CHAPTER 2

DESIGN AS SCIENTIFIC PROBLEM-SOLVING

Following Proclus' aphorism that "it is necessary to know beforehand what is sought," a ground rule of intellectual endeavor seems to be that any new field of study, to be recognized properly, must first scrutinize its bounds and objectives: where it stands in the universe and how it proposes to relate to the established disciplines. Such clarification is the object of this chapter.

In this chapter, we examine the logic and methodology of engineering design from the perspective of the philosophy of science. The fundamental characteristics of design problems and design processes are discussed and analyzed. These characteristics establish the framework within which different design paradigms are examined. Following the discussions on descriptive properties of design, and the prescriptive role of design paradigms, we advocate the plausible hypothesis that there is a direct resemblance between the structure of design processes and the problem solving of scientific communities. The scientific community metaphor has been useful in guiding the development of general purpose, highly effective, design process meta-tools [73].

2.1 INTRODUCTION

2.1.1 MOTIVATION AND OBJECTIVES

Design as problem solving is a natural and most ubiquitous of human activities. Design begins with the acknowledgment of needs and dissatisfaction with the current state of affairs and realization that some action must take place in order to solve the problem, so scientists have been designing and acting as designers (sometimes unconsciously) throughout their lives. As such, it is of central concern to all disciplines within the artificial sciences (engineering in the broad sense).

Design science is a collection of many different logically connected knowledge and disciplines. Although there is no single model that can furnish a perfect definition of the design process, design models provide us with the powerful tools to explain and understand the design process. Design has been discussed, among others, in contexts such as general design methodologies [105, 52, 108, 36, 6, 21, 22], design artifacts representation [30, 48, 94, 122, 92, 83], computational models for the design process [78, 84, 91, 96, 120, 71], knowledge-based CAD systems [32, 117, 97] and design theories [46, 112, 124, 72, 73, 13].

Our research in engineering design [13, 72, 73] has led us to believe that *evolution* is fundamental to design processes and their implementation by computer-aided design (CAD) and "expert" design systems in many domains. In spite of the disparity between the models, and regardless of whether one is designing computer software, bridges, manufacturing systems or mechanical fasteners, evolutionary speaking they are similar. As the design process develops, the designer modifies (due to bounded rationality) either the tentative (current) design, or the specifications - based on new information obtained in the current design cycle. The modification is performed in order to remove discrepancies, and eventually establish a fit between the two parts. The evolved information reflects the fundamental feature of bounded rationality. The new information determines the tentative design knowledge, stating the relation among high and low levels of design specifications. It also determines the inference rules (or inference mechanism) that specify the method for deriving new design specifications and/or design artifacts. Both the sets of design knowledge and inference rules reflect the beliefs, skills and expertise unconsciously developed by designers through the repetitive experiences. The converging design process includes a testing stage for verifying the tentative design against the tentative specifications to establish the direction of their future elaboration. This process terminates with an acceptable design. These characteristics were arrived at from arguments based on the concept of "bounded rationality" [106].

In this chapter, we present a largely philosophical discussion of our motivations. We focus our attention on how *scientific communities* solve problems. Our thesis is that design as an evolutionary problem solving activity conforms to the structure of problem solving of scientific communities. That scientific communities are successful at generating and deciding between alternative explanations for phenomena is indisputable. Scientific progress, looked at globally and with a time scale of many decades seems coherent and purposeful. At any one time many conflicting theories and paradigms may support to explain the same phenomenon. Scientific communities themselves can be the subject matter of scientific research. The nature of science has been a fertile topic in philosophy from the pre-Socratic through the present day. We are particularly indebted to a number of philosophers and historians of science of this century among them Popper's, Kuhn's, Laudan's and Lakatos [86, 57-60, 66, 61-63]. We hope to gain insight from this research that will be useful in guiding the development of general purpose, highly effective design process meta-tools [73].

2.1.2 OVERVIEW OF THE CHAPTER

Section 2.2 scrutinizes the bounds and objectives of design from the perspective of the design problem. The basic characteristics as articulated in this section are:

1. Generally, designers act and behave under conditions of bounded-rationality;
2. Alternatives, options and outcomes are usually not given in advance (ill-

structured problems), and must be found and developed by some research process;
3. Usually, the optimum decisions will not be sought and satisfying decisions will fully be accepted;
4. Computationally speaking, most design optimization problems (well-structured problems) are intractable. Hence, the optimal decisions will generally not be sought and satisfying decisions will fully be accepted.

As a result of these basic postulates, we argue in Section 2.3 that the design process can be viewed as a stepwise, iterative, evolutionary transformation process. These characteristics establish the framework within which different design paradigms are examined. Section 2.4 glean useful ideas from the metaphor of scientific research to define design paradigms. Having defined a design paradigm, we survey the contemporary design paradigms. All the paradigms share the characteristic of observed evolutionary phenomenon which occurs between the time when a problem is assigned to the designer and the time the design is passed on to the manufacturer.

Following our previous discussions on descriptive properties of design, especially the adaptive and evolutionary properties discussed in Section 2.3, and the prescriptive role of design paradigms (Section 2.4), we pose in Section 2.5 the hypothesis that there is a direct and striking resemblance between the structure of design processes and the structure of problem solving of scientific communities. The basic correspondence is summarized as follows:

1. The counterpart of the Kuhnian paradigm or Laudan's research tradition is the designer's knowledge-base needed to generate the set of design solutions;
2. The counterpart of a set of phenomena, events or problems are design problems that are entirely characterized by and generated as a result of measurable and non-measurable requirements (specifications);
3. The counterpart of a scientific theory (set of hypotheses) is the tentative design/form serving (much the same as scientific theories) as a vehicle for the designer to capture her thoughts;
4. Scientific discovery follows the hypothetico-deductive method, or the more justifiable procedure (following Popper) of conjecture and refutation. It is with a direct correspondence with the evolutionary nature of design processes;
5. Incremental redesign activity corresponds to the continual and incremental evolvement of scientific theories within a normal science, whereas innovative redesign activity corresponds to a transition to a new paradigm (conceptual or paradigm-shift).

Regardless of whether or not the scientific community metaphor serves as the bases for explanations for the evolutionary design process, it has also a heuristic value in explicitly carrying out the act of design. In Chapter 6, we develop a model of the process based on double interleaved activities of *analysis* and *synthesis*, which explode the specification world (the counterpart of scientific phenomena), and the design artifact (the counterpart of a scientific theory), until a successful solution is achieved. We illustrate the application of this evolutionary design model, among others, to the design of mechanical fasteners (Chapter 6), and gearbox (Chapter 17). Section 2.6 outlines a design methodology, based on the scientific community metaphor, by emphasizing the variational (or parametric) design part. Section 2.7 concludes the chapter.

2.2 PROPERTIES OF THE DESIGN PROBLEM

2.2.1 THE UBIQUITY OF DESIGN

The natural point to begin any discussion of design is to state succinctly in a single sentence what it is that one does when one designs and what the end product is. Such an endeavor has been attempted in a variety of contexts including architecture, engineering and computer science. Clearly, an over-simplified or single sentence definition of design will not do. One reason why definitions fail is the omnipresence of design or problem solving as a natural human activity [146]. We have been designing and acting as designers (sometimes unconsciously) throughout our lives. Designing is pervasive in many human activities, for example an engineer conceiving of a new type of toaster or configuring a manufacturing cell, a financial manager configuring a profitable portfolio, or a cook concocting a new pizza. Underlying these design tasks is a core set of principles, rules, laws and techniques that the designer uses for problem solving. According to common sense, design is the process of putting together or relating ideas and/or objects in order to create a whole which hopefully achieves a certain purpose [19]. Design, according to the Encyclopedia Britanica, "is a process of developing plans as schemes of actions; more particularly a design may be the developed plan or scheme, whether kept in mind or set forth as a drawing or model... Design in the fine arts is often considered to be the creative process per se, while in engineering, on the contrary, it may mean a concise record of embodiment of appropriate concepts and experiences. In architecture and product design the artistic and engineering aspects of design tend to merge; that is; an architect, craftsman, or graphic or industrial designer cannot design according to formulas alone, nor as a freely as can a painter, poet, or musician." In its effort to promote research in the field, the National Science Foundation defines design as "the process by which products, processes, and systems are created to perform desired functions, through specifications." These specifications include desired object features, functions, constraints, etc. Another broad definition is that design is any arrangement of the world that achieves a desired result for known reasons. The process of design itself involves some of the same constraints as diagnostic processes or planning processes. Design approaches have traditionally been subjective; that is, a standardized set of rules is not readily available which can be applied to all classes of design problems.

2.2.2 DESIGN AS A PURPOSEFUL ACTIVITY

Design begins with the acknowledgment of needs and dissatisfaction with the current state of affairs and realization that some action must take place in order to correct the problem. Most design theorists, including [105, 4, 67, 98], have derived a number of consequences of this ostensibly intuitive observation:

- There is a distinction between engineering science (the 'science of the artificial' as Simon coined) and natural science (e.g. physics, chemistry and biology) that can be expressed in a variety of ways. First, the aims and methodology of natural science and engineering differ. That is, natural science is concerned with 'analysis' and engineering with 'synthesis' [22]. Second, natural science is 'theory-oriented' while engineering is 'result-oriented'; and third, the engineering activity is creative, spontaneous and intuitive, while science is rational [146, 98];
- Design is a pragmatic discipline concerned with how things should be done. Thus, the design activity is influenced by the designer's world view and values. Consequently, the recognition and identification of the design problem, the nature of the design solution and the determination of valid research topics in engineering design, are all intimately a function of the designer's perspective [22].

2.2.3 DESIGN IS A TRANSFORMATION BETWEEN DESCRIPTIONS

Louis Kahn, the famous architect, viewed design as a process by which the transcendent forms of thinking and feeling produce the realization of form. By form, Kahn meant the essence created by a certain relationship of elements within the whole. Thus, in practical terms, a design problem is characterized in terms of a set of requirements (specifications, goals and constraints) such that if an artifact or system satisfies the requirements and is implemented according to the proposed design, the design problem will be solved [93, 76, 111].

2.2.4 CATEGORIES OF DESIGN REQUIREMENTS

The most basic type of requirement is empirical, measurable or well-defined in nature. A requirement is well-defined when it specifies externally observable or empirically determinable qualities for an artifact [22]. Some requirements can naturally be stated as empirical, which means that one knows precisely what procedures to construct or use in order to determine whether or not a given design meets such requirements. Design problems that are entirely characterized by such requirements fall within the category of what Simon [102] termed well-structured problems. The most important varieties of well-defined requirements are functionality, performance, reliability and modifiability [146]. Functional requirements refers to the capability of the designed artifact to do certain desirable things [22], that is, the minimum set of independent specifications that completely define the problem. Thus, the functional requirements are the non-negotiable characteristics of the desired solution. We distinguish between functionality and behavior as different levels of description, where the function of a piece of a system relates the behavior of that piece to the function of the system as a whole. Performance refers to the competence of the desired artifact to achieve its functionality well. In practical terms, it usually refers to economy in the use of some observable set of resources. Reliability of artifacts is defined as the probability that the artifacts will conform to their expected behavior throughout a given period of time [146]. Modifiability refers to the ease with which changes may be incorporated in the design of artifacts [22]. Modifiability requirements completely support the evolutionary characteristic of the design process, and the act of successive changes or improvements to previously implemented designs.

A design problem may also be generated as a result of requirements that are not measurable. Such requirements are termed as ill-defined requirements (conceptual), and any reasonably interesting and complex design problem will contain ill-defined requirements. A design problem produced fundamentally as a consequence of a set of ill-defined requirements is referred to as an ill-structured design problem [102]. The initial requirements may be neither precise nor complete. Hence, in order to show that a design solution satisfies a set of initial requirements, (including ill-defined objectives), all requirements must eventually be converted into well-defined requirements. The process by which this information is transformed into well-defined design objectives is called the design requirements extraction process. Hence, the extraction, elaboration or refinement of requirements is an inherent and integral part of the generation of design [22].

2.2.5 BOUNDED RATIONALITY AND IMPRECISENESS OF DESIGN PROBLEMS

Decision making during the design activity deals with highly complex situations. The traditional methods of decision-making are based on the classical model of pure rationality, which assumes full and exact knowledge about the decision situation being considered. In design, assumptions about the exact knowledge are almost never true. At least to a large measure, the requirements are not comparable and therefore, the preference ordering among them is incomplete. The departure from ‘pure-rationality’ based methods is needed in design because of the fact that the designer has a limited information-processing capacity and the information is vague. Generally, designers act and behave under conditions of ‘bounded-rationality’ [104, 106]. The concept of bounded rationality was developed by Simon in the context of administrative decision making [104], and subsequently elaborated inter alia to design decision-making. Such limitations may arise in several ways: the designer may not know all the alternative sequence of decisions; or even assuming all the conditions are known, the designer may be unable to decide the best sequence of decisions ; or finally, the time and cost of computing the best possible choices may be beyond the bounds of the available resources.

2.2.6 THE SATISFICING NATURE OF DESIGN PROBLEMS

The bounded rationality led Simon to postulate that, more often than not, the optimum decisions will not be sought and satisfying decisions will fully be accepted. That is, instead of requiring an optimal design, designers accept a "good" or "satisfactory" one. In Simon's terms this attitude toward design, which allows the use of heuristic methods, is called 'satisficing'. The postulate of satisfying decisions is related to the psychological theory of 'aspiration level' given in the classical work of [68]. Another related concept is 'incrementalism' given by [69]. 'Incrementalism' is also based on the limited information-processing capacity of the decision-makers (designers) which forces them to make decisions similar to those previously made.

2.2.7 THE INTRACTABILITY OF DESIGN PROBLEMS

Optimization theory is applied as a recognized technique that can assist designers in the decision-making process of design. Utilizing optimization theory to solve design problems poses optimization problems which demonstrate inherent intractability. Typical instances of design optimization problems include:

1. Design of mechanisms employs graph enumeration and graph isomorphism problems are known to be NP-complete;
2. Design of printed circuit boards (PCB) includes partitioning, placement and routing problems, which are known to be intractable. Such problems are referred to as NP-complete, or Non-deterministic Polynomial time Complete problems [31]. The CPU time required to solve an NP-complete problem, based on known algorithms, grows exponentially with the "size" of the problem. There exist no polynomial time transformations for NPC problems nor are there any polynomial time algorithms capable of solving any NP problems, therefore these problems are considered to be "open" or unsolved problems. The potential to solve these NP and NPC problems depends on the availability of certain heuristics. Hence, in spite of knowing that there does indeed exist an optimal solution to a design problem, the designer may still resort to satisficing methods.

