hence is ephemeral. While highly mature and well understood design domains may have complete artifact theories, in most domains artifact theories evolve during design. That is, designers not only produce a manufacturable description of the artifact, but also produce the corresponding artifact theory. We observe that this involves both adaptation and reuse of elements of existing artifact theories as well as development of new elements. Hence, we propose the view of design as building and reuse of artifact theories as the basis for understanding design and for developing design environments. We describe artifact theory in terms of several disparate views of design and bring them together leading to a unifying view. We discuss the implications of the view for computational design environments and outline our current research efforts in advancing and supporting this view.

1. Introduction

In designing an artifact, designers bring together knowledge from various sources for use in context as shown in Figure 1. The sources may include basic sciences such as physics and chemistry, engineering sciences such as thermodynamics and fluid mechanics, manufacturing and production sciences, empirical knowledge from handbooks and manuals, catalogs, previous designs, experience and creativity of designers, *etc*. A comprehensive design environment (computational or otherwise) should provide access to these various sources of knowledge, facilitate their use, help designers organize and reuse what they learn, and support sharing, collaboration and negotiation of design knowledge. The focus of this paper is on understanding the nature of design knowledge and its growth and the resulting implications for design environments.

The motivation for this work arises from a simple but important observation that engineering design not only involves knowledge use, but also knowledge building. This can be corroborated by the fact that even moderately complex design problems involve extensive argumentation and reconciliation, experimentation and understanding, development of new modeling assumptions and methods, and other activities typical of the task of knowledge building. While some elements of the knowledge required for design may remain invariant, often many elements are newly developed. However, most design environments are not evolutionary in that they do not support capture of newly generated knowledge and its reuse in future designs — at least not computationally. This paper presents a basis for understanding knowledge building during design processes and for developing design environments that can evolve with the accumulation of knowledge.

We introduce *artifact theory* as the encapsulation of knowledge about an artifact that forms the basis for understanding and supporting design knowledge and its growth. A theory is systematically organized knowledge applicable in a relatively wide variety of circumstances, especially a system of assumptions, accepted principles, and rules of procedure devised to analyze, predict, or otherwise explain

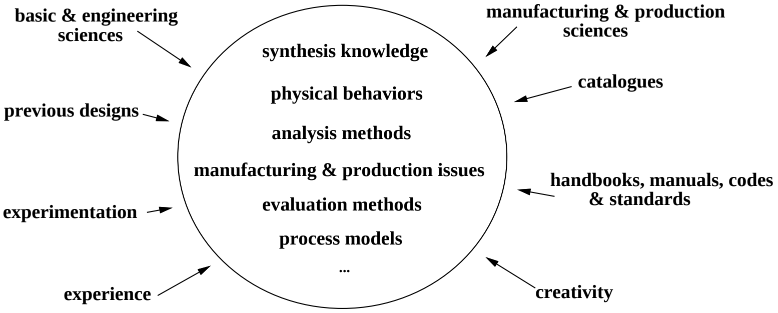

Figure 1: Designers use knowledge from many sources to design an artifact

the nature or behavior of a specified set of phenomena. In general, a theory should answer a variety of questions about the phenomenon under its purview. An artifact theory is a contextual theory that provides us with knowledge for designing and analyzing an artifact and for explaining and predicting the nature of the artifact. By contextual, we mean that the purview of an artifact theory is limited to a single artifact or a set of artifacts.

The outcome of a design process typically is viewed as a manufacturable description of the artifact consisting of detailed geometric models and drawings, specification of materials, list of parts and assembly specifications, *etc.* To reflect the knowledge building aspect of the design process, we extend this view and propose that design is a process of constructing a theory of the artifact, not merely constructing a manufacturable description (Figure 2).

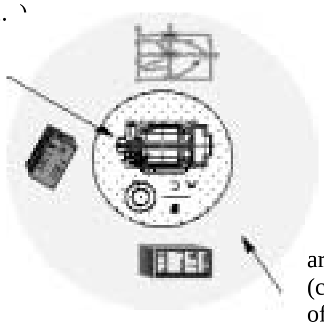

Figure 2: The outcome of design is an artifact theory which includes a manufacturable description

In general, artifact theory building involves both development of new elements of the theory as well as use and adaptation of elements of existing theories. This view of design as building and reuse of artifact theories and the shift towards expectation of artifact theories as the outcome of design provide several benefits to design practice. First, an artifact theory can explain the state of the artifact and can predict consequences of any changes. Second, artifact theories, or elements there of, can be reused or adapted for new designs. Despite this, there is a gap between support for theory building and for theory use. Many design systems take the view that design is mainly utilization of existing artifact theories and do not allow the building or evolution of theories. They ignore the activity of knowledge building, resulting in loss of the artifact theory that is so painstakingly developed during design. The view of design as building and reuse of artifact theories suggests that design environments should not only support use of existing theories, but should also support capture of new theories. With the addition of such theories, design environments should be constantly enriched and evolved.

The rest of the paper focusses on elaborating the need for contextual theories, advancing the view of design as building and reuse of artifact theories, and identifying its implications for design environments. We provide a philosophical and empirical basis for artifact theory and bring together several disparate views of design to establish a unifying theme. We then describe the nature of design process with respect to how knowledge is created, shared and reused in design, and identify its implications for computational environments. Finally, we raise some research questions and outline our current research efforts on elaborating and characterizing artifact theories, formalizing them in terms of languages, and supporting their development and use as a natural process of design.

2. Artifact Theory

What is a theory and what is its nature? We present a brief overview of scientific theories, especially with respect to their nature in design. There is abundant literature and debate on the definition, structure, and growth of scientific theories in the areas of philosophy of science and logic. Although some broad and general consensus exists, viewpoints differ. Readers interested in views presented by various schools of thought regarding this issue should refer to the critical introduction by Suppe [1].

2.1. Nature of Scientific Theories

Without delving into debates on the philosophy of science, we will discuss some essential functions and properties of scientific theories. Mehlberg summarizes empirical and theoretical aspects of scientific theories which we accept as the basis for discussion of theories [2]. He suggests that empirical aspects are determined, in substance, by a few essential functions discharged by any scientifically acceptable theory within the scientist's overall activity. He also argues that scientific theories could not discharge their essential functions without employing three formal components. These components provide a mechanism by which knowledge is put into storage and used at will, *i.e.*, they are the means by which a theory is formalized. The essential functions of a scientific theory are listed as below.

1. *Summarizing.* One of the principal functions of a theory is to summarize potentially large amount of information in a few condensed statements. For example, the theory of mechanics can be condensed into a single variational principle from which many mechanical laws can be derived.
2. *Predicting.* A theory should not only summarize established laws and facts, but also predict what may be established in the future. A theory must predict any fact which would be observed under any specifiable circumstances. For example, the theory of mechanics is expected to

predict mechanical laws which might be discovered.