2.2.8 THE FORM OF DESIGN

Designing an artifact can be considered a transition from concepts and ideas to concrete descriptions. By form (a synonym to design) we mean the essence or ultimate output of a design process created by a certain relationship of elements within a whole. For example, the form of a piston for a model aircraft engine, is a piece of short cylinder designed to fit closely and move inside another cylinder or tube. The piston consists of a cylinder, piston rod and pin. Despite whether it is made of plastic, iron or steel, it is recognized as a piston as long as the cylinder, piston, and pin remain in a certain relationship to one another.

The concept of form is elusive, abstract and complex. The design process involves conceiving of the concepts relevant to the form and the relationships between them, and representing the concepts using specific well-defined language. In the case of engineering design, such design descriptions range from specifications in formal language (such as computer-aided engineering systems, symbolic programming techniques associated with AI and hardware design/description languages) through description in quasi-formal notation (such as linguistic descriptions and qualitative influence graphs) to very informal and visual descriptions (such as functional block diagramming, flow-Diagrams and engineering drawings). The concepts underlying a design are captured in three views: The functional view describes the design's functions and processes, thus connecting its capabilities. This view also includes the inputs and outputs of the activities, i.e., the flow of information to and from the external activities. For example, in the design process of integrated circuits the functional level includes a register-transfer diagram. The behavioral view describes the design's behavior over time, the states and modes of the design, and the conditions and events that cause modes to change. It also deals with concurrency, synchronization and causality. Good examples are constraints that components must satisfy such as timing properties. The behavioral and functional views are invariant characteristics of the design or form. The structural view describes the subsystems and modules constituting the real system and the communication between them. It also captures geometrical information. While the two former views provide the conceptual model of the design, the structural view is considered to be a physical model, since it is concerned with the various aspects of the system's implementation. As a consequence, the conceptual model usually involves terms and notions borrowed from the problem domain, whereas the physical model draws more upon the solution domain. Examples include details about materials, layout, process parameters, heat conductivity and other physical parameters.

The design/form serves several distinct roles in the development of an artifact. First, a design/form constitutes a tangible representation of the artifact's conceptual and physical properties, and thus serves as a vehicle for the designer to visualize and organize thoughts. Second, it serves as a plan for implementation. To accomplish this, the design/form should contain a systematic representation of the functional relationships of the components. Such demarcation of form/design and implementation has not always been necessary [146]. Jones [52] and Ferguson [147, pp. 3-4] have mentioned that the artisans of the 18th and 19th century did not demarcate between conceptualizing an artifact and making it; and that the transition from "designing without drawings" to the engineer's way of "designing with drawings" is ascribed mainly to the increasing complexity of modern devices (such as an internal-combustion engine), and the need to enhance the interaction between the client who wanted a machine built and those who would build the machine [147]. Third, the design description must also serve as a document (for instance, in the form of user-manuals) that describe how to harness the final artifact by the user. Finally, the form/design serves as a vehicle for reflecting the evolutionary history that led to the emergence of the final form/design, thus facilitating the inspection, analysis and redesign (change) of the artifact [22].

2.3 PROPERTIES OF THE DESIGN PROCESS

2.3.1 SEQUENTIAL AND ITERATIVE NATURES OF DESIGN

Many design theorists argue that the design process can be viewed as a stepwise, iterative, evolutionary transformation process [105, 124, 112]. Consider the following two assertions (with justifications) regarding the nature of the typical design process:

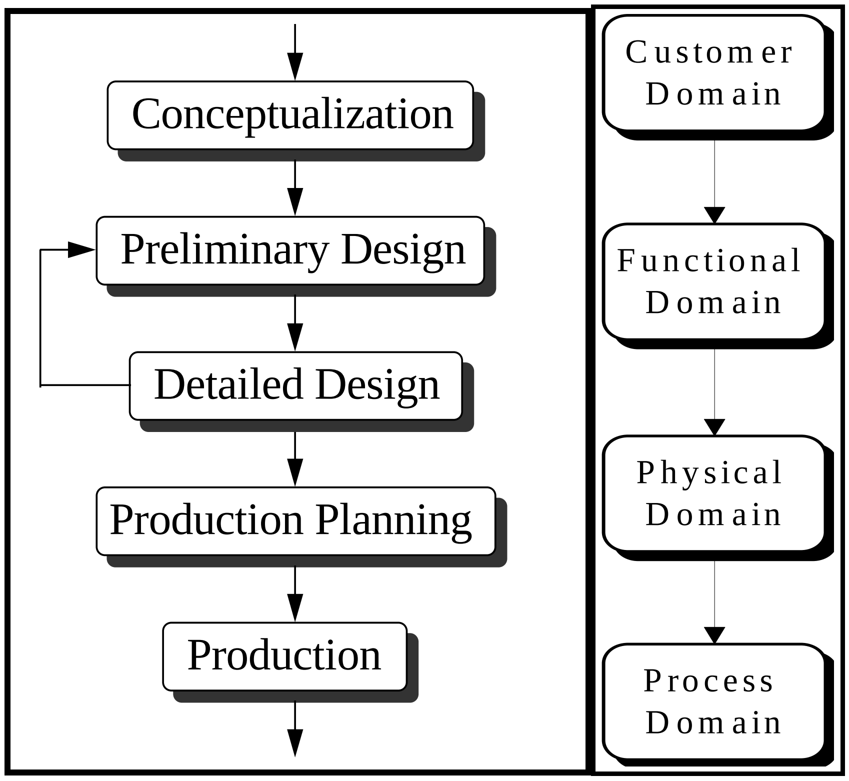
Assertion #1: Design is a sequential process

Almost every flowchart ever created that has attempted to describe the design process has shown evidence of the fact that design is a sequential process (see Figure 2.1). The design process evolves from concept through realization and it is impossible to go backwards. A part cannot be assembled until the components are machined; the components cannot be machined until the NC code is created; the NC code cannot be created without a dimensioned part model; the part model cannot be dimensioned without a set of requirements and a general notion of what the part looks like; and presumably the last two items come from a need that must first be identified. All this points to the seemingly undeniable truth that there is an inherent, sequential order to most design processes.

Assertion #2: Design is an iterative process

One can reason equally effectively, however, that design is an iterative process. First, designers are only human and have a bounded rationality. They cannot simultaneously consider every relevant aspect of any given design. As the design process progresses, new information, ideas, and technologies become available that require modifying the design. Second, design systems are limited; there is no known system that can directly input a set of requirements and yield the optimum design. Rather, the designer must iteratively break down the set of requirements into dimensions, constraints, and features and then test the resulting design to see if the remaining requirements were satisfied (see Figure 2.2). Finally, the real world often responds differently than is imagined. The real world is full of chaotic reactions that are only superficially modeled in any design system. All this points to the seemingly undeniable truth that there is an inherent, iterative nature to the design process.

In order to reconcile these two disparate visions of the design process, we categorize design iteration as occurring either *between* design stages (*inter-stage* iteration) or *within* a design stage (*intra-stage* iteration) and then create a new model of the design process combining both approaches to design (Figure 2.3). In this model, design still flows sequentially from initial concept through realization, each design stage providing the data and requirements for the subsequent stage. Within each design stage, however, the designer iteratively creates a design that meets the given requirements. This model largely represents the current state-of-the-art in CAD/CAM/CAE systems. While there are numerous software modules to assist the designer during intra-stage design iteration (e.g., QFD software to help identify customer needs and CAE software to analyze a current design), the tools are generally not well integrated at the inter-stage level.



A. [145] B. [144]

Figure 2.1 Traditional Views of Mechanical Design

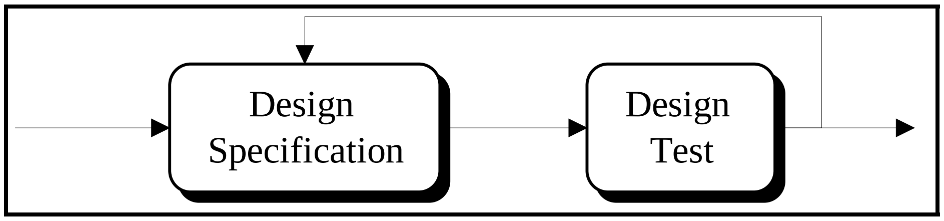


Figure 2.2 Specification and Test Iteration

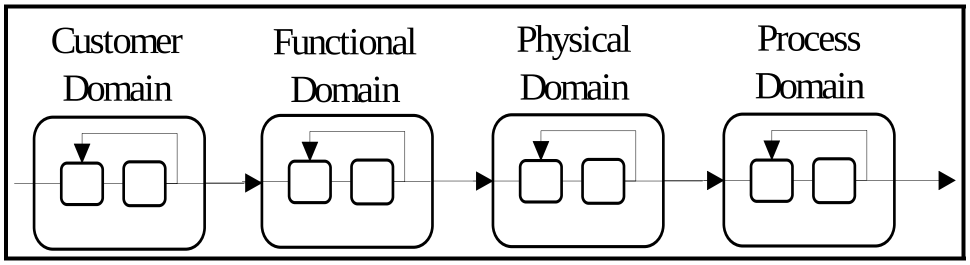


Figure 2.3 Combining Sequential and Iterative Design

2.3.2 THE EVOLUTIONARY NATURE OF THE DESIGN PROCESS

The concepts underlying the evolutionary characteristic of design are captured in three views: purposeful adaptation of artificial things, *ontogenetic*[1] design evolution and *phylogenetic*[2] design evolution (both latter phrases are borrowed from biology; [38] and [148]). Purposeful adaptation, according to Simon, can be thought of as an interface between the “inner” environment, the substance and organization of the artifact itself, and an “outer” environment, the surroundings in which it operates. If the inner environment is appropriate for the outer environment, or vice versa, the artifact will serve its intended purpose. For instance, a ship’s chronometer reacts to the pitching of the ship only in the negative sense of maintaining an invariant relation of the hands on its dial to the real time, independently of the ship’s motions. Regardless of whether or not the adaptation model is a universal feature of artificial systems, it also has a heuristic value. Hence, we can often predict behavior from knowledge of the artifact’s goals and its outer environment with only minimal assumptions about the inner environment.

Ontogenetic design evolution refers to the design processes that share the

[1] Ontogeny: The life history of an embryonic individual
[2] Phylogeny: The evolutionary history of a lineage

characteristic of observed evolutionary phenomenon which occurs between the time when a problem is assigned to the designer and the time the design is passed on to the manufacturer [22]. During this period the design evolves and changes from the initial form to the acceptable form. In this case, we say that there is a fit between the design and the requirements. The evolutionary model of design seems to support the cognitive model of design: Yoshikawa [124] argues that the ontogenetic design process can be decomposed into small design cycles. Each cycle has the following sub-processes:

1. Awareness - problem identification by comparing the object under consideration and the specifications;
2. Suggestion - suggesting the key concepts needed to solve the problem;
3. Development - developing alternatives from the key concepts by using design knowledge;
4. Testing - evaluating the alternatives in various ways such as structural computation, simulation of behavior, etc. If a problem is found as a result of testing, it also becomes a new problem to be solved in another design cycle;
5. Adaptation - selecting a candidate for adaptation and modification.

Protocol studies on how technically qualified people design were conducted by several researchers [e.g., 2, 37, 119, 53]. Subjects were given problems to solve in a specified amount of time and told to talk aloud while they were developing the design. Based on these studies, the researchers formulated several models of the design process. However, in spite of the disparity between the models, evolutionary speaking they are similar: as the design process develops, the designer modifies either the tentative design or requirements, based on new evidence (information) obtained in the current design cycle, so as to remove the discrepancy between them and establish a fit between the two parts.

Regardless of whether or not the evolutionary model is a universal feature of design processes, the adaptive model has also a heuristic value and serves a useful purpose in explicitly carrying out the act of design. Solving a problem by beginning with a set of goals, identifying subgoals which when achieved realize the goals, then further identifying sub-subgoals that entail the subgoals, and so on, goes by several names in the computer science, cognitive science and AI literature. Goal directed problem solving, stepwise refinement, and backward chaining are notable jargons used [3, 18, 79, 50]. One of the most celebrated of these ‘weak’ methods is means-ends analysis. This method was proposed by Newell, Simon and associates in the late 1950s and first used in the General Problem Solver (GPS) one of the earliest and most influential systems developed within the problem space/heuristic search paradigm. Means-ends analysis relies on the idea that in a particular task domain, differences between possible initial or ‘current’ and goal states can be identified and classified. Thus, for each type of difference, operators can be defined that can reduce the difference. Associated with each operator is also a precondition that the current state must satisfy in order for the operator to be applied. Means-ends analysis then attempts to reduce the difference between the current and goal states by applying the relevant operator. If, however, the preconditions for the operator are not satisfied, means-ends analysis is applied recursively to reduce the difference between the current state and the precondition.

Phylogenetic design evolution refers to the act of redesign, which is defined as the act of successive improvements or changes made to a previously implemented design. An existing design is modified to meet the required changes in the original requirements. A conventional instance of redesign is encountered in discussions of the history of electronic computers where it is convenient to refer to architectural families/computer generations. The members of the family/generation are related to one another through an ancestor/descendant relationship [8, 146]. In general, the concept of computer family/generation is tied directly to advances in technology. For example, vacuum tubes and germanium diodes characterize the first generation, discrete transistors the second and so forth.

The act of redesign can be illuminated and explained by considering two modes of evolution, namely incremental and innovative. The redesign activity may be defined as incremental if

1. Over a long period of time the overall artifact's concept has remained virtually constant;
2. Artifact improvements have occurred through incremental design at the subsystem and component levels and not at the overall system level. That is, there has been no major conceptual shift.

The automobile is an example of an incremental redesign related to an overall artifact's concept. The design team concerned with the next new car will take it for granted that there will be a wheel approximately at each corner and that, more or less, it will have the basic attributes of the Model 'T' [87]. Many other artifacts may be said to fall into the incremental redesign category, for instance, bicycles, tractors, ships and scissors.

Innovative redesign activity is concerned with innovative, novel conceptual design. Pugh and Smith [88] observed that in all probability, while many overall artifact's concepts are fixed, there is a tremendous opportunity for dynamism and innovation at the subsystems and components levels. For example, the differential gear is used in all cars today. There have been tremendous advances in gear technology, manufacturing processes and materials improvements but the concept is static. The innovative redesign activity is followed by incremental redesign activity. Notwithstanding, the limited slip differential is an innovation and improvement of the subsystems level - it is an innovative redesign activity. An innovative redesign is also encountered in the evolution of the ball valve. The first British patent was granted to Edward Chrimes in 1845. This artifact appears to have been conceptually static until the early 1970s with the introduction of the Torbeck valve, and later the Ve Cone valve. As another example, consider the evolution of bicycles which underwent at least seven stages of innovation and improvement of the subsystems level:

1. The pedal system was installed to replace footwork operation, enhance control of the wheels, and increase speed;

2. Incremental improvements in technology led to increasing the bicycle's speed;
3. The increase in speed created difficulties in stopping with feet. Thus, breaks were installed;
4. Wheel diameter was enlarged to increase speed;
5. The increase in wheel diameter led to instabilities in the bicycle. Thus, chain transmission systems were installed to increase speed and safety by lowering the need for larger wheel diameters;
6. Instabilities associated with increased speed and the beating of the wheels against the roads led to the emergence of tires;
7. In order to enable the rider to have greater control of the pedals, the Free Wheel system was instated which created a more dynamic connection between the pedals and wheels.

There are three additional points to note in this regard. Firstly, the artifacts in the phylogenetic design evolution are mature artifacts that either have been implemented or are operational. Secondly, the time lapse for the entire phylogenetic design evolution is measurable in terms of years (the first ball valve was introduced in 1845, while the first innovative emergence of the Torbeck valve was introduced only in the early 1970s) rather than days, weeks, or months as in the ontogenetic case. Finally, a single cycle of redesign will, in general, by itself constitute one or more cycles of ontogenetic evolution.

2.3.3 DESIGN PROCESS CATEGORIES

Sriram et al. [110] have classified the design process into four categories: creative design, innovative design, redesign and routine design. These classifications of design are process dependent and product independent. In creative design, the domain specific knowledge (e.g. heuristic, qualitative and quantitative) that is needed to generate the solution set and the set of explicit constraints (such as functionality, performance, environmental, manufacturability and resource constraints) may be partially specified, while the set of possible solutions, the set of transformation operators and the artifact space are unknown. Thus, the key element in this design activity is the transformation from the subconscious to the conscious. In innovative design, the decomposition of the problem is known, but the alternatives for each of its subparts do not exist and must be synthesized. Design might be an original or unique combination of existing components. Sriram et al. argue that a certain amount of creativity comes into view in the innovative design process [see also 120]. Redesign is defined as the act of successive changes or improvements to a previously implemented design. An existing design is modified to meet the required changes in the original requirements. In general, two scenarios may lead to the condition of redesign: first, when the design is passed on to the implementer, the artifact may fail to satisfy one or more critical requirements, and thus must be modified so that it satisfies the requirements. Second, the environment for which the artifact had been originally designed changes (e.g. in technology or other purposes for the artifact differ from those previously assumed) and produces new requirements. In routine design, the artifact's form, its method of design, and its mode of manufacture are known before the design process actually begins. It follows that an a priori plan of the solution exists and that the general nature of the requirements (satisfied by this design) is also a priorily known. The task of the designer is essentially to find the appropriate alternatives for each subpart that satisfies the given constraints [110, 14, 15]. Sriram et al. explain that at the creative end of the spectrum, the design process might be spontaneous, fuzzy, chaotic and imaginative. At the routine end of the spectrum, the design is predetermined, precise, crisp, systematic and mathematical.

2.3.4 THE DIAGONALIZED NATURE OF DESIGN

Newer design tools are beginning to affect the stepwise and iterative design process. The technology for both inter-stage and intra-stage categories of iterative design (see Figure 2.3) has become more available. Computer prices are constantly plummeting as their capabilities rise. Design software that used to require an expensive workstation can now run on a personal computer. Design software itself is becoming ever more capable. Higher-end design software packages (e.g., Pro/Engineer or I-DEAS Master Series) allow the designer to create a part and then are able to calculate the NC code to machine it and update the NC code when the part is modified (thereby iterating between the Physical Domain and the Process Domain stages). Recent CAE packages can analyze a part model, calculate key information about the part, and return the designer immediately back to where they were (thereby reducing the intra-stage iteration). Looking ahead, it becomes clear that the model just discussed is exactly backwards from the ideal. Inter-stage iteration is able to respond to conceptual changes and new information and should be fully allowed by the design system. Each inter-stage iteration, however, results in changes that must propagate through the design stages, requiring intra-stage iteration at each stage. As opposed to the aforementioned design model, the ideal design process will, instead, consist of maximizing the inter-stage design iteration and minimizing the intra-stage design iteration.

Maximizing Inter-Stage Design Iteration

In a design model with no inter-stage iteration, design insights are always limited to the current design stage. Because of the inherent iterative nature of design, there has always been a need for design systems that support inter-stage iteration. Recently, however, there have been even more demands made for inter-stage iteration in the form of incremental design.

We are part of a global competitive marketplace that is becoming more global and more competitive every day. Incremental design has become the standard approach towards design in many areas. Most new products are only slightly modified from their predecessors with slight cosmetic or feature enhancements. In order to decide which new products to develop, large consumer goods companies often create several different prototypes, test market all of them, and develop whichever one sells the best. In this ever-changing environment, fast time-to-market has become critically important. Companies cannot be required to completely redesign a product simply to add new features or modify the specifications or incorporate new materials or new technologies. Computer companies cannot afford to redesign their computer just because a new CPU is introduced. Most of the design specifications do not change. Likewise, in designing a new computer keyboard, many design issues have already been decided including which keys to include and in what order to place them.

Rapid prototyping involves visiting each design stage quickly, in an effort to rapidly create a final product. Changes then are made to the product at each design stage. Rapid prototyping demands productive incremental design. Productive incremental design demands smooth inter-stage iteration. Incremental design begins with a completed design and iterates back to a previous design stage to effect changes on the design. This concept, however, is the antithesis of the sequential design model.

The problem then becomes how to model inter-stage design iteration while acknowledging the sequential nature of design. Towards this goal, we have created the *diagonalized design* paradigm. Diagonalized design reflects the reality that the designer has a bounded rationality and that new information is constantly being gathered during the design process, not simply before each design stage. For example, consider the diagonalized view of mechanical design (see Figure 2.4). Design still progresses from concept through realization, but the designer can incrementally modify the design in any previously defined design stage and the design is automatically updated in all the later design stages.

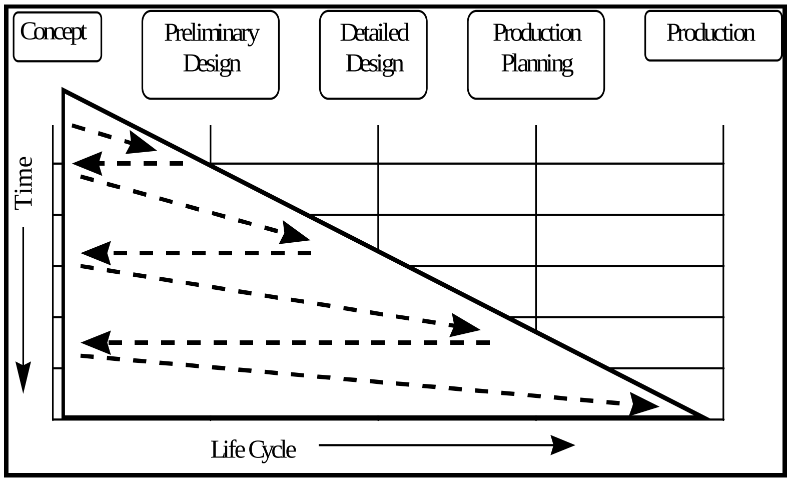


Figure 2.4 Perfectly Flexible Mechanical Design Process

While perfect flexibility in the design process is the goal, the bottom line can be adjusted to reflect more realistic current conditions. The area of the enclosed trapezoid then, becomes a measure of the flexibility of the design process. The flexibility is very dependent upon local conditions. Specific software, available technologies, corporate policies, or other factors greatly affect the flexibility of the design process. These different ranges of flexibility can be shown using diagonalized design. For example, Figure 2.5 represents a limited flexibility, where iteration is restricted to recent design stages; Figure 2.6 shows an inflexible design process where no iteration is allowed; finally, Figure 2.7 demonstrates a design process that is very flexible in the beginning stages of design, but becomes less so as the design moves towards production. By tilting the bottom line the other way, the opposite condition could be shown where beginning stages of design are inflexible, but production is highly flexible.

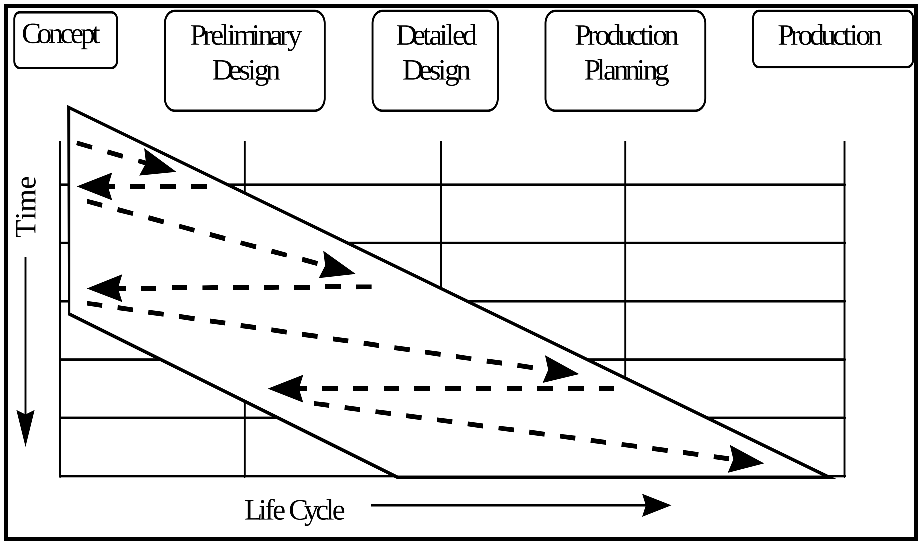


Figure 2.5 Limited Flexibility Mechanical Design Process

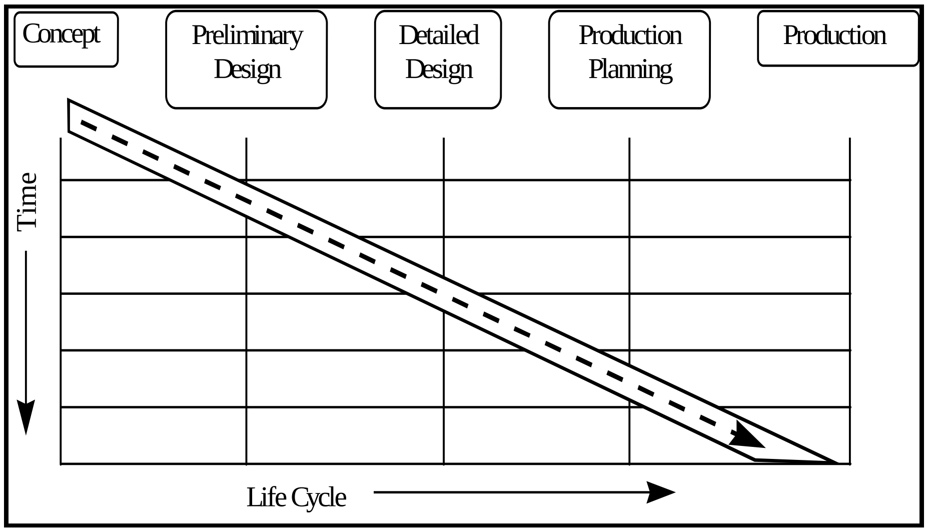


Figure 2.6 Inflexible Mechanical Design Process

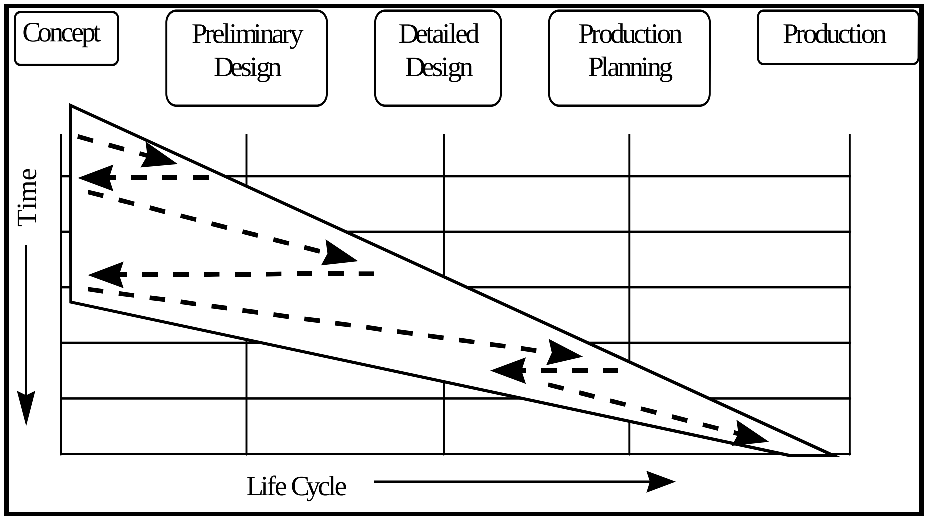


Figure 2.7 Conceptually Biased Flexibility

Design systems capable of fully iterative design will have to support iteration between each set of two sequential design stages:

1. Customer Domain to Function Domain Iteration

2. Function Domain to Product Domain Iteration
3. Product Domain to Process Domain Iteration

The difficulty of implementing iteration in a design system varies both with the level of the iteration as well as the generality of the system.