3. *Explaining.* The essence of explanation consists in reducing a situation to elements with which we are so familiar that we accept them as matter of course, so our curiosity rests. A theory should also determine why a fact known to have taken place actually did so, why a law known to be valid in its proper realm of phenomena is actually valid there, *etc.*
4. *Controlling.* A theory should enable us to bring about desirable changes in our environment by following procedures which the theory indicates.
5. *Informing.* A scientific theory should provide us with relevant and dependable information about objects which are observable. The dependability of information is due to its being supported by the outcome of other investigations. In other words, scientific theories should provide us with empirical knowledge.

The following are the three formalisms by which a scientific theory discharges the essential functions listed above.

1. *Logical Formalism.* Using the logical formalism, the elements of science are built one upon another to generate new knowledge — the reverse of the process of explanation [3]. The axioms and definitions of a theory are manipulated using logical formalisms to generate theorems of the axiomatic system. Substitution of different logical formalisms in a theory may give different predictions or explanations.
2. *Mathematical Formalism.* Mathematical formalisms are essential to discharge empirical functions. Examples of mathematical formalisms are Hilbert spaces (in quantum mechanics) and linear algebra (in structural mechanics). The difference between logical formalism and mathematical formalism is that substitution of different mathematical formalisms in a theory does not affect the explanations or predictions of the theory.
3. *Metaphysical Formalism.* Scientific theories are based on some assumptions or axioms that are undecidable on logico-mathematical grounds and are not susceptible to any observational test. This class of axioms forms the metaphysical formalism of a theory. For example, in dynamic theory of gases, one of the axioms is that gases consist of rigid molecules that fly about in all directions, colliding with each other and with the containing wall.

While these empirical and theoretical aspects provide a basis for evaluating a theory, all generally accepted scientific theories do not discharge all the functions equally well and do not have all three theoretical formalisms. Typically, a theory will discharge some functions better than others.

2.2. Reconciling Theory and Practice in Design

Philosophy of science strives to understand the nature and structure of scientific theories, including their roles in the growth of scientific knowledge. The subject of philosophy of engineering, which is not as well established as philosophy of science, addresses similar issues in the context of engineering. It addresses questions such as what is the nature of theory in engineering design practice and what is engineering knowledge? Addis presents the views of various philosophers of engineering and weaves the views together to establish the nature and role of theory in design [3]. He addresses the question of whether the normally accepted classification of theory and practice is relevant in the context of design, specifically in structural engineering. Most engineers agree that a gap exists between theory and practice; however, Addis argues that there are several possibilities of gaps, as shown in Figure 3, and that it is often unclear which gap is under discussion.

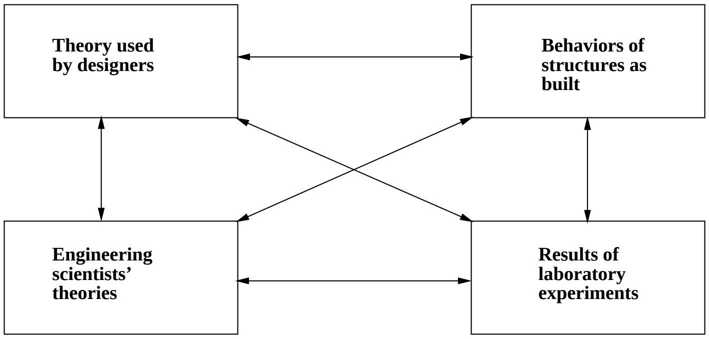

Figure 3: Gaps between theory and practice in design [3]

An engineering science is closer to a pure science such as physics, and its ultimate aim is to understand and explain the phenomena under its purview. Both engineering and pure sciences make use of theories for such explanation. On the other hand, engineering design is concerned with the production of artifacts in conditions less predictable and less under control. The aim of engineering design and engineering science are quite different. Engineering design uses what is typically called engineering knowledge, which may consist of empirical data, rules, laws, intuition, design procedures, experience and codes of practice. While engineering science and theories are useful, for example in establishing empirical rules and in analyzing the artifact, they are rarely directly useful in designing. Engineering knowledge does not fall easily under either category of theory or practice.

If we wish to bridge the gap between theory and practice, *i.e.*, make practice the application of theory, we must extend our definition of theory to include not just engineering science but all engineering knowledge. Addis argues that this extended definition of engineering theory discharges its functions in ways similar to scientific theory. The role of a theory in physical sciences is to explain certain phenomenon, whereas its role in design is to aid in the transformation of functional specifications to a solution that can be manufactured. Although the role of engineering knowledge in design is well recognized both in design research and practice (exemplified by the applications of knowledge-based systems), there have been relatively few efforts to formalize this knowledge in terms of theories.

2.3. Artifact Theory as a Contextual Theory

With this extended definition of theory for design, which includes engineering knowledge, let us ask the question: Is there a theory that is useful for the purpose of designing any given artifact independent of the context? Before answering this question, let us briefly examine the nature of artifact theories in terms of its empirical aspects. These include informational, explanatory, predictive, constructive (or controlling) and summarizing functions which are listed in Table 1. What elements should an artifact theory consist of to discharge these functions? The informational function demands that the theory must contain and deliver relevant information about the artifact. This information may include the structure of the artifact, its geometry, functions, and so on. The explanatory and predictive functions are similar and demand that the dependencies between information be captured in the theory (for example, what requirements influenced a particular design decision and how). The constructive function demands that the theory capture the knowledge required to design the artifact. This knowledge may include knowledge about analysis methods, synthesis methods, and evaluation methods. The summarizing function demands that the theory be compact and make explicit knowledge that can be reused across several different artifacts.

A general theory for design should encompass all the engineering knowledge, concisely or otherwise, and should be able to deliver its functions with respect to all artifacts. Ideally, a general theory should be able to transform given functional specifications into realizable physical attributes independent of the context, the same way that general theories in physical sciences (*e.g.*, general theory of relativity) explain much of the universe. However, no such well trenched theories exist for guiding the design process. Later in this paper, we describe some efforts in developing theories for design and analyze them with respect to the empirical and theoretical aspects of scientific theories.

In the absence of general theories for design, a pragmatic approach is to consider developing and using more contextual and specific theories for designing specific artifacts. An artifact theory is such a contextual theory that is essential to design and build a specific artifact or a class of artifacts. This notion of artifact theory can be exemplified by the existence of theories such as the theory of bridge construction and horology (the science of making time pieces). If we look at the history of the growth of theories in physical sciences, we notice that more unifying and more

Function	Elements	Examples (from motor design)
Informational	Information about the artifact: structure, functions, geometry, ...	• What is the composition of the cooling system? • What are its functional requirements?
Explanatory & predictive	Explanation for the state of the artifact Prediction of effects of any changes	• What requirements caused the selection of symmetric cooling flow? • What happens if requirements become more stringent?
Constructive	Knowledge about design methods, analysis methods, *etc.*	• How do I go about designing a motor with certain specs? • How do I simulate motor thermal characteristics? • How do I evaluate the performance of the motor? How do I benchmark it?
Summarizing	Patterns in the artifact information model	• What are possible configurations for a cooling system? • How do I adapt the motor thermal model?

Table 1: Elements of artifact theory and their functions

general theories came only after several disparate unconnected theories which were first developed to explain different physical phenomena. Therefore, starting with the development of contextual theories is a reasonable approach toward developing a more general understanding of design.