It is much harder to generalize the design process in earlier iteration levels. Iterating back to the conceptual level of design requires some form of parameterization of the design space. This is certainly possible for very domain-specific design systems but as the generality of the design system increases, however, the general nature of design becomes harder to implement. As a result, there are no known products which adequately address the first level of iteration. There has been much success, however, in generalizing the design process during later stages, both in modeling essentially any type of part and in calculating the NC code to machine arbitrary surfaces. The last level of iteration, however, is highly dependent on the manufacturing process. In one company, this may simply require reprogramming one or more robotic manipulators (the reprogramming could be automated). In this case, smooth iteration would be possible. In another company a product may depend on a highly capital intensive, inflexible design process (e.g., injection molding) and then it becomes harder and more expensive to incorporate design changes. This final stage of iteration may often be difficult to implement, but its implementation is still typically easier to understand than that of earlier iteration levels.

Minimizing Intra-Stage Design Iteration

During intra-stage iteration, the user is simply trying to meet the requirements that were input into the design stage. There are several approaches that could be taken towards minimizing intra-stage design iteration including:

1. Incorporating New Information
2. Modeling Real-World Interactions
3. Allowing Realistic Design Constraints

The three options are addressed through chapters 6, 13, and 14. To illustrate the limitations with how current design systems utilize design constraints, consider the Design-Analysis loop presented in Figure 2.2 as applied to the Detailed Design stage of mechanical design. A part is defined in terms of its dimensions and in a constraint-based design system, the designer enters constraints that the design system satisfies by adjusting the values of the dimensions. During the Design stage, the designer fully defines the dimensions of the design. In the Analysis stage, the designer calculates the values of other desired attributes. Clearly, the only need for the Analysis stage is to calculate whatever attributes cannot be constrained. Furthermore, the fact that the designer analyzes the design indicates that there are degrees of freedom in the design that were artificially constrained in order to analyze the part. Therefore, it can be summarized that a designer is forced to constrain attributes of the design they do not care about so they can calculate those attributes of the design they do care about. They then incrementally modify the unimportant attributes until the important attributes have achieved their proper values.

2.4 SURVEY OF DESIGN PARADIGMS

2.4.1 DEFINING A DESIGN PARADIGM

According to the dictionary, a paradigm is “a model, a pattern, or a standard.” In [22, 146], it was pointed out that a design discipline may comprise several alternative paradigms at any given time. For instance, when we refer to the process of designing finite-state dynamic systems based on four paradigms: finite-memory machine, Moore machine, Mealy machine and combined machine. All of these paradigms are based on the assumption that one subsystem of the designed structure system is a temporary storage of states of some variables, while the remaining subsystems represent function dependencies among appropriate variables. The paradigms differ in the nature of the function dependencies, which affects the constraints imposed upon the structure of the system to be designed. Another example of the role of paradigms is the notion of functions as building blocks for computer programs which form the basis for the development of a distinct style of programming called functional programming [43].

The common notion of a paradigm was enriched by Thomas Kuhn’s seminal treatise on the nature of the genesis and development of scientific disciplines [57, 58, 59, 60, 75]. The concept of a design paradigm is best elucidated by the Kuhnian paradigm concept as will be illustrated in this section. To Kuhn, a paradigm in its essence comprises of a *Disciplinary Matrix*. A disciplinary matrix refers to a network of theories, techniques, beliefs, values, etc. that are shared by, and generally agreed upon a given scientific community. The following components are identified within a disciplinary matrix [58, 60]:

1. Symbolic Generalizations, examples include Newton’s laws of motions and Ohm’s laws in electricity;
2. Beliefs (or Commitment) in metaphysical and heuristic models, such as the belief that the structure of an atom resembles a tiny planetary system [44], or that logical languages are the most effective medium for expressing the declarative knowledge in artificial intelligence systems [81];
3. Values, for example the desire for a simple theory or solution as exemplified in the principle known as Occam’s Razor;
4. Exemplars or Shared Examples, which are defined as the concrete problem-solution networks encountered by students of scientific disciplines in the course of their training, education and research apprenticeship (through the solving of textbook exercises, exams, and laboratory experiments) and by scientific practitioners during their independent research careers.

A particular set of assumptions upon which several different design methods may be based, is often referred to as a methodological design paradigm. A methodological design paradigm may be, like models within a Kuhnian paradigm, metaphysical in origin, or purely heuristic in nature. Similar to the role of a Kuhnian paradigm in the context of scientific discovery, a design paradigm serves as a framework or starting point for the solution of design problems. It is, thus, fundamentally an abstract prescriptive model of the design process that serves as a useful scheme for constructing practical design methods, procedures, and (computational) tools for conducting design [146]. A design method does not constitute a design paradigm. By definition, methods based upon the same paradigm are equivalent in the sense that they share the same set of possible solutions. This set consists of all solutions to the problem except those that violate any of the assumptions that constitute the paradigm. Hence, a given design method may be regarded as concrete and a practical embodiment of a design paradigm; it is an explicitly prescribed set of rules which can be followed by the designer in order to produce a design. A paradigm, according to the definition, will provide a framework or scheme for one or more design methods, just as it may serve as a framework or scheme for descriptive and automated tools.

Various schools of thought may become associated with a design paradigm. For example, in hardware logic design ('gate-level' design), the so called 'Eastern School' (a 'Naturalist' methodology) favored the use of block diagram in designing basic circuits, while the 'Western' School (a 'Formalistic' methodology) advocated the use of Boolean algebra [146]. Another example, in structural engineering of bridge design, concerns the debate between advocates of mathematical analysis of structural forces as subordinate to the development of structural form, and the approach that sophisticated analysis of the structural forces has priority over (and is determined by) structural form [10, 146].

2.4.2 DESIGN PARADIGMS

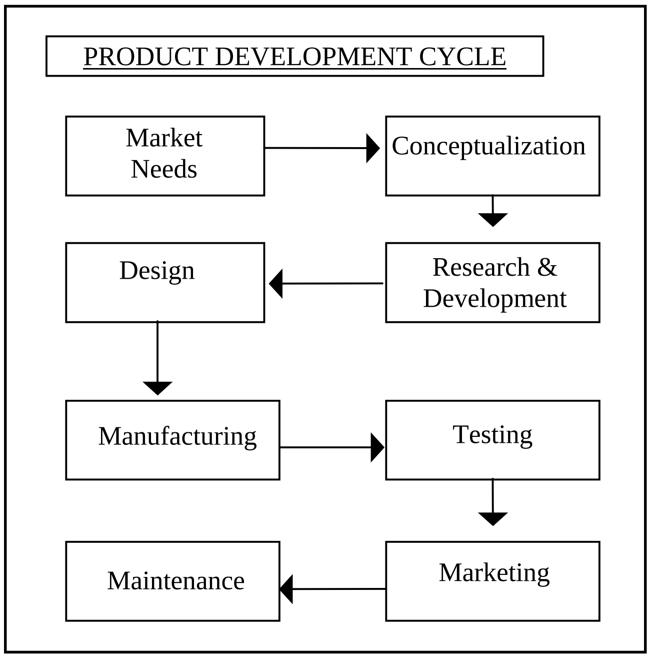
There are two major approaches to increasing our understanding of design disciplines that lack sound scientific theories; case studies (the counterpart of exemplars or shared examples) and models (the counterpart of a disciplinary matrix). The case studies approach was prevalent in such disciplines as psychology, prior to the establishment of the experimental method. This technique is also predominant in engineering design which relies mostly on the situation interpretation. The second approach is to use a model to define and understand the design process. Various perspectives and models need to be considered in order to gain a better understanding of the design process. Although there is no single model that can furnish a perfect definition of the design process, models provide us with powerful tools to explain and understand the design process. Models can be classified into five major types of paradigms: Analysis-Synthesis-Evaluation (ASE), Case-Based, Cognitive, Algorithmic and Artificial Intelligence. Following is a review of each of these paradigms. The interested reader may refer to Dasgupta [146] for discussions of these issues.

2.4.3 THE ANALYSIS-SYNTHESIS-EVALUATION (ASE) DESIGN PARADIGM

The ASE design paradigm is a very widely believed paradigm in the engineering discipline. Three basic phases of design described by [20, 7, 51, 70] are analysis, synthesis and evaluation. Analysis is concerned with defining and understanding what must be translated by the designer to an explicit statement of functional requirements (goals). Synthesis is involved with finding the solutions among the feasible alternatives. Evaluation is concerned with assessing the validity of the solutions relative to the original functional requirements [20]. In general, several instances of these three phases may be required in order to progress from a more abstract level to a more concrete level in the design process. A general model of design can be visualized as a feedback loop of synthesis, analysis and evaluation. The ASE model of design process is inherently iterative; the designer repeatedly goes back to refine and improve the design until it satisfies the requirements. Analysis and synthesis are on the forward path of the design loop, while the evaluation process is on the backward path, verifying the synthesized solutions [99]. A cycle is iterated so that the solution is revised and improved by reexamining the analysis. It has been argued that these three phases of the design, which are imperative for any design, irrespective of domain, form a framework for planning and organizing design activity.

Figure 2.8 depicts a more comprehensive version of a commonly used model of product development and design process. The design activity is viewed as part of the total product development process. Engineering a product involves several stages [109]: The first stage involves a market survey for potential products. This is followed by the conceptualization stage, where a product is conceived either as the result of a need or motivated by a potential profit. In the research and development stage, the information needed for the design of the product is developed. The design stage involves configuring the product based on several constraints. This is followed by the manufacturing process which yields the actual product (often preceded by developing a prototype). The product is then tested for quality in the testing stage and marketed in the marketing stage. The maintenance stage of the product is provided as a service by most organizations. The above process is iterative; for example, problems may arise during manufacturing and the product may have to be redesigned.

The process of solving a typical design problem involves various stages: problem identification, analysis, decomposition, synthesis, testing, evaluation and detailed design. The first task is concerned with identifying the problem (often fuzzy in nature), resource limitation, target technology, etc. Analysis involves listing the requirements and performance specifications, as well as specifying the constraints and objectives. Synthesis is the process of selecting components to form a system that meets design objectives while satisfying constraints that govern the selection. The components may themselves be complex entities which need to be synthesized first. Decomposition is often resorted to as a means for synthesizing the artifact into smaller and smaller components [17]. The artifact is decomposed into a hierarchical assembly of systems and subsystems until terminating in a functional or physical attribute. In such a case, components, individually and through interaction with each other, meet the design goals. The testing stage involves the response of the system to external effects. This is determined by using an appropriate model of the system (such as stress and thermodynamic analysis) to check the feasibility of a design. Evaluation of such a design involves critiquing the solutions relative to the goals and selecting among alternatives. Traditionally, evaluation criteria have been represented either as production rules in production rule-based systems or as constraint rules in blackboard-based systems [27]. Other measures of performance for engineering design were constructed with the help of the theory of fuzzy sets [23, 26, 123]. The scope of these evaluators has been restricted to testing the validity of a solution rather than its degree of acceptability. In order to achieve these goals, design critiquing which involves evaluating a design in terms of its effectiveness in satisfying a set of design objectives and constraints, was recently proposed [90, 56]. Detailed design involves the determination and evaluation of several preliminary geometrical layouts of designs. Various components of the design are refined so that all applicable specifications are satisfied. All seven stages are inextricably intertwined and are not distinct phases in the process of design. Essentially, there are three possibilities for feedback-edges from testing back to analysis and synthesis, and from evaluation back to problem identification. In the first case, the design (or form) fails to satisfy one or more of the requirements. The design must then be modified by returning to the synthesis stage. In the second case, new requirements (or constraints) emerge during testing, and the design fails to satisfy one or more of them. The new requirements must then be integrated with the 'current' requirements and further analysis must be done. The outer cycle (Figure 2.8) demonstrates that the evaluated solution might revise the perceived needs.



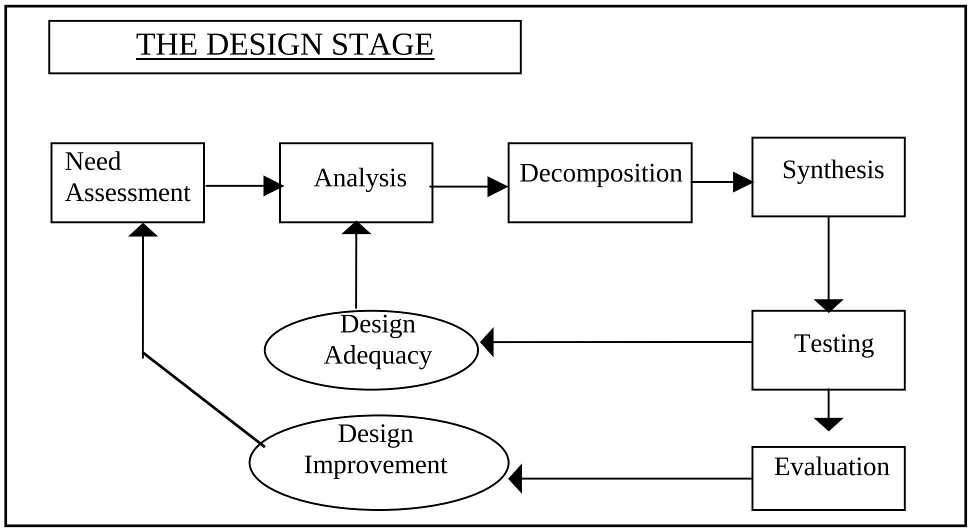


Figure 2.8 Comprehensive Model of the Design Process

Another view (or style) of the above design stages, which is popular in many European countries, is described by [85]. This design model involves the following stages: Clarification (which is similar to the above first two stages); Conceptual Design (which is similar to the above three later stages); Embodiment, where several preliminary geometrical layouts of designs are obtained and evaluated; and Detailed Design (same as above).

The Conceptual Design activity, constitutes the major part of the design process. The stage of conceptual design considerably determines the direction, flexibility and bounds of the design. It has been shown [121] that the conceptual design stage constitutes only 3% of the total product resource costs (research and development), while it determines almost 50% of the product's features including performance, manufacturability, production costs and other concurrent engineering factors. Hence, designers should carefully devise efficient design synthesis and evaluation tools.

The preceding section described mainly the descriptive role of the ASE paradigm. However, from the perspective of design paradigms a more interesting issue is whether the ASE can serve as a basis for developing designs. Alexander [4] devised a method which includes a stage of extensive, detailed and comprehensive analysis of requirements, then one or more stages of synthesis. Sriram and Cheong [109] have provided a brief description of an industrial product, Supercritical Fluid Chromatography, which is based largely on the ASE paradigm. Finally, traditional configuration design procedures of Flexible Manufacturing Systems (FMS) comply with this paradigm [29].

Several methods within the ASE design paradigm have recently evolved from the same foundation of competitiveness in terms of achieving high-quality and low-cost products (these include Concurrent Engineering, Design For Manufacture, Quality Function Deployment and Robust Designs techniques, see [100]). While it is beyond the intention of this book to go into these methods in great detail, an indication of the relation to the ASE design paradigm is given. The most significant of these methods is Quality Function Deployment (QFD), developed in Japan in the 1970s, and popularized by the automobile industry. QFD can be (roughly) described as a four-phase approach to design [74]:

1. Customer requirements planning - translates customer expectations ('the voice of the customer') in the form of market research, competitor analysis and technological forecasts into the desired and specific product characteristics;
2. Product specifications - converts the customer requirements plan for the finished product into its components and the characteristics demanded;
3. Process and quality control plans - identify design and process parameters critical to the achievement of the requirements;
4. Process sheets (derived from the process and quality control plans) - are the instructions to the operator.

Thus, interpreting the QFD process in the terminology of the ASE design paradigm shows that Steps 1-2 constitute an analysis phase, whereas Step 3 constitutes a synthesis phase. The design process style that is invoked in the QFD process is a top-down method (explicitly defined in later sections).

As product designs tend to become conceptually static, QFD will tend to become a more powerful method. It can also be used as a guideline for incremental redesign activity (recall Section 2.3.2). If, however, the design implementation is in the start-up growth stage (as a consequence of innovative redesign, for example), and the customer has yet to experience the benefits of these changes other methods may be invoked.

Although the ASE paradigm is a very widely believed design paradigm in the engineering disciplines, it bears several problems: First, with the explicit ordering of the three stages [22]. Second, with the division of the cognitive activities of analysis and synthesis [95, 114]. Third, with its preclusion of the role of the designer's viewpoint and system of beliefs (or a priori conceptual model) in the process of design. However, if a design problem is well-structured, the design space is sufficiently small (see Section 2.2.4), and the designer uses conceptual models (that is, the overall design of the artifact is known beforehand), then the ASE may be an appropriate paradigm (both descriptive and prescriptive).

2.4.4 CASE-BASED DESIGN PARADIGM

In contrast to other design domains, such as software engineering and circuit design [113], a simple and obvious correspondence between specific functional requirements of the artifact and individual components in the design does not usually exist. Due to the tightly coupled and interacting nature of mechanical designs, reasoning from prior design cases is proving to be a suitable design methodology as opposed to direct 'decompose and recombine' (or 'generate and test') strategies that have successfully been utilized in very-large-scale integration (VLSI) design [116, 111]. Cases are the primary way in which engineering students are taught to design. This is because there are no general algorithms for design. The designer activity is a consequence of his experience and training, much of which is based on previous exposure to similar design problems. This is particularly true in engineering design [85; 39]. Even when a novice engineer joins a design project, an important part of the engineer's training involves going through the design records of previous projects. Cased-based problem solving is based on the premise that a design (or a machine) problem solver makes use of experiences (cases) in solving new problems instead of solving every new problem from scratch [54]. Design cases reflect good design principles, such as function sharing [113] and incorporate decisions that take advantage of, or compensate for, incidental component interactions. Lansdown [65] argues that "innovation arises from incremental modification of existing 'tried and true' ideas rather than entirely new approaches ...the transformation from initial to final description is continuous and design is more like fine-tuning a set of already working ideas rather than inventing something new, although the results might not resemble anything previously imagined." Coyne et al. [20] use the similar term 'prototype model': "A prototype typifies, or exemplifies, a class of designs, and thus serves as a generic design ...a description of a class of designs also may be prototyped or knowledge or rules may even constitute a prototype." A particular design can then be instantiated or exemplified from the class (prototype) of designs.

Coyne et al. [20] classify the case-based paradigm into three activities: creation, modification and adaptation. Creation is concerned with incorporating requirements to create new prototypes. Modifications is concerned with developing a working design from a particular category of cases. Adaptation is concerned with extending the boundaries of the class of cases. Pugh and Morley [89] have conducted extensive design process research by interviewing successful design teams in British industry. The research indicates that embodiment design (models, prototypes etc.) may be produced very early in the process of design, both in incremental and innovative design activities (see III-*A*).

2.4.5 THE COGNITIVE DESIGN PARADIGM

A cognitive model is representative of how people perform a mental task or activity and the interrelationships of active intelligent human designers with computerized tools such as computer aided drafting systems. Protocol studies on how engineers design were conducted by several researchers [2, 37, 119]. In these studies, designers were given problems to solve in a specified amount of time and were asked to think aloud (protocol analysis as a technique to study problem solving behavior is discussed and used extensively in [79]). Based on these studies, the researchers have formulated several models of the design process. For example, Ullman et al. propose a model of the mechanical design process called the Task/Episode Accumulation process. Their model views the design process as consisting of the Conceptual Design, the Layout Design, the Detail Design, and the Catalog Selection stages. A set of ten operators are used to accomplish these stages. They also observed that designers normally pursue only a single alternative, rather than considering multiple alternatives. Sriram and Cheong [109] indicate (based on case studies) that while designers may have difficulty retaining several alternatives in their memory, the designers feel that tools that will aid them to pursue various choices would produce more innovative designs. Many of the features incorporated into CAE (computer aided engineering) tools were influenced by the cognitive studies. CAE tools have been utilized for diverse domains. Paper path handling, air cylinders, buildings and circuits are few examples of domain dependent/independent frameworks developed in the mid 80's. These systems used hierarchical refinements and constraint propagation problem solving strategies.

Throughout the spectrum of the design process categories, the process of creation or ideation often follows a definite pattern [42]:

1. Preparation - defining the situation and gathering facts;
2. Frustration - struggling against mental blocks;
3. Illumination - a sudden spark of insight;
4. Evaluation and execution - assessing alternatives and implementing the optimal choice (contingent to the designer's world view).

The following attributes are identified as common elements of creativity:

1. Capacity for intuitive perception: the recognition of associations and similarities

among objects and concepts;
2. Concern for implications, meanings, and significance;
3. Ability to think imaginatively without regard for practicalities;
4. Open-mindedness toward, for change, improvement, and new ideas rather than rehashing old techniques and traditions. The creative designer is warned of the cost of spending too much effort researching solutions to similar problems of the past.

A number of techniques are available which appear to animate the creative process (a prescriptive view of the cognitive paradigm). Examples are: the trigger word method in which a designer asks himself a series of active questions. The checklist method [82] that relies on a number of questions on modifications. The morphological method that analyzes the problem and determines the independent parameters involved which are then listed on a grid and evaluated systematically. The Gordon technique that attempts to identify the fundamental concepts underlying a given situation rather than emphasizing the obvious characteristics. The brainstorming technique which refers to the spontaneous generation of ideas by a diverse group of individuals, some of whom may know little about the particulars of the problem.

2.4.6 THE CREATIVE DESIGN PARADIGM AND THE SIT METHOD

We briefly present the creative design paradigm and the Structured Inventive Thinking (SIT) method, which efficiently implements and enhance creative problem solving in engineering design. For details see references [150, 151].

The SIT method is a three-step procedure: problem reformulation; general search strategy selection; and an application of idea provoking techniques. The most innovative part of the method is the problem reformulation stage. The given problem is modified through the application of clear, objectively defined and statistically proven set of sufficient conditions for creative solutions. Extensive empirical cases that were analyzed proved with high statistical confidence that the method leads the designer to creative solutions.

The cases also prove the correlation of the SIT method and classical psychological tests of creativity, in two aspects. Students who are creative according to the psychological tests are also achieving better results with the SIT method. Most important, teaching the SIT method enhance creativity in students and engineers.

The SIT theory states that if an idea for a solution of a technological problem satisfies two sufficient conditions, that idea will be deemed creative by field experts. Using SIT, the problem solver first reformulates the problem by changing the goal from ‘find a solution’ to ‘find a solution that satisfies the conditions’. The problem solver then proceeds to the process of searching the solution. At this stage the designer selects one of two general solution strategies, each leading to a different set of idea provoking techniques.

The SIT method uses ideas developed initially by Altshuller who examined thousands of inventions and patents from which he extracted 39 properties that characterize creative solutions. Based on his findings the TRIZ method was developed.

SIT differs from TRIZ in some fundamental aspects, especially in the fact that in SIT only two fundamental principles lead the paradigm, and these principles are clearly and rigorously formulated as the sufficient conditions. Reformulating the problem through the sufficient conditions generates a well-defined and clear criterion leading toward testing a candidate solution. Another important difference between the two methods is that SIT applies a minimal set of techniques, so that after some training the SIT process can become second nature to the problem solver.

SIT is used in many companies including Ford Motor Company in the US (trained by Dr. Sickafus) and many Israeli hi-tech companies. The method is taught as a full credited academic course in Tel-Aviv University, and in the National University of Singapore. SIT practitioners have attained many creative solutions.

Formal Expression of Sufficient Conditions

We start with needed notation related to a situation in which a problem is described in terms of a given (existing) technological system that suffers from (known) undesired effects.

S_i	the given system in the problem state (I for input)
S_o	the system in the solution state (o for output)
$N(s)$	the neighborhood of a system S (the collection of objects which are not an integral part of the system but can be found in the system's proximity or have special affinity to that system)
$O(S)$	the collection of object types from which the system S is composed. Each object stands for the single technological concept that underlies its functioning in the system.
UDE	the collection of variables which contribute, directly or indirectly to the undesired effects, that appear in the problem description
$f^+(y,x)$	y is an increasing function of x, when all other variables remain constant.
$f^-(y,x)$	y is a decreasing function of x, when all other variables remain constant.
$f^0(y,x)$	the value of y is independent of the value of x

If $x, y \in UDE$, $f^+(y,x)$ is called a problem characteristic function.

Using these notations and definitions the two sufficient conditions can be expressed as follows:

The *Closed World* (CW) condition:

$$O(S_o) \cup N(S_o) \subseteq O(S_i) \cup N(S_i) \qquad (1)$$

The *Qualitative Change in Problem Characteristic* (QC) condition:

$$\text{For } x, y \in UDE, \ [f^{+}(y,x)]_{S_i} \wedge [f^{0}(y,x) \vee f^{-}(y,x)]_{S_o} \qquad (2)$$

The expression for the closed *world condition* means that no new object can be added to the system, unless it is a neighborhood object, but objects can be removed from the system. Since $O(S)$ stand for object types and not the objects themselves, more objects of the same type are allowed to be introduced into the system (e.g. add more wheels to a car).

The expression for the *qualitative change in problem characteristic condition* means that a problem characteristic needs to change from an increasing function to either a decreasing or an unchanging function.

The sufficient conditions were developed through an empirical survey of numerous engineering problems and their corresponding routine and creative solutions. Once the conditions were extracted an explanation for the rational behind them may have been induced: Commonly routine design problem solving processes begin with an attempt to tune parameters, and when this fails to produce the desired results, engineers turn to searching alternative technological concepts. The QC condition makes parameter tuning ineffective, and the CW condition does not allow a replacement of existing concepts. Routine processes thus fail to produce the desired results, and the problem solver is forced to resort to more creative processes.

Description of the SIT Mechanism

The SIT mechanism comprises three main steps: problem reformulation through the sufficient conditions; selection of a general thinking strategy; and selection and application of a relevant idea provoking technique.

(i) Problem Reformulation:

At this stage the problem solver sets the target for the problem-solving task using the two sufficient conditions. The CW condition is added to current constraints, while the QC condition changes the goal: instead of the initial (and natural) requirement to decrease the level of an undesired effect, the problem solver is guided to qualitatively change a mathematical relation between any two problem related variables (problem characteristics).

Technically at this stage the user forms a list of system objects, a list of system neighborhood objects, and a list of problem characteristic variables. The problem solving task is defined as follows: Find a solution in which at least one of the defined increasing functions will become decreasing or unchanging subject to the constraint that the solution will incorporate only elements of the given system and its neighborhood that appear in the relevant lists.

(ii) Strategy Selection:

The framework of the sufficient conditions naturally gives rise to two thinking strategies. A candidate solution is composed of three elements: the desired *physical end state* - deduced from the QC condition, the *objects to be modified*, and the *required modification*. The CW condition confines the objects to be modified only to existing ones, and thus significantly narrows the search space. Two scenarios are possible at this stage: The problem solver can deduce a required physical end state from the QC condition. This situation commonly occurs when the desired End State can be achieved through a simple physical operation that will not interfere with other operations required from the system.

The problem solver cannot conceive a desired physical end state, or the state he can think of contradicts other fundamental requirements from the system. These two scenarios define the two possible strategies. Following the first strategy, the problem solver first formulates a conceptual solution: a simple operation that once added to the system, the QC condition is guaranteed to be satisfied. He then proceeds to find an existing object that will carry out the desired operation. Consider for example the problem of testing acid liquid material. In this case, the tested material causes corrosion of the vessel that holds the material. In the course of the solution of the material testing problem, the problem solver can think of the idea to physically separate the acidic liquid from the vessel - an idea that guarantees the satisfaction of the QC condition. He then selects an existing object: the tested samples to carry out this operation. This strategy is called the *extension strategy* to indicate the fact that the system is temporarily extended through the addition of an imaginary object that will carry out the new operation.

If the problem solver reaches at this point the second situation (that is he cannot conceive of a desired end state that would guarantee the satisfaction of the QC condition) he can follow a different strategy: through a trial and error process he tries different possible modifications to existing objects until at some point hopefully he hits a state where the QC condition is satisfied (the CW condition is guaranteed to be satisfied since none of the tried modifications violates it). This strategy is called the *restructuring strategy* to indicate the fact that in the trial and error process the problem solver changes the structure of existing objects and their organization.

The problem solver is guided to select the extension strategy if he can conceive a conceptual solution, and to select the restructuring strategy otherwise. The actual significance of selecting a thinking strategy lies in the application of a different set of idea provoking techniques for each strategy. If the extension strategy was selected the user is directed to apply either *unification* or *multiplication*, if the restructuring strategy is selected the user is directed to apply either *division*, or *increasing variability*. The extension techniques help the problem solver identify an existing object that will carry out the new operation, while the restructuring techniques help him increase the degrees of freedom of possible changes to the system.

(iii) Idea Provoking Techniques:

Idea provoking is the final stage of the method. Their main role is to free the problem solver from fixated mental states.