Contextual theories are developed by acquiring and systematizing knowledge in context. This may lead to the formation of layers of artifact theories reflecting various levels of contextualization. The theories at lower levels are formed by borrowing and adapting theories at higher layers and adding additional context dependent knowledge. Figure 4 shows the role of context in the organization of theories. While the most fundamental sciences, such as physics and chemistry, are constituents of all artifact theories, more contextual theories exist for specialized classes of artifacts, more specific theories, *e.g.*, the theory of design and construction of electric motors, are formed by borrowing and adapting general theories and adding additional context-dependent knowledge. Even more specialized theories may exist for a class of electric motors, like AC induction motors. The most specific theories are about individual instances of artifacts. Different artifact theories may overlap and contain several common elements that can be transferred and reused across artifacts.

3. Design Studies

To understand building and reuse of artifact theories, we have undertaken three participatory case studies. The first concerned the design of a third generation AC induction motor for hybrid electric vehicles. This case study was an experiment in distributed design between Carnegie Mellon University and Stanford University [4].

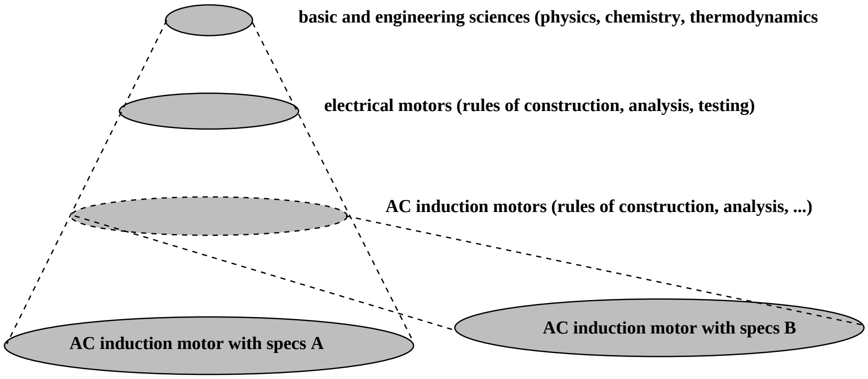

Figure 4: Hierarchy of artifact theories

Since two previous generations had been designed and manufactured, we could study the evolution of artifact theories over generations. The second case study concerned the design of a laser tracker which was part of the Madefast experiment [5]. The third project was the wearable computer design project at the Engineering Design Research Center at Carnegie Mellon [6, 7].

In the design projects studied, designers used many categories of knowledge including personal knowledge, knowledge from previous designs, knowledge about physical behaviors, and manufacturing knowledge. A detailed analysis of knowledge use in the motor design project is presented in [4]. One of the key observations regarding the design process relates to the large amount of information generated. For example, the documentation generated during the design of the AC induction motor is several hundred pages long. The final report describing design specifications, solutions, experiments, simulations, decisions and rationales is about 300 pages long. Since this design had two years of history, the previous documentation was used extensively. A significant portion of the design effort involved accessing and interpreting previous design information. The sought-after information was not only about final design drawings, but also about simulation models, underlying assumptions, experiments and their results, and manufacturing processes and constraints. Embedded within the information in design reports is knowledge that designers seek in designing other similar artifacts. A good design report is a compilation of the theory about the artifact (albeit non-computable); use of these reports exemplifies reuse of elements of existing artifact theories.

In these design case studies, designers did not have enough knowledge *a priori* to transform functional specifications into physical attributes. In practice, it is hard to implement the idealism of function to form mapping via behavior. A significant amount of contextual knowledge must almost always be invented. While this is clearly true for novel designs, the requirements for knowledge are unstable even in mature domains. This instability arises from the variety of requirements imposed by context, such as new customer requirements, new technologies, *etc*. For example, in a case study of the design of transformers in a large company, Finger *et al.* show that although design of transformers is over 100 years old, there is still significant novelty in the designs [8]. In related work, McMahon identifies several modes of incremental design including parameter space exploration, improved understanding of explicit-implicit attribute relationships, change in product design specification, modification of feasible design space, and change in the design principle [9]. Researchers working in the area of learning and design reuse have also looked at how knowledge from prior designs must be rationalized and made explicit [10, 11].

Designers must seek out knowledge in the context of the design problem, debate the relevance of this knowledge, and generate new knowledge. As suggested by Vincenti [12], such knowledge development may involve introducing blind variations, experimenting, and selectively keeping what works. Vincenti also states that variations are more likely to be tried out vicariously by analysis and experiment in place of direct trial in design of mature products. Another significant mechanism for the growth of engineering knowledge is the occurrence of design failures, when designer's conjectures are falsified [13]. Both Vicenti's and Petroski's models of knowledge growth were observed in the case studies described above.

Designers are aware of the instability of the design knowledge needed; Meyer, who developed a formal grammar for architectural and structural design of tall buildings, presents critiques of his prototype grammar by two domain experts [14]. The first expert criticizes the grammar because it will never be able to capture and formalize the knowledge behind design of tall buildings completely since the knowledge is always changing. In other words, artifact theories keep evolving.

> "It's a hopeless task. The minute that you've got it you'll want to do something else... The problem of designing a building is more like writing a poem than writing a sentence. The subtleties of that are so immense that you can't hope to duplicate it... The structural engineering of buildings, I think, is like being a builder of armature for sculpture. The stuff that's inside the Statue of Liberty is Eiffel's. To try to write a program that would anticipate every sculpture that a sculptor could come up with would be sort of a big waste of time, I think, although somebody might argue." [14]

However, the second expert realizes that it is desirable to formalize what we already know about the design process.

> "Other than these comments, I think you are on the right track. I like the idea of developing a grammar to establish the entire vocabulary, to establish a set of rules of how you can operate within the process." [14]

4. Disparate Views & Reconciliation

Several disparate views of design have been proposed in the past to guide design practice as well as development of computational environments. In this section, we briefly describe some of these views and identify common themes relating to artifact theory. The purpose of this section is to elaborate on artifact theory from the perspective of different views and bring them together, leading to a unifying theme of design as building and reuse of artifact theories.

General Design Theory (GDT). GDT [15] is a theory of design based on axiomatic set theory. It states that, given ideal knowledge, design is a mapping process from function space to attribute space with no substantial computation required. This theory, being too idealistic to model real-world design process, has been extended with the adaptation of ideal knowledge to real knowledge [16, 17]. Design with real knowledge requires the ability to continually model the designed artifact until it evolves into a set of candidates that satisfy the given specification. However, GDT, both in the ideal and real knowledge, only applies to domains with known topological structure. Thus, Reich hypothesizes that GDT may be most applicable in established design domains with well-developed categorizations and accumulated knowledge about how to mediate between the function and the attribute categorizations [18]. Designing in these domains may range from simple selection from a catalogue, to composing systems from available components, to a stepwise refinement process.