The Unification Technique. The unification technique helps the problem solver identify a system or neighborhood objects that will carry out the operation defined in the conceptual solution. Applying the technique is a four-step process:
(1) Formulate the needed operation.
(2) Form a list of all main system and neighborhood objects.
(3) Select an object from the list and complete the following sentence: The *selected object* will carry out the *operation.*
(4) Determine the necessary modifications of the selected object, so that it can carry out the desired operation.

Example - the four steps applied to the material testing problem:
(1) The needed operation: to separate the acidic liquid from the vessel.
(2) A list of objects: vessel, samples, and acidic liquid.
(3) Selected object: samples. The samples will separate the acidic liquid from the vessel
(4) Required modification: the shape of the samples will change so that it can contain liquid.

The Multiplication Technique. The purpose of this technique and its first 2 steps (out of four) are identical to the Unification technique. The last two steps are listed below:
(3) Select an object from the list and complete the following sentence: The selected object will multiply. The new copy (or copies) of the of this object will carry out the operation.
(4) Determine the necessary modifications of the new copy (or copies) of the selected object, so that it can carry out the desired operation.

The Division Technique. Being a restructuring strategy technique the purpose of the division technique is to help the user identify new degrees of freedom for modifying and reorganizing system objects. It is a three-step process:
(1) Form a list of system objects.
(2) Select an object from the list and complete the following sentence: The object will be divided to its more basic elements/to smaller parts of the same part/in a random way (select one option from the three).
(3) Search for meaning: try to use the new degrees of freedom to create a state in which the QC condition is satisfied: different parts in different locations, different order of parts etc.

The Increasing Variability Technique. This is another important technique to aid the problem solver in the creation of new degrees of freedom and new ways to solve the problem. It is a four- step process:
(1) Form a list of system objects.

(2) Select an object.
(3) Select two parameters W and Z that are currently not related, that is W is not a function of Z (a new degree of freedom will be the type of relation between them).
(4) Search for meaning: try to use the new degrees of freedom to create a state in which the QC condition is satisfied.

Examples for the Application of SIT

This section will demonstrate how SIT is used to find creative solutions to technological problems. Each example will include problem description, a set of routine solutions (most common responses of engineers), detailed description of how SIT is used to solve the problem, and a description of a creative solution - the output of the SIT process. It is important to note that common SIT applications consist in many trial and error processes (different strategies, different techniques, and different application of the techniques), however the description below does not reflect that important nature of SIT application. For sake of brevity we describe in each example only the direct path to the solution.

EXAMPLE 1 - CORN IN A PIPE

A curved steel pipe is one of the components of a corn grain processing plant (as described in Figure 1). The pipe's function is to conduct the flow of air with the corn grains. The problem is the grains’ impact on the pipe at the bend, which erodes the pipe wall. The air speed cannot be reduced since this will reduce the plant's capacity.

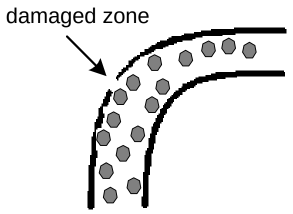


Figure 1 The Corn in the Pipe Problem

Routine ideas:
(1) To strengthen the pipe at the erosion zone (change the material, make it thicker).
(2) To make the curved part of the pipe from a different piece that can be easily replaced.
(3) To coat the pipe with a protective layer that will be replaced from time to time.

SIT Step 1 - problem reformulation:
(1) List of UDE parameters: cost of production, erosion rate, grain flux, grain

hardness.
(2) List of system and neighborhood objects: pipe, grain, and airflow.
(3) The reformulated problem: make erosion rate unrelated to/decreasing function of grain flux, don't add any new object to: pipe, grain and airflow.

SIT Step 2 - strategy selection:
Since a conceptual solution can be conceived - to separate the grains from the pipe the extension strategy is selected.

SIT Step 3 - select and apply an idea provoking technique (unification or multiplication):
Unification was selected, application of the technique:
(1) Formulate the needed operation: to separate the grains from the pipe
(2) Form a list of all main system and neighborhood objects: grains, pipe, airflow
(3) Select an object from the list (grain) and complete the following sentence: The *grain will separate the grain from the pipe.*
(4) Determine the necessary modifications of the selected object, so that it can carry out the desired operation: grains should stick to the curved part of the pipe

The solution: The geometry of the curved area of the pipe will change as to create a pocket that will enable the grains to accumulate there. This will protect the pipe from the damage of grains' impact (see Figure 2).

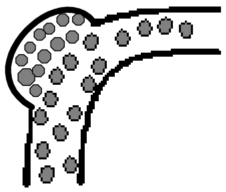


Figure 2 The Solution to the Corn in the Pipe Problem

EXAMPLE 2 - DERAILING DETECTION DEVICE

The braking system of trains includes a pipe that passes along the train, in which the air is at a pressure of 5 atmospheres. When the pressure drops, the train stops. Under emergency conditions (such as derailing), the air must be released very quickly. To ensure fast enough release of the air, it should exit through an opening that is at least 10 cm^2. During normal operating conditions, this opening should be closed with a stopper. The air pressure itself should release the stopper.

A new derailing detector has been developed. The idea is that in normal operation, the stopper is to held in place by the derailing detector, and when derailing occurs the detector stops exerting force on the stopper and it is released.

The problem is that the derailing detector can exert only 0.5 Kgf, not enough to balance the 50 Kgf applied by the internal pressure (see Figure 3).

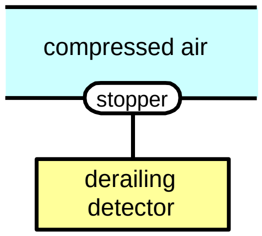


Figure 3 The Derailing Detector System

Routine ideas*:*
(1) To use a lever.
(2) To add more derailing detectors, each will support a smaller stopper.
(3) To squeeze the stopper in its place so that friction will carry out some of the load.

SIT Step 1 - problem reformulation:
(1) List of UDE parameters: probability of false alarm, probability of premature stopper opening, load on the derailing detector, air pressure, and stopper area.
(2) List of system and neighborhood objects: pipe, air, stopper, and derailing detector.
(3) The reformulated problem: make load on the derailing detector unrelated to/decreasing function of air pressure, don't add any new object to: pipe, air, stopper, and derailing detector.

SIT Step 2 - strategy selection:
Since a conceptual solution can be conceived - to exert on the stopper a force that is identical and in opposite direction to the force exerted by air pressure, the extension strategy was selected.

SIT Step 3 - select and apply an idea provoking technique (unification or multiplication):
Multiplication was selected, application of the technique:
(1) Formulate the needed operation: to exert on the stopper a force that is identical and in opposite direction to the force exerted by air pressure.
(2) Form a list of all main system and neighborhood objects: pipe, air, stopper, and derailing detector.
(3) Select an object from the list (stopper) and complete the following sentence: The *stopper* will be multiplied. The new copy (or copies) of this object will *exert on the stopper a force that is identical and in opposite direction to the force exerted by air pressure.*
(4) Determine the necessary modifications of the new copies of the selected object,

so that it can carry out the desired operation: The new stopper should be slightly smaller than the original one so that the derailing detector will still have to carry some load.

The solution: The new stopper will be mounted exactly above the original one, they will be connected through a thin wire (see Figure 4).

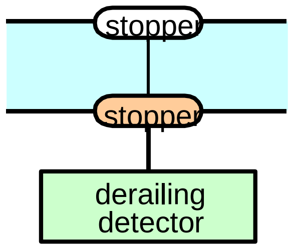


Figure 4 The Solution to the Derailing Detector Problem

EXAMPLE 3 - THE TUMOR PROBLEM

Suppose you are a doctor faced with a patient who has a malignant, inoperable tumor in his stomach. Unless the tumor is destroyed the patient will die. There is a ray that can be used to destroy the tumor. If the rays are directed at the tumor at sufficiently high intensity, the tumor will be destroyed. Unfortunately, at this intensity, the healthy tissue that the rays pass through on the way to the tumor will also be destroyed. At lower intensities, the rays are harmless to the healthy tissue but they will not affect the tumor.

Routine ideas:
(1) In this problem the most common ideas either violate problem definition (for example to try to operate although the problem text explicitly states that the tumor is inoperable) or suggest alternative treatment such as chemotherapy.

SIT Step 1 - problem reformulation:
(1) List of UDE parameters: probability of patient's death, damage to healthy tissues, and rays intensity.
(2) List of system and neighborhood objects: rays, tumor, and healthy tissues.
(3) The reformulated problem: make damage to healthy tissues unrelated to/decreasing function of rays intensity, don’t add any new object to rays, tumor, and health tissues.

SIT Step 2 - strategy selection:
Since a conceptual solution cannot be conceived the restructuring strategy is selected.

SIT Step 3 - select and apply an idea provoking technique (division or increasing variability)
Division was selected, application of the technique:
(1) Form a list of system objects: rays (this is the only system object).
(2) Select an object from the list (rays) and complete the following sentence: The *rays* will be divided *to smaller parts of the same type.*
(3) Search for meaning: try to use the new degrees of freedom to create a state in which the QC condition is satisfied (different parts in different locations, different order of parts etc.): The smaller rays will be directed at the tumor from different angles.

The solution: To direct a few weak beams at the tumor from different angles, so that they converge at the tumor and develop sufficient intensity there to destroy the tumor. The QC condition was satisfied since it is possible to increase the ray intensity at the tumor by adding more weak rays without affecting healthy tissues through which the rays pass.

EXAMPLE 4 - SOLID FUEL ROCKET ENGINE

One of the problems that designers of solid-fuel rocket engines (Figure 5) faced was the necessity of achieving a constant thrust from the engines. The solid-fuel rocket engine has the shape of a hollow cylinder burning in an internal envelope. The problem with such geometry is that the thrust is not constant owing to a change in the area of the internal envelope (the radius increases). When the internal combustion area increases, the thrust increases.

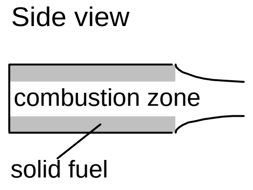


Figure 5 Side View of A Rocket Engine

Routine ideas:
(1) A new parametric design: the dimensions of the cylinder are changed so that it will become longer and narrower. These changes maintain its total volume and combustion area, but since the difference between initial and final radius is smaller, the variance is smaller.
(2) "Cigar burning" - a cylinder burning in its base.

SIT Step 1 - problem reformulation:
(1) List of UDE parameters: energy waste, uneven thrust, thrust increase, burning

area increase, perimeter increase.
(2) List of system and neighborhood objects: solid fuel, rocket, and thrust.
(3) The reformulated problem: make burning area unrelated to/decreasing function of perimeter, do not add any new object to solid fuel, rocket, and thrust.

SIT Step 2 - strategy selection:
Since a conceptual solution cannot be conceived the restructuring strategy is selected.

SIT Step 3 - select and apply an idea provoking technique (division or increasing variability).
Increasing variability was selected, application of the technique:
(1) Form a list of system objects: solid fuel, rocket, and thrust.
(2) Select an object: solid fuel.
(3) Select two parameters W and Z that are currently not related, that is W is not a function of Z (a new degree of freedom will be the type of relation between them): Z - cross section shape; W - combustion progression.
(4) Search for meaning: try to use the new degrees of freedom to create a state in which the QC condition is satisfied: The shape of the cross section will change through the combustion progression from a very complicated winding shape to a pure circle.

The solution: the shape of the cross-section is such that it maintains a constant perimeter while combustion progresses. The cross-section changes from a complex shape to a simple circle, thus, although the average radius increases, the perimeter remains constant. Figure 6 demonstrates the idea. This solution preserves the initial concept of a hollow shape burning in the internal envelope thus complying with the closed world condition. Since the variance in thrust is constantly zero, totally independent of the difference between initial and final radius, the solution satisfies the QC condition as well. Note that, at its time, this solution was a breakthrough in solid fuel engines. In our workshops we see students using the sufficient conditions finding this solution quite quickly.

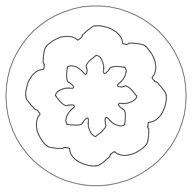


Figure 6 The New Inner Envelope as Combustion Progresses

Conclusion

This section presented a set of objective sufficient conditions for creative solutions, and a three-step method that structures the search toward obtaining engineering design solutions.

The application of the sufficient conditions to a specific problem modifies the given problem definition: The QC condition changes the goal of the search and the CW condition confines the search space. By applying the sufficient conditions, the creative problem solver actually solves a different problem than his non-creative companion, an important factor in finding a different, sometimes surprising solution. Note that the QC condition is related to the functional decoupling requirement in Nam Suh's paradigm (see Section 2.5.1).

The detailed extensive study of the validity of the sufficient conditions and the SIT method can be found in the references.

2.4.7 THE ALGORITHMIC DESIGN PARADIGM

The algorithmic design paradigm is the focus of a considerable amount of current and past research of design automation. The algorithmic design paradigm views the design process as the execution of an effective domain-specific procedure that yields a satisfying design solution (relative to a given initial requirements) in a finite number of steps. The main premise of this paradigm is that the requirements are well-defined and there are precisely defined criteria for determining whether or not a design meets the requirements. That is, the notion that design problems are well-structured (see Section 2.2.4).

There exist a number of instances of the algorithmic paradigm which serve to optimize complex systems: exhaustive search, rapid search and mathematical programming techniques. Within the algorithmic paradigm, exhaustive search is a lengthy process resulting in global optimization within the field of inquiry. A style is defined as a set of attributes that enables the designer to discriminate between one set of artifacts and another in the same group. A style, therefore, represents certain search modes (design process styles) on the part of the design algorithm, the result of which is the nature of the final design [see also 103]. A number of search modes exist for algorithmic paradigms: breadth-first, depth-first, greedy method, branch and bound, dynamic programming and so on.