Our notion of artifact theory is similar to ideal and real knowledge in GDT. Ideal knowledge includes knowledge about topologies of functional spaces and attribute spaces as well as mappings between them. The real knowledge includes knowledge about how to mediate between functional and attribute spaces, *i.e.*, both to propose refinements as well as to evaluate them. GDT tells us that these are the elements of knowledge (*i.e.*, artifact theory) required for design. GDT assumes the existence of topological spaces prior to design, knowledge about mappings between them, and knowledge about how to mediate between functional and attribute spaces. In reality, complete knowledge rarely exists *a priori*, and often a significant part of the knowledge is invented during designing of an artifact. In terms of GDT, our view of design suggests the task of designers is to put together the ideal and real knowledge required for the development of the artifact. Through repeated design of similar artifacts, this knowledge can crystallize into ideal knowledge.

Design as a Computable Function. Fitzhorn proposes that the design process can be modeled using the abstract model of computability: the Turing machine [19]. In his model, designs are enumerated strings from a possibly multidimensional grammar, and design specifications or constraints are formal state changes that govern string enumeration. He defines a Turing machine in which the state transition functions are dynamic. This adaptation does not to alter any characteristics of general Turing machines. The design process is then mapped onto the dynamic-state Turing machine. The design process is a series of state transitions from initial specifications to final artifact description via transition functions, which are constraints on the design. The transition functions themselves are dynamic. The design includes three distinct, but interrelated elements: the design process, the artifacts of design, and a set of design constraints. The process is independent of the design context. The artifacts and constraints are domain dependent, and he assumes they can be formally defined.

In this model of design, the formal models of artifacts (strings) as well as transition functions on these models are context specific; these context-specific models and transition functions form the artifact theory. To use this model of design as the basis for implementing design systems, one must assume the availability of artifact theories, *i.e.*, formal models of artifacts and transition functions that transform these models. These assumptions typically do not hold true in reality, and development of these formal models and transition functions itself is a central part of the design process. As Fitzhorn points out, the exposition of design as a computable function tells us more about what cannot be done than what can be. However, it formalizes the knowledge required for designing facilitating better understanding of artifact theories.

Theory of Technical Systems. Hubka and Eder present a comprehensive and unifying theory to promote the understanding of technical systems [20]. They extend the definition of theory in the context of design in a manner similar to that discussed in Section 2.2. The term *technical system* means all human-made artifacts, including technical artifacts and processes. The primary aim of the theory is to classify and categorize knowledge about technical systems into an ordered set of statements about their nature, regularities of conformation, origination, development, and empirical observations. The content of the theory includes many aspects of technical systems: their constituents, structures, and the models that describe them; properties of technical systems and their inter-relationships; evaluation of technical systems from the viewpoints of engineers, users, and society; formal representation of technical systems (such as drawings); and historic evolution of technical systems. Hubka and Eder argue that the lack of a general meta-theory results in the unsystematic collection of know-how. The theory of technical systems is intended to be such a theory which brings together independent domains and generalizes them. Hubka and Eder distinguish between general theories and special theories of technical systems: many special theories provide concrete statements that are derived from the general theory. As shown in Figure 5, the specialized theories are enriched by cross-currents and cross fertilization from various types of machinery and from established scientific theories such as thermodynamics and fluid dynamics.

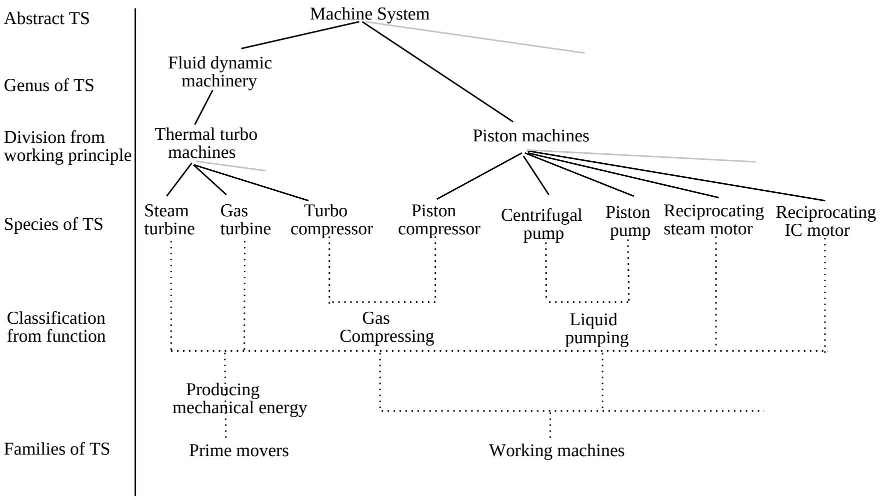

Figure 5: Formation of types of technical systems from [20]

In a way, a special theory of a technical system is equivalent to an artifact theory, both in its insistence on context and in its role. Hubka and Eder describe and characterize several elements of a general theory of technical systems, concrete elements of which are developed within the realms of specialized theories, in other words, artifact theories. However, they do not share our view of design as theory building; they propose the theory of technical systems as a means to collect and organize knowledge about engineering artifacts for use.

Shared Memory in Design. Konda *et al.* present a unifying theme for design theory emphasizing the importance of context [21]. They argue that a theory of the design process should be constructed that emphasizes the empirical and descriptive aspects of design, moving away from universal, context-less prescription. They present the concept of shared memory as the embodiment of both context and shared meaning and propose that the emphasis should be placed on developing shared memory. They distinguish between vertical shared memory and horizontal shared memory: vertical memory encapsulates increasingly detailed aspects of a given profession’s knowledge, whereas horizontal memory encapsulates the record of interdisciplinary communication. Horizontal shared memory always requires mutual translation of terms and concepts across groups, because members of design groups working on the same design do not share the same experiences, concepts, perspectives, exemplars, methods, or techniques. Creation of shared memory requires a deeper understanding of how individuals create shared meanings, what special languages, representations, models, *etc.*, are required to enable different individuals to share knowledge and experience. The authors also argue that design systems should not be limited to aiding an activity, but should also focus on creating communication channels to facilitate development of shared memory.

Artifact theory can be seen as a kind of shared memory because it embodies context and shared meaning among designers. Artifact theory is an encapsulation of knowledge about an artifact (or a class of artifacts), *i.e.*, it is tied to the artifact whereas the context of shared memory is unspecified — artifact theory, in a way, makes shared memory concrete.

Artifactual Engineering. Recently, Yoshikawa has proposed a new field of study called *artifactual engineering* [22]. The purpose of creating a new field is to bridge the gap between specialized engineering domains (*e.g.*, mechanical and electrical) that have evolved separately. Yoshikawa argues that we need to establish an engineering discipline that denies the existence of domains; he calls it *artifactual engineering*. He argues that as long as engineering is portioned into domains and depends on traditional territorialized principles, the goal of artifactual engineering will be to show us facets that are not normally visible.

> If we try to build up process theories in artifactual engineering now, we ought to learn from the history whereby structural dynamics was established by referring to knowledge needed in making bridges and building before structural dynamics came into being. The lesson is that it is necessary not only to produce a massive amount of products but also to consider industry, which has produced vast amounts of knowledge, an important source of wisdom. This knowledge has not been systematized, and formally it is the same type as the knowledge required for making a delicious omelet. If this can be called primitive knowledge, then in artifactual engineering it may be possible to attain research methods aiming to establish process theories, including general information about the degree to which systematic knowledge using it is organized into effective knowledge via abduction. [22]

Artifactual engineering shares concepts with theory of technical systems and shared memory in design. Although artifactual engineering denies the existence of specific domains (as currently established in engineering), its relation to the role of context is unclear. However, similar to our view of design, artifactual engineering emphasizes the study of artifacts and their design processes as opposed to specific engineering domains alone. In our view, generalization of artifact theories to form more general theories is the essence of artifactual engineering.

Axiomatic Design. Suh proposes two axioms for design, namely the functional independence axiom and the information minimization axiom [23]. These axioms can be considered to form the metaphysical basis for a theory of design. While this theory has a metaphysical formalism (*i.e.*, the axioms of design), it lacks logical and mathematical formalisms, *i.e.*, how to design from such axioms logically and algorithmically. While the axioms themselves are useful, further work must be done in developing logical and mathematical formalisms to complement them. These formalisms can be developed within the context of specific artifacts leading to development of contextual artifact theories based on Suh's axioms of design.

These contextual theories should interpret the axioms in the context of specific artifacts and provide design procedures based on them.

Programming as Theory Building. In the domain of software engineering, Naur argues that computer programming is equivalent to constructing a theory about the program [24]. Naur argues that programming is primarily building knowledge of a certain kind as an auxiliary product. This knowledge is generated, in part, through matching some significant part of an activity in the real world to the formal symbol manipulation done by a program running on a computer. Naur describes how programmer's knowledge is encapsulated into a theory. Having such a theory enables programmers to explain, justify, and answer queries about their program. This notion of "theory about the program" is similar to our notion of artifact theory.

5. Building, Sharing & Reuse of Artifact Theories

To support design as artifact theory building and use, we need to understand the nature of the design process in terms of how knowledge is created, shared, and used. In this section, we argue that the artifact theory is necessarily shared among the design participants and is developed collaboratively by the team. This has an important implication that design systems can support capturing of elements of theories by supporting and capturing systematic discourse. Based on this, we propose a model of design process that can be the basis for design environments.

5.1. Artifact Theory in Design Discourse

We assert that the artifact theory is shared among participants in the design discourse. While there may be some tacit elements in design knowledge, we believe that much of this knowledge can be expressed, captured, and communicated. Designers working on the same design must be able to express the theory of the artifact to one another. While design knowledge generated by a single designer may remain implicit, what needs to be communicated will be expressed and made explicit. This assertion can be verified by the content and quantity of information shared during the design discourse in the case studies described earlier.

This assertion can be verified by the content and quantity of information shared during the design discourse in the case studies described earlier. The theories developed and shared in discourse typically reside in various communication tools and are expressed in many forms. We define design discourse as the interactions between participants in the design — both human designers and computer tools. We assert that the artifact theory is shared among participants in the design discourse. Among these participants are clients, designers, manufacturers and suppliers. Human designers deal with ill-conditioned issues of the problem that are understood well enough to be incorporated in computer tools. Computer tools deal with issues that are well understood and can be formulated in specific computational methodologies.

Several researchers have explored various aspects of design discourse and placed discourse at the center of design process. Bucciarelli, based on ethnographic studies involving an X-ray system design, describes the style of design discourse and presents three illustrations: constraining discourse, naming discourse and decision discourse [25]. The first is about setting of performance specifications early in the design. The second is about naming, which is a design phenomenon that crystallizes images of parts and functions of the design in the minds of participants. The third is decision making. All three, *i.e.*, setting requirements, naming, and decision making, can be seen as the outcome of an argument. An example of a model of discourse is IBIS [26], in which designers propose, criticize, refine, abstract, and make concrete ideas and concepts that lead to a final product. Konda *et al.* argue that design is an activity in which designers move toward a shared understanding of the design artifact by negotiation and reconciliation of several different perspectives [21]. The view taken in this document is that designers construct a theory of the artifact collaboratively through negotiation and reconciliation of different perspectives and interests. From the case studies, the following properties of design discourse can be observed.

- *Information Content.* Large quantities of information are generated and communicated in the discourse among the design participants. In [4], we present gross quantitative measures of information generated and communicated during the AC induction motor design. We observe that the discourse changes as design moves from defining and finalizing specifications to detailing a manufacturable description of the artifact.
- *Tool Usage.* For reasons arising from the need for efficient information exchange and from the diversity of participants, design information is transmitted in a variety of representational forms [27]. Different media are suitable for different needs and representational forms, and hence different media are used for modeling and communication of design information.
- *Knowledge Intensiveness.* Design discourse is knowledge intensive, *i.e.*, much knowledge is exchanged in discourse. In previous studies, we have presented analyses of patterns in knowledge use in the evolution of design in the case studies and show that much of it is in deed embedded in the discourse [4, 28].

5.2. Model of the Design Process

In this section, we present a model of the design process that incorporates theory construction, theory sharing through discourse, and use of existing theories. This model, shown in Figure 6, is a basis for developing computational environments. The model consists of several participants including designers and computer tools, the artifact theory being constructed, a collection of existing artifact theories that are available for reuse, conceptual knowledge, and external knowledge. Designers' characteristics include competency in generating and evaluating design solutions, ability to acquire and assimilate conceptual knowledge, ability to communicate with other designers, and ability to use and build elements of artifact theories. Computer tools may be used in the design process to perform specific tasks, for example, a finite element analysis tool for thermal analysis. While these tools may be independent of the artifact, their use in the artifact design (usage models) is included in the artifact theory. These usage models may include analytical assumptions, assumptions about appropriate representations for the artifact, and so on. For example, when a general mathematical tool such as MATLAB is used to perform simulation, the modeling methods and assumptions underlying the dynamic model are a part of the artifact theory. Conceptual knowledge refers to the knowledge associated with individual designers, which is not codified and hence not accessible to others. External knowledge refers to any knowledge accessed and used by designers outside the design environment.

The process of theory construction involves several tasks, including reconciliation of different perspectives, conceptualization and generation of ideas, reuse of existing theories, and experimentation. This necessitates several interactions between the elements of the model shown in Figure 6. Reconciliation of perspectives requires that designers communicate with one another. This communication may require sharing the theory itself, represented by the interactions between the designers and the artifact theory. The interactions between the designers and the existing artifact theories reflects access and reuse of these theories.