The price of global optimization through an exhaustive search of alternatives is tremendous. The alternative to an exhaustive search is rapid search, where a set of simple but arbitrary guidelines are adopted to limit the search space. For example, in serial optimization, as each stage is optimized (e.g. selecting the types of materials and method of manufacture), the selections at the subsequent stages are evaluated conditionally with the assumption that the preceding choices hold. The algorithm proceeds in this way throughout the series of stages. Serial-optimization may be the most widely used design style in conscious human decision-making. In term of effort, it is clearly superior to exhaustive search. The greatest disadvantage of any rapid search method lies in the questionable proximity to the global optimum; the rapid-search algorithms use arbitrary guidelines for optimization. Notwithstanding the global optimization potential, many instances within the algorithmic design paradigm produce satisfying rather than optimal solutions. To understand the difference between exhaustive and rapid search, consider the domain of arc welding design. Over the years, numerous researchers have studied various aspects of the welding process, such as understanding the underlying fluid mechanics, heat transfer, phase transformation and solid mechanics of the welding process. Attempts to incorporate them into a rapid search strategy often results in a nonsystematic, somewhat random method in which designers formulate a set of decisions that ultimately result in the welding process. Most decisions are made to optimize one aspect of the process, rather than the process as a whole. This often results in a suboptimal design of the welding process. A globally optimized welding process, through exhaustive search, may be achieved by requiring a complete and thorough understanding of the complex interactions among various aspects of the welding process. The difficulty with this approach is the global understanding required of an incredibly large data base.

Mathematical programming techniques are a recognized topic in many engineering design courses, and are discussed at length in texts on engineering design theory [101, 24]. Mathematical programming techniques can be used to identify the potential design configuration (e.g. the physical design of electronic circuits) by optimizing the configuration based on the functional requirements. In general, in these methods the solution to the problem is developed by solving the mathematical model consisting of an objective function that is to be optimized and a set of constraints representing the limitation of the resources.

In summary, although most interesting design problems are incomplete, open-ended and ill-structured (see Section 2.2.5), they may decomposed into one or more well-structured components. In this case, the algorithmic paradigm may be successfully utilized to solve each of these well-structured sub-problems. Thus, the algorithmic paradigm may be considered as a tool that can support and be invoked by other paradigms [17, 22].

2.4.8 THE ARTIFICIAL INTELLIGENCE DESIGN PARADIGM

Artificial intelligence (AI) is the field that attempts to make computers perform tasks that usually require human intelligence. The AI design paradigm is based upon capturing the knowledge of a certain domain and using it to solve problems [34]. In order to automate a design process, a design system must be able to differentiate between various choices and determine the best path. The AI design paradigm views design as a problem-solving process of searching through a state-space, from an initial problem state to the goal state, where the states represent the design solutions. Transitions from one state to another are affected by applying one of a finite set of operators, based on the functional requirements (goals) and design constraints (constituting the domain specific knowledge) and meta-rules (constituting the domain independent knowledge). The design process involves representing much of their knowledge about the problem declaratively. Roughly speaking, declarative knowledge is encoded explicitly in the knowledge-base in the form of sentences in some language (usually in the form of IF condition THEN action), and procedural knowledge (which typifies the algorithmic design paradigm) is manifested in algorithms.

The AI paradigm of design relies heavily on the Function-Structure-Behavior of an artifact and their inter connections through causality. People often refer to artifacts based on the functions they provide, and that existing structures could be combined into new ones, to achieve the desired function [30]. Bobrow [11] defines function as the relation between a goal of a human user and the behavior of the system. Structure is defined as the information about the interconnection of modules, organized either functionally - how the modules interact - or physically - how it's packaged. Behavior can be defined as the relationship between input from the environment and the output of affect the component usually interfaces to the environment. In summary, the proponents of the AI design paradigm claim that through a causal reasoning approach using the problem-solving process of searching and the basic physics behind behaviors, the structure and set of behaviors to achieve a given goal in as little time as possible can be accomplished.

The use of knowledge-based expert systems (KBES) has become common enough to understand their benefits in a problem such as this. An expert system is able to use previously defined rules and cases to choose through a new problem to solve it. This would enable an expert system to pick previous design structures and behaviors and match them to the design specifications. How a knowledge-based expert system should be constructed depends on where the problem lies on the analysis-synthesis spectrum [109]. In analysis (e.g., process diagnostics), the problem conditions are posed as parts of a solution description; the possible outcomes exist in the knowledge-based of a KBES. Essentially, the solution to these problems involves the identification of the solution path. In synthesis problem (or design) problems conditions are given in the form of properties that a solution must satisfy as a whole; an exact solution does not (normally) exist in the knowledge-base, but the inference mechanism can generate the solution by utilizing knowledge in the knowledge-base. Figure 2.9 depicts a multi-level framework for describing knowledge-based design as enunciated by [117]. KBESs are instances of automatic problem solvers that rely heavily on domain-specific heuristics, are also often called strong methods [80]. Knowledge-based expert systems provide the support for many of the automatic or computer-aided design systems developed in recent years, such as buildings design, circuit design, paper path handling and air cylinders [117; 97]. Sriram and Cheong [109] have identified the following tenets of computer-aided design systems:

1. Incorporate design plans, design knowledge and design constraints;
2. Deal with evolving specifications;
3. Display geometry and have the ability to associate constraints with geometrical entities (relating structure to behavior);
4. Provide access to distributed knowledge/database of devices;
5. View multiple alternatives simultaneously;
6. Provide access to manufacturability knowledge;
7. Generate assembly sequences automatically from geometry.

When less is known about the design task environment, domain-independent control strategies are more appropriately evoked. Problem solvers that rely heavily on domain-specific heuristics, are also often called weak methods [80]. Weak methods are used to effect and control the search through the state-space. The so-called means-ends analysis [79] lies at the center of these methods (see also Section 2.3.2). Means-ends analysis was employed, among others, by [77] in the domain of automatic program synthesis; and by [34] in designing room configurations.

The AI design paradigm may be combined with other design paradigms to establish a 'grand' problem solving strategy for the designer or design system. It is only very recently that the use of past cases is beginning to recognized in the design automation literature [76, 32, 47]. The process of case-based design consists of the following steps that are iteratively applied as new sub-goals are generated during problem-solving [115]:

1. Development of a functional description through the use of qualitative relations explaining how the inputs and outputs are related;
2. Retrieval of cases which results with a set of design cases (or case parts) bearing similarity to a given collection of features;
3. Development of a synthesis strategy which describes how the various cases and case pieces will fit together to yield a working design;
4. Realization of the synthesis strategy at the physical level;
5. Verification of the design against the desired specifications through quantitative and qualitative simulation;
6. Debugging which involves the process of asking relevant questions and modifying them based on a causal explanation of the bug.

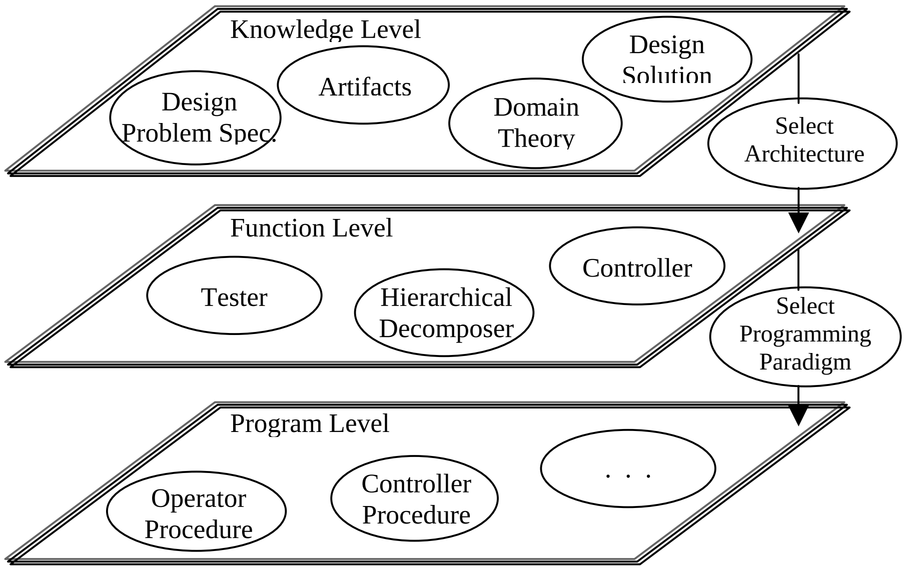


Figure 2.9 Multilevel Framework for Describing Knowledge-based System (adapted from [109])

Case-based problem solving has several advantages over knowledge-based expert systems: first, cases provide memories of past solutions, failures and repairs that have been used successfully. Secondly, from a knowledge acquisition view, there is experimental evidence that designers don't explicitly think in terms of rules [107]. Moreover, asking a designer to give examples of cases, is much easier than asking him to give a list of rules he uses to design. Finally, design case studies are readily available in the literature.

Simon pointed up [105] that the design process strategies (domain-dependent as well as domain-independent) "can affect not only the efficiency with which resources for designing are used, but also the nature (form style) of the final design as well." Conversely, the designer is likely to use some special features of the problem to identify one or a small number of styles that are evoked in the process of design (that is, a form style as a determinant of a control strategy or design process style). The main three types of design process styles [105] are bottom-up, top-down and meet-in-the-middle. Bottom-up design style involves starting with basic structures and combining these structures until the final form is accomplished. This design style incorporates search trees and optimal branch calculations. Top-down design starts with the final behavior required and sub-divides this behavior into smaller behaviors which are ultimately linked to components and their respective structures. This design process style is similar to bottom-up design in that it uses search trees and optimal branch calculations. Meet-in-the-middle design process style incorporates the previous two design process styles. Either top-down or bottom-up is chosen according to the difficulty encountered in the design process and the amount of previous information available.

To sum up, the AI design paradigm is very useful in solving tightly coupled, highly integrated and ill-structured design problems. However, when faced with an original design problem with no previous rules or past cases to help it, the expert system or case-based problem solver are incapable of original creativity. Thus the expert system or case-based problem solver are capable of helping in the design process but are not the solution to design automation. Moreover, the real problem with AI in design is in determining a way to make the computer program be creative in a manner that is manageable and not an NP hard solution. If the program just generates every possible solution, something the computer excels at, the amount of time generating and checking the solutions may be infinite. The design program must be able to generate a small selection of design solutions and choose between them for the ultimate choice. The human designer is able to do this quite easily but has an extensive language and other extraneous factors to assist in this process.

2.4.9 DESIGN AS A SOCIAL PROCESS

The ASE design paradigm (the "consensus" paradigm) describes the engineering design process as a sequence of activities leading to intermediate results. Moreover, many design theorist still insist that design theory requires universal methods analogous to universal methods used in the natural sciences. In contrast, it is the contention of the *social constructivist* approach that the study of design and sociological studies should, and indeed can, benefit from one another [127].

According to the social constructivist approach, design as a social process involving designers, customers, and other participators consists of creating and refining a shared meaning of requirements and potential solutions through continual negotiations, discussions, clarifications, and evaluations [126]. From the social constructivist perspective, the consensus model overlooks or underplays the social factors involved in the design process. For example, negotiation can enter into all phases of the process. There is also no mention of the problems of sharing knowledge among members of a design team responsible for different parts of the design. Each member of the design team is usually also a member of different research and engineering traditions which conceptualize problems differently and see the design as a whole, on the basis of different analogical models [125].

Another contention of the constructivist approach is that the consensus model aims at formal models of design. However, even within the sphere of formal models, a considerable degree of informal activities take place. Moreover, formal models are incomplete in their ability to trade off alternatives, which are mediated by inherently social processes. The connecting thread between these activities or approaches is that they are all expected to provide insight into the problem at hand. These activities are meant to facilitate a better understanding of needs and problems encountered, and potential solutions [126]. The resulting modeling activities can be modified into modeling activities which are results of both socio-linguistic, and more precise formal languages. Formal models are evaluated along several dimensions: accuracy, applicability, intent, and mutual consistency. Still, the very definition of these dimensions is a negotiated outcome of the design process, and not an input, a priori or otherwise [126]. The process of acceptance of formal methods is based on their reliability in a situated context of the domain of application [125, 126]. For example, the use of optimization methods in chemical engineering is much more accepted and stable than in mechanical engineering.

Several empirical findings and proposals (pertaining to how designers work) seem to support the social constuctivist view [125, 126, 128]:

- Different designers and social groups use different vocabularies to describe the same or very closely related sets of things [129]. For example, for some, the artifact air tire introduced in the bicycle, was a solution to the vibration problem of small-wheeled vehicles. For others, the air tire was a way of going faster. For yet another group of engineers, it was an ugly looking way of making the low-wheeler even less safe (because of side-slipping) then it already was [127].
- Engineers typically spent a significant portion of their time (more than 50%) in documenting and communicating - much of it in the form of formal or informal negotiations. The negotiation most often takes the forms of one-on-one meetings and paper being passed about within the organization.
- The sociocultural, political, legal, and ecological situation of a social group shapes its norms and values, which in turn influence the meaning given to the artifact. Because different meanings can constitute different lines of development, this seems to offer an operationalization of the relationship between the wider context and the actual content of technology.
- “Simultaneous” or “concurrent” engineering are terms that have gained a lot of currency recently. In each of these uses of the term, the underlying premise is

> that traditional design processes lack information on the later phases of the product realization process (such as production and operation) in the early phases of the development of the product [125]. In a study analyzing the traditional design and development process Danko and Prinz [130] conclude that successful design depends on the exchange of information between appropriate groups in the process. Further, rework and redesign are rampant, because field recognized conflicts often require design and field changes. Clark and Fujimoto [131] conclude, based on their study of American, Japanese, and European automobile companies, that Japanese firms are organized to maximize knowledge sharing between suppliers and the parent, and through very quick problem solving cycles that involve separate departments. This integration of problem solving helps reduce mistakes and rework and enables tool and die shops to handle changes with fewer transactions and less overhead [131].

In interpreting these findings, note that the underlying phenomenon being described is that of collaboration (or, in the traditional approach, lack thereof) among individual separated by disciplines or functional responsibility. However, effective simultaneous engineering is effective interdisciplinary design which, in turn, is the creation of effective *shared meaning,* the persistent form of it being *shared memory* [125]. Konda et al [125] define shared memory as the taking-hold of shared meaning created in specific design situations and applied to other design situations. Shared memory needs a substrate, or an infrastructure, in which the initial construction of a shared language are stored and perhaps reactivated at a later time on the same or a different project. Shared memory is distinguishable from shared meaning in that the former can have a more physical existence in, for example, databases, cross-indexes, models, papers, and so on [125, 126].