Other interactions are shown with dashed arrows which may include interactions through synchronous communication such as face-to-face meetings, telephone and

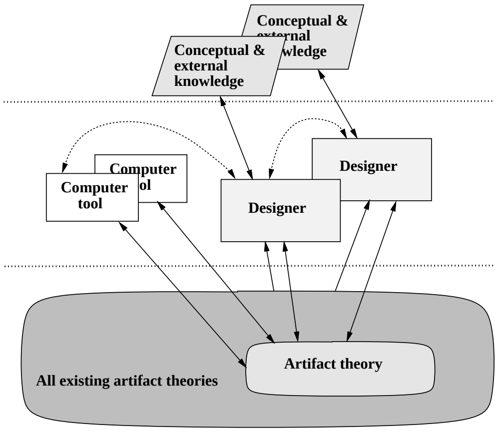

Figure 6: Model of design process

video conferences, application sharing via computers, and shared whiteboards. Asynchronous communication may be by e-mail and faxes. Other interactions included interactions between designers and tools and between designers and conceptual and external knowledge. Conceptualization of knowledge requires understanding of the context — both the context about the artifact and what is available in the world that is relevant. This requires that relevant information be available on as-needed basis. Recent advances in information technologies can benefit enormously in this regard — information on manufacturing processes, catalog parts, and so on can be made readily available.

6. Implications for Design Environments

Although design involves artifact theory building, the outcome of design is usually not seen as the theory, but rather only as a manufacturable description. Of the four elements of a theory listed in Table 1, only the first and part of the second are usually considered to be the outcome of design and most current design environments conform to this view. However, the foremost implication of the view of design as building and reuse of artifact theories is that design environments should support explication of artifact theories in computational forms as well as their use in future designs. In effect, they should bring together and integrate the theory building and theory use aspects of design. Whether or not support is provided for explicit theory building, theory building is inherent in the design process. Since these theories necessarily are shared among design participants, by supporting design discourse in a structured fashion, design environments can assure explicit artifact theories as the outcome of design.

In general, computational tools used in a design process can be categorized into two broad categories: those that primarily support use of existing knowledge and those that primarily support creation and sharing of new knowledge through discourse. The former category of tools have certain knowledge about the artifact and support a specific design activity. For example, in the motor design studied, a tool that performs thermal analysis of an AC induction motor has embedded in it knowledge about AC induction motors. The latter facilitate the design process, communication, argumentation, *etc.*. These support articulation and sharing of newly discovered knowledge, for example, word processors, e-mail, and documentation tools such as PENS [29] which provides limited support for structured modeling.

A gap exists between those tools that support theory use and those that support creation and sharing of theories (Figure 7). This is because the theories shared in the latter category of tools are not immediately visible and are not immediately incorporated into the former. A closer relationship between the two categories of tools will make the design processes more efficient and productive. Ideally, design environments should facilitate articulation of the knowledge developed during design in computable forms that is explicit and that can be readily used.

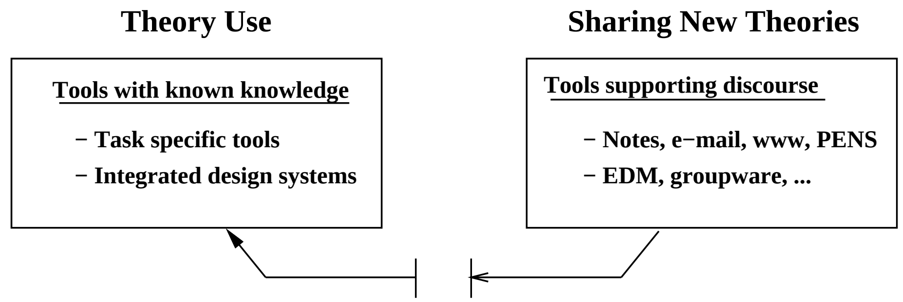

Figure 7: Gap between theory building and theory use

The view of design as building and reuse of artifact theories has some important implications for development of integrated design systems. Integrated design systems result from well developed and consolidated artifact theories. To integrate task-specific tools in the context of designing of an artifact, we need to know the interactions between the tasks, for example, the mappings between vocabularies of different tools. For example, in the AC induction motor design project, the tools used include AutoCAD for drafting and geometric design, MathCAD for thermal analysis model solving (resistance model) as well as for fluid dynamic analysis, Algor for finite element thermal analysis of end turns, and machining planning software tools at the job shops. The questions in integrating these tools include: do we have knowledge about interactions between these tools? If we know the interactions and incorporate them in an integrated design system, how stable are our assumptions about these interactions? The answers depend on the dynamic characteristics of the artifact design — how often the technologies change, how much and how often the context and requirements change, *etc*. The less dynamic the artifact design is, the more likely our assumptions about interactions remain valid, making integrated design systems possible. Unfortunately, many unpredictable interactions often occur between designers and disciplines, and therefore concurrent design systems cannot connect together discipline-specific tools in predefined ways [30].

Distributed agent architectures for concurrent engineering systems have been proposed, for example [31], where the focus is on integrating existing systems via shared models of knowledge. The claimed advantage of the agent-based model of collaboration is that computational resources can be utilized opportunistically and added incrementally [32]. The fundamental problem of information sharing and development of shared vocabularies (ontologies) still remains a bottleneck. It usually takes considerable effort to build such ontologies to make an agent a player in problem solving. Such ontologies have been developed and demonstrated in specific applications — however, the generality of such development processes and feasibility of the overall philosophy of plug and play is still questionable. Olsen *et al.* [32], based on experiences with agent-based architectures, conclude that information technology must address a basic need — the need for collaborators to establish information sharing agreements and to incorporate these agreements into the tools they use. These observations leave us with several questions: when is integration of tools possible and when not? It is conceivable that we can integrate tools at the level of sharing detailed geometric information, for example using PDES/STEP standards — but to what levels can we raise this integration? What aspects of design can be integrated and what cannot be?

7. Research Questions

In order to both practice and support design as building and reuse of artifact theories, several questions must be answered which include: What is the content of an artifact theory? Can we formally represent and capture it, and if so how? How can we transfer elements of artifact theories across artifacts? How can we support theory building and use in computational environments? In this section, we outline our current research efforts in answering these questions. A more complete description is given in [28].

What do designers look for in documents and reports concerning previous designs and what do they transfer across designs? In addition to looking for answers to specific questions, we believe they look for patterns in artifact information and transfer them across generations of artifacts and across different artifact types. These patterns are major constituents of artifact theories. The artifact information consists of information about composition of the artifact, *i.e.*, how various concepts or subsystems compose the artifact. For example, the motor designed is composed of a stator, a rotor, a specific type of cooling system, *etc*. Artifact information also consists of descriptor models that describe various aspects of the artifact and its subsystems or subconcepts. For example, the motor has a thermal model, an electromagnetic model, and a design process model, among others. The patterns both in artifact composition and in descriptor models of the artifact capture significant knowledge about the artifact. The patterns in artifact composition capture synthesis knowledge, *e.g.,* knowledge about what concepts can be combined well and how they can be combined. The patterns in descriptor models capture knowledge about specific aspects of the artifact. If these patterns can be made explicit during design, they become visible and can be reused in similar designs. Our assertion is that that design environments should support not only modeling of artifact information but also explication and reuse of patterns in artifact information.

To develop computational environments that support theory construction and use in terms of patterns, we need appropriate computational representations for patterns in the artifact information as well as for artifact information itself. We propose that the artifact information can be represented in terms of generalized graphs, and patterns can be represented in terms of grammars on these graphs. This is true for both patterns in concept composition as well as descriptor models. Posing the theory in terms of formal grammars is appropriate since grammars can capture knowledge effectively as corroborated by many knowledge-based applications in engineering design, for example [14]. Grammars also facilitate formal analysis and characterization of artifact theories and development of metrics for design such as complexity metrics and similarity metrics.

A computational environment can cohesively support theory building and use by providing support for explicating patterns in artifact information in the form of grammars and for using these patterns effectively in future design and modeling activities. In [28], we study theory construction and use in terms of grammars using an environment based on *n*-dim [33]. This environment supports concurrent development of models of artifact information as well as grammars underlying them. It also provides several features required for collaborative modeling including information management, access control, negotiation, *etc.* thereby facilitating the discourse through which artifact theory is shared and consolidated.

8. Conclusions

Universal theories for design do not exist; that is, we do not have theories that are all-encompassing and that provide support in designing any artifact of choice. In the absence of such universal theories, a pragmatic approach is to consider developing and using contextual theories for designing specific artifacts. We have introduced *artifact theory* as such a contextual theory. We have argued that design involves development of not only a manufacturable description of the artifact but also its theory. This theory building often involves both creation of new elements as well as use of elements of existing theories. This view of design — as building and reuse of artifact theories — provides a framework for understanding the nature and growth of design knowledge. This view, as a guiding principle for design practice, can benefit organizations significantly in building their respective corporate memories. However, in correspondence with this view, design environments should support structured discourse and structured modeling of design information making the underlying knowledge explicit leading to computable and reusable artifact theories.

Acknowledgements

This work is supported by the Engineering Design Research Center at Carnegie Mellon University, an Engineering Research Center of the U.S. National Science Foundation, under Grant No. EEC-8943164.

References

1. Suppe F 1977 *The Structure of Scientific Theories*. University of Illinois Press, Urbana IL
2. Mehlberg H 1962 The theoretical and emperical aspects of science. In: Nagel E, Supes P, Tarskied A (eds) *Proceedings of the 1960 Congress on Logic Methodology and Philosophy of Science*. Stanford University Press, Palo Alto CA, pp 275-284
3. Addis W 1990 *Structural Engineering: The Nature of Theory and Design*. Ellis Horwood Limited, West Sussex England
4. Reddy J M, Chan B, Finger S 1996 Patterns in design discourse: a case study. In: *Knowledge Intensive CAD, Volume 1*. Chapman & Hall, London, pp 265-283
5. Cutkosky M, Tenenbaum J, Glucksman J 1996 Madefast: an exercise in collaborative engineering over the internet. *Communications of the ACM* 39(9):78-87
6. Siewiorek D P, Smailagic A, Lee J CY, Adl-Tabatabai A R 1994 Interdisciplinary concurrent design methodology as applied to the navigator wearable computer system. *Journal of Computer and Software Engineering* 2(3): 259-292
7. Finger S, Stivoric J, Amon C. et al. 1996 Reflections on a concurrent design methodology: A case study in wearable computer design. *Computer-Aided Design* 28(5):393-404
8. Finger S, Gardner E, Subrahmanian E 1993 Design support systems for concurrent engineering: A case study in large power transformer design. In: *Proceedings of The International Conference on Engineering Design ICED'93*. The Hauge, pp 1433-1440
9. McMahon C 1994 Observations on modes of incremental change in design. *Journal of Engineering Design* 5(3):195-209
10. Duffy A H B, Kerr S M 1993 Customised perspectives of past designs from automated group rationalisations. *Artificial Intelligence in Engineering* 8:183-200
11. Duffy A H B, Duffy S M 1996 Learning for design reuse. *Artificial Intelligence for Engineering Design Analysis and Manufacturing* 10:139-142
12. Vincenti W 1990 *What Engineers Know and How They Know It*. John Hopkins University Press, Baltimore MD
13. Petroski H 1985 *To Engineer is Human: The Role of Failure in Successful Design*. St Martin's Press, New York
14. Meyer S A 1995 Description of the structural design of tall buildings through the grammar paradigm. PhD thesis, Carnegie Mellon University
15. Yoshikawa H 1981 General design theory and a CAD system. In: *Man-Machine Communication in CAD/CAM*. North Holland, Amsterdam, pp 35-58

16. Tomiyama T, Yoshikawa H 1986 *Extended General Design Theory*. Tech Report CS-R8604 Centrum voor Wiskunde en Informatica, Amsterdam

17. Tomiyama T, Yoshikawa H 1985 Extended general design theory. In *Design Theory in Computer-Aided Design* North Holland, Amsterdam, pp 95-130

18. Reich Y A 1995 Critical review of general design theory. *Research in Engineering Design* 7(1):1-18

19. Fitzhorn P A 1994 Engineering design is a computable function. *Artificial Intelligence in Engineering Design and Manufacturing* 8(1):35-44

20. Hubka V, Eder W E 1988 *Theory of Technical Systems: A Total Concept Theory for Engineering Design*. Springer-Verlag, New York

21. Monarch I A, Konda S L, Levy S N, Reich Y, Subrahmanian E, Ulrich C 1993 Shared memory in design: theory and practice. In: *Social Science Research Technical Systems and Cooperative Work*. Paris, pp 227-241

22. Yoshikawa H 1992 Proposal for artifactual engineering: aims to make science and technology self-conclusive. In: *Illume A Tepco* Semiannual Review Tokyo Electric Power Co Inc., Tokyo, pp 41-56

23. Suh N P 1988 *The Principles of Design*. Oxford University Press, Oxford

24. Naur P 1985 Programming as theory building. *Microprocessing and Microprogramming* 15:253-261

25. Bucciarelli L L 1988 An ethnographic perspective on engineering design. *Design Studies* 9(3):159-168

26. Rittel H, Webber M 1973 Dilemmas in a general theory of planning. *Policy Sciences* 4:155-169

27. Hubka V (ed) 1991 *Proceedings of ICED'91 International Conference on Engineering Design*. WDK, Zurich

28. Reddy J 1996 Building and reuse of artifact theories: A view of design and its implications for computational environments. PhD thesis, Carnegie Mellon University

29. Hong J ,Toye G, Leifer L 1995 PENS: personal electronic notebook with sharing. In: *Fourth IEEE Workshop on Enabling Technologies*. Berkeley Springs West Virginia

30. Finger S, Konda S, Subrahmanian E 1995 Concurrent design happens at the interfaces. *Artificial Intelligence for Engineering Design Analysis and Manufacturing* 9:89-99

31. Cutkosky M R, Engelmore R S, Fikes R E et al. 1993 PACT, an experiment in integrating concurrent engineering systems. *Computer* 26(1):28-37

32. Olsen G, Cutkosky M, Tenenbaum J M, Gruber T 1994 Collaborative engineering based on knowledge sharing agreements. *Proceedings of the ASME Database Symposium*. Minneapolis MN pp 11-14

33. Levy S, Subrahmanian E, Konda S, Coyne R, Westerberg A, Reich Y 1993 *An Overview of n-dim Environment*. EDRC Technical Report 05-65-93 Carnegie Mellon University

1. Suppe, F., *The Structure of Scientific Theories*, University of Illinois Press, Urbana, 1977.

2. Mehlberg, H., ‘‘The Theoretical and Emperical Aspects of Science,’’ *Proceedings of the 1960 Congress on Logic, Methodology and Philosophy of Science*, E. Nagel, P. Supes and A. Tarski,ed., Stanford University Press, Palo Alto, CA, 1962, pp. 275-284.