The phrase shared memory is used to denote the increasingly detailed aspect of a given profession's knowledge (*vertical shared memory*), as well as the sharing of meaning among multiple disciplines, groups, and group members (*horizontal shared memory*). Vertical shared memory is the codified corpus of knowledge, techniques, and models that exist in every professional group. The need for horizontal sharing among multiple disciplines arises from the general observation that engineering product development is a collaborative process, where engineers from diverse disciplines cooperate to specify, design, manufacture, test, market, and maintain a product [128]. The need for horizontal sharing *within* a profession arises from the differences in functioning contexts and meaning. For example, the beginning of the bicycle's development brought out two different kind of social groups within the profession of cyclists: (1) the social group of cyclists riding the high-wheeled Ordinary consisted of "young men of means and nerve" [132]. For this social group the function of the bicycle was primarily for sport; and (2) some parts of the safety low-wheelers and the safety ordinaries can be better explained by including a separate social group of feminine cyclists.

Shared memory needs to be established in and nurtured by appropriate organizational structures incorporating the necessary technical substrates through which information, tools and, most importantly, people interact. Shared memory needs a substrate, or an infrastructure, in which the initial construction of a shared language are stored and perhaps reactivated at a later time on the same or a different project [125]. The substrate must address the following objectives:

- Facilitate capturing the context and history of a particular design (cases of both formal and informal modeling) in order that the experience (good or bad) can be reused within another design context;
- Facilitate the negotiation among designers and with other relevant parties (such as users);
- Forecast the impact of design decisions on manufacturing;
- Provide designers interactively with detailed manufacturing process planning;
- Develop mechanism for reaching agreements among designers through consensus standardization.

Following the general observation that the majority and most critical of activities in design are informal [126], no single representation or abstraction technique can be imposed on designers *a priori*, without severely limiting their capability to model. Therefore, the substrate of shared memory should allow multiple classifications, languages and methods designers find appropriate to carry out the above activities. Thus, the substrate can benefit from research on tools developed in Artificial Intelligence (AI), such as qualitative physics, semantic network representations, rule structures, machine learning, information retrieval techniques (relational databases), hypermedia, graph grammars, etc.

To achieve the objectives outlined above, several system architectures - based on current trends in programming methodologies, object-oriented databases, and knowledge based systems were developed. For example, the *n-dim* project, currently underway at the Engineering Design Research Center, Carnegie Mellon University, is a computer environment to support collaborative design [133]. n-dim is also a history capturing mechanism for complex corporate activities such as design. Other important aspects of n-dim are a task-level view for configuring and managing the design process, and an information-management system that allows for defining and displaying a user’s current design context by means of *models* (linked information objects; see [126]). Other related work include the DICE (**D**istributed and **I**ntegrated environment for **C**omputer-aided **E**ngineering) being pursued at the Massachusetts Institute of Technology [128], the STEP/PDES effort, the RATAS project [134], the EDM model [135], and the spatial representation work being pursued at Carnegie Mellon University [136] are relevant tools that support collaboration among distributed teams of persons carrying out a complex process such as design.

Although automation facilitates the creation of effective shared memory, Clark and Fujimoto [131] argue that “competitive advantage will lie not in hardware and commercial software, but in the organizational capability to develop proprietary software and coherently integrate software, hardware, and “humanware” into an effective systems.” For example, some degree of horizontal memory (which derives collaborative behavior) can be achieved based on *reward structures* to encourage the exchange of information among multiple disciplines, groups, and groups members. Hence, enhancing communication between human designers from different perspectives is feasible using organizational methods such as assignment of team responsibility and proximity of the designers, or through techniques such as Quality Function Deployment (QFD) for matching quality control and customer preferences [137]. In short, automation should not drive context - automation should be driven by context.

2.5 SCIENTIFIC STUDY OF DESIGN ACTIVITIES

The field of design theory is relatively new, which has been particularly stimulated by three computer-related technological advances: computer-aided design (CAD) [9], knowledge-based expert systems (KBES) [117], and concurrent engineering (CE) [100]. The main source of the slow development and confusion about design theory is that engineering design lacks the sufficient scientific foundations. Dixon [25] argued that engineering design education and practice lack an adequate base of scientific principles, and are guided too much by the specialized empiricism, intuition, and experience. Kuhn [58, 59] concluded that design is at a prescience phase and it must go through several phases before it constitutes a mature science (hence theory), which is that state of a discipline in which there is a coherent tradition of scientific research and practice, embodying law, theory, application and instrumentation. In order to achieve this kind of maturity, designers must borrow the methodologies from other disciplines (such as artificial intelligence, neural networks, logic and fuzzy logic, object oriented methods) that have reached relative scientific maturity [20].

A design method (within a design paradigm) does not constitute a theory; theory emerges when there is a testable explanation of why the method behaves as it does [25]. Design methods do not attempt to say what design is or how human designers do what they do, but rather provide tools by which designers can explain and perhaps even replicate certain aspects of design behaviors. The major components and aims of design theories are:

1. To construct a systematic inquiry into a phenomenon which is to uncover some intelligible structure or pattern underlying the phenomenon. That is, a theory of design must be in part descriptive [22];
2. Theories must be in part prescriptive, that is has the capability of specifying how design should be done, and allow us to construct more rational methods, and tools to support practical design;
3. Theories must be simple; that is, when two design theories are possible, we provisionally choose that which our minds adjudge to be the simpler, on the supposition that this is the more likely to lead in the direction of the truth. It includes as a special case the principle of William of Occam - "Causes shall not be multiplied beyond necessity";
4. Theories must be consistent with whatever else we know or believe to be true about the universe in which the phenomenon is observed;
5. The value of a design theory be determined, to a great extent, by its generality

and domain-independence (including mechanical engineering, electrical engineering, civil engineering and computer science).

While discussions of design theory, from a number of different perspective, have appeared in [105, 45, 49, 21, 52, 108, 35], in the following sections an attempt is made to demonstrate and concentrate on two interesting design theories - the axiomatic theory of design, and design as scientific problem-solving. The former theory is more grounded in the 'real' environment of design while the latter theory is a more abstract, speculative or philosophical.

2.5.1 THE AXIOMATIC THEORY OF DESIGN

The axiomatic theory of design is a structured approach to implementing a product's design from a set of functional requirements, that was developed by [113, 112]. It is a mathematical approach to design which differentiates the attributes of successful product and demonstrates design that are not manufacturable. Suh defines design as the culmination of synthesized solutions (in the form of product, software, processes or system) by the appropriate selection of design parameters that satisfy perceived needs through the mapping from functional requirements in the functional domain to design parameters in the structure domain. Suh pointed out that the fact that empirical decisions often lead to superior designs (as supported by technological progress) indicates that there exists a set of underlying principles, heuristics or axioms which govern the decision making process. If these axioms can be formalized and their corollaries derived, then it should be possible to establish a scientific basis for guidelines in manufacturing design [113]. The axiomatic approach is based on the following premises:

1. There exist a small number of axioms which will always lead to superior decisions in terms of increased overall productivity;
2. These heuristics may be established and examined through an empirical studies;
3. Axiomatic is not a mechanism for generating acceptable designs, but a tool to aid in the decision making process.

The hypothesized principles of manufacturing axiomatic may be given as two axioms:

1. *The Independence Axiom* - In an acceptable design, the design parameters and the functional requirements are related in such a way that specified design parameter can be adjusted to satisfy its corresponding functional requirement without affecting other functional requirements (functional independence is analogous to the concept of orthogonality in linear algebra.) The independence axiom does not imply that a part has to be broken into two or more separate physical parts, or that a new element has to be added to the existing design. Functional decoupling may be achieved without physical separation, although in some cases such physical separation may be the best way of solving the problem

(recall the bicycle's evolution);

2. *The Information Axiom* - The best design has minimum information content. It coincides with the principle of 'simplicity'. The search for simplicity, likewise the search for beauty, is a powerful aesthetic imperative that serves as a basic component of a designer's value system. Simple in the design means, for example, being able to minimize the number and complexity of part surfaces. The simplicity principle implies that if a design satisfies more than the minimum number and measure of functional requirements originally imposed, the part or process may be overdesigned. Simple design will result in reducing product cost (such as inventory and purchasing costs), and enhancing quality.

Suh et al. have developed a number of theorems and corollaries which may readily be shown to be implied by the axioms. These corollaries bear strong resemblance to many design guidelines and design for manufacture (DFM) techniques (recall the ASE design paradigm) that were developed in manufacturing companies [12]. Some of these guidelines (corollaries) include: minimizing functional requirements, decoupling of coupled design, integration of physical parts, symmetry, standardization and largest tolerance.

2.5.2 DESIGN AS SCIENTIFIC PROBLEM-SOLVING

Discussions in the literature of design processes generally treats separately a category of the artificial sciences (engineering disciplines) and the natural sciences (such as physics, biology and geology). Several criteria that demarcate the artificial sciences from the natural sciences have been identified [105, as well as others]:

1. Engineers are concerned with how things ought to be, that is in order to attain goals, and to function while science concerns itself solely with how things are. In an ultimate philosophical sense, the natural science has found a way to exclude the normative, and to concern itself with solely with descriptive aspects of nature;
2. Engineering is concerned with 'synthesis' while science is concerned with 'analysis';
3. Engineering is 'creative, intuitive and spontaneous' while science is 'rational and analytic'.

Schön [98] has proposed that design creates an entirely new epistemology (epistemology is concerned with the question of what knowledge is and how it is possible), which he terms 'reflection-in-action' as contrasted to scientific discovery which concerns with 'technical rationality'. Cross [21] and Coyne et al. [20] have claimed that the aims of design and those of science differ. Coyne et al. summarized elegantly the demarcation between the natural science and design as: "science attempts to formulate knowledge by deriving relationships between observed phenomena. Design, on the other hand, begins with intentions and uses the available knowledge to arrive at an entity possessing attributes that will meet the original intentions. The role of design is to produce form or more correctly, a description of form using knowledge to transform a formless description into a pragmatic discipline concerned with providing a solution within the capacity of the knowledge available to the designer. This design may not be 'correct' or 'ideal' and may represent a compromise, but it will meet the given intentions to some degree".

That there are indeed differences in aims will be agreed in general. Unfortunately, this demarcation of the natural sciences from engineering as a result of the differences in their respective aims has led to the fictitious attitude that the methodology of science and engineering are fundamentally different. Laudan [66] stresses at the outset of his essay Progress and its Problems that scientific problems are not fundamentally different from other kinds of problems, though they are different in degree. Indeed, we shall show that this view can be applied, with only a few qualifications, to all intellectual disciplines, and design activity in particular. Engineers wishing to construct an artifact capable of implementing a process (such as problem solving) often study naturally occurring systems that already implement the process. This approach has led, inter alia, many researchers to investigate psychological models of human thought as a basis for constructing the constituent methods within the AI design paradigm [79, 119]. We shall describe in the sequel those theories (which conclude, among others, [105, 55, 28, 5, 1, 22]) that support the thesis that the engineering and natural science are methodologically indistinguishable, and that the structure of problem solving of scientific communities (scientific discovery) can be used to justify many of the decisions encountered in the design process. The relationship is one that can be seen at a high level of abstraction. The most important aspects of this relationship are forms of scientific discovery which pursue the hypothetico-deductive (H-D) or the related procedure of abductive inference, and Kuhn's model of scientific progress [58, 59, see also Section 2.4.1] whose primary element is the "paradigm" (and more elaborated ideas such as the notion of research programmes by Lakatos [61, 62], and Laudan's [66] idea of research tradition). We focus our attention on scientific progress and how scientific communities solve problems, following by a brief outline of the parallelism to design processes. We hope to gain useful insight from these parallelism that will aid us to corroborate the thesis that design and natural science are methodologically indistinguishable.

The publication of Thomas Kuhn's The Structure of Scientific Revolutions in 1962 was an important milestone in the development of the historiography of science. It was the first attempt to construct a generalized picture of the process by which a science is born and undergoes change and development envisaged by Kuhn's model and further developed and refined by [61, 66]. Their approach may be summarized as follows (see Figure 2.10):

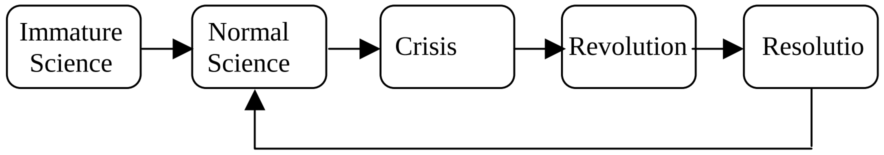


Figure 2.10 Five Stages in the History of Scientific Disciplines According to Kuhn

1. *Immature Science*: A pre-paradigm stage in which the natural phenomena that later form the subject matter of a mature science are studied and explained from widely differing points of view [58, 59].

2. *Normal Science*: The emergence of a paradigm (e.g., Newtonian mechanics, quantum mechanics), embodied in the published works of one or more great scientists, defining and exemplifying the concepts and methods of research appropriate to the study of a certain class of natural phenomena, and serving as an inspiration to further research by its promise of success in explaining those phenomena.

A period of normal science conducted within a conceptual and methodological framework derived from the paradigmatic achievement, involving actualization of the promise of success, further articulation of the paradigm, exploration of the possibilities within the paradigm, use of existing theory (a set of hypotheses) to predict facts, solving of scientific puzzles, development of new applications of theory, and the like. Lakatos, as well as Laudan, contend that science is seldom dominated by just one paradigm, as Kuhn claims in his account of normal science, but rather that competition between paradigms generally co-occurs with processes of development within a paradigm. Lakatos replaces Kuhn's term paradigm with the term research programme (for example, we can distinguish between the flat-world and round-world research programmes). The common thread linking different theories into a common research programme is a "hard core" of basic assumptions shared by all investigators. This core is surrounded by a "protective belt" of auxiliary assumptions. The "hard core" which may consist of assumptions such as "No action at a distance," remains intact as long as the research programme continues, but researchers can change the auxiliary assumptions in the protective belt to accommodate evidence that either has accumulated or is developed in the course of research. Laudan invokes the idea of a large-scale unit in science that he calls a "research tradition". Like Lakatos' research programmes, research traditions for Laudan consist of a sequence of theories, but they lack a common core that is immune to revision. What holds a research tradition together are simply common ontological assumptions about the nature of the world and methodological principles about how to revise theories and develop new theories.

Kuhn, Lakatos and Laudan agree that the main activity of scientists (within a paradigm, research programme or tradition) is problem solving. Scientific problem solving consists of generating hypotheses to account for phenomena, and procedures for substantiating and refuting these hypotheses. The Positivists called the former procedure (substantiation) for developing scientific theories the hypothetico-deductive (H-D) method