3. Addis, W., *Structural Engineering: The Nature of Theory and Design*, Ellis Horwood Limited, West Sussex, England, 1990.

4. Reddy, J. M., Chan, B. and Finger, S., *Patterns in Design Discourse: A Case Study*, Chapman & Hall, London, 1996, pp. 265-283.

5. Cutkosky, M., Tenenbaum, J. and Glicksman, J., ‘‘Madefast: An Exercise in Collaborative Engineering Over the Internet,’’ *Communications of the ACM*, Vol. 39, No. 9, 1996, pp. 78-87.

6. Siewiorek, D. P., Smailagic, A., Lee, J. C.Y. and Adl-Tabatabai, A. R., ‘‘Interdisciplinary Concurrent Design Methodology as Applied to the Navigator Wearable Computer System,’’ *Journal of Computer and Software Engineering*, Vol. 2, No. 3, 1994, pp. 259-292.

7. Finger, S., Stivoric, J., Amon, C., Gursoz, L., Prinz, F. B., Siewiorek, D., Smailagic, A. and Weiss, L., ‘‘Reflections on a Concurrent Design Methodology: A Case Study in Wearable Computer Design,’’ *Computer-Aided Design, special issue on Concurrent Design*, Vol. 28, No. 5, 1996, pp. 393-404.

8. Finger, S., Gardner, E. and Subrahmanian, E., ‘‘Design Support Systems for Concurrent Engineering: A Case Study in Large Power Transformer Design,’’ *Proceedings of The International Conference on Engineering Design, ICED’93*, The Hauge, August 1993, pp. 1433-1440.

9. McMahon, C., ‘‘Observations on Modes of Incremental Change in Design,’’ *Journal of Engineering Design*, Vol. 5, No. 3, 1994, pp. 195-209.

10. Duffy, A. H. B. and Kerr, S. M., ‘‘Customised Perspectives of Past Designs from Automated Group Rationalisations,’’ *Artificial Intelligence in Engineering*, Vol. 8, 1993, pp. 183-200.

11. Duffy, A. H. B. and Duffy, S. M., ‘‘Learning for Design Reuse,’’ *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, Vol. 10, 1996, pp. 139-142.

12. Vincenti, W., *What Engineers Know and How They Know It*, John Hopkins University Press, Baltimore, 1990.

13. Petroski, H., *To Engineer is Human: The Role of Failure in Successful Design*, St. Martin’s Press, New York, 1985.

14. Meyer, S., *A Description of the Structural Design of Tall Buildings Through the Grammar Paradigm*, PhD dissertation, Department of Civil Engineering, Carnegie Mellon University, Pittsburgh, PA, USA, May 1995.

15. Yoshikawa, H., ‘‘General Design Theory and a CAD System,’’ *Man-Machine Communication in CAD/CAM*, North Holland, Amsterdam, 1981, pp. 35-58.

16. Tomiyama, T. and Yoshikawa, H., ‘‘Extended General Design Theory,’’ Tech. report CS-R8604, Centrum voor Wiskunde en Informatica , January 1986.

17. Tomiyama, T. and Yoshikawa, H., ‘‘Extended General Design Theory,’’ *Design Theory in Computer-Aided Design, Proceedings of the IFIP WG 5.2 Working Conference 1985, Tokyo*, North Holland, Amsterdam, 1985, pp. 95-130.

18. Reich, Y., ‘‘A Critical Review of General Design Theory,’’ *Research in Engineering Design*, Vol. 7, No. 1, 1995, pp. 1-18.

19. Fitzhorn, P. A., ‘‘Engineering Design is a Computable Function,’’ *Artificial Intelligence in Engineering Design and Manufacturing*, Vol. 8, No. 1, 1994, pp. 35-44.

20. Hubka, V. and Eder, W. E., *Theory of Technical Systems: A Total Concept Theory for Engineering Design*, Springer-Verlag, New York, 1988.

21. Monarch, I. A., Konda, S. L., Levy, S. N., Reich, Y., Subrahmanian, E. and Ulrich, C., ‘‘Shared Memory in Design: Theory and Practice,’’ *Social Science Research, Technical Systems and Cooperative Work*, Paris, March 1993, pp. 227-241.

22. Yoshikawa, H., ‘‘Proposal for Artifactual Engineering: Aims to Make Science and Technology Self-Conclusive,’’ *ILLUME, A TEPCO Semiannual Review*, Tokyo Electric Power Co. Inc., Tokyo, 1992, pp. 41-56.

23. Suh, N. P., *The Principles of Design*, Oxford University Press, Oxford, UK, 1988.

24. Naur, P., ‘‘Programming as Theory Building,’’ *Microprocessing and Microprogramming*, Vol. 15, 1985, pp. 253-261.

25. Bucciarelli, L. L., ‘‘An Ethnographic Perspective on Engineering Design,’’ *Design Studies*, Vol. 9, No. 3, July 1988, pp. 159-168.

26. Rittel, H. and Webber, M., ‘‘Dilemmas in a General Theory of Planning,’’ *Policy Sciences*, No. 4, 1973, pp. 155-169.

27. *Proceedings of ICED’91*, International Conference on Engineering Design, Zurich, , August 27-29 1991.

28. Reddy, J., *Building and Reuse of Artifact Theories: A View of Design and its Implications for Computational Environments*, PhD dissertation, Civil & Environmental Engineering, Carnegie Mellon University, Pittsburgh, PA, June 1996.

29. Hong, J., Toye, G. and Leifer, L., ‘‘PENS: Personal Electronic Notebook with Sharing,’’ *Fourth IEEE Workshop on Enabling Technologies*, Berkeley

Springs, West Virginia, April 20-22 1995.

30. Finger, S., Konda, S. and Subrahmanian, E., ‘‘Concurrent Design Happens at the Interfaces,’’ *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, Vol. 9, 1995, pp. 89-99.

31. Cutkosky, M. R., Engelmore, R. S., Fikes, R. E., Genesreth, M. R., Gruber, T. R., Mark, W. S., Tenenbaum, J. M. and Weber, J. C., ‘‘PACT, an Experiment in Integrating Concurrent Engineering Systems,’’ *Computer*, Vol. 26, No. 1, 1993, pp. 28-37.

32. Olsen, G., Cutkosky, M., Tenenbaum, J. M. and Gruber, T., ‘‘Collaborative Engineering based on Knowledge Sharing Agreements,’’ *Proceedings of the ASME Database Symposium*, Minneapolis, MN, September 11-14 1994, pp. 11-14.

33. Levy, S., Subrahmanian, E., Konda, S., Coyne, R., Westerberg, A. and Reich Y., ‘‘An Overview of *n*-dim Environment,’’ EDRC Technical Report 05-65-93, Carnegie Mellon University , 1993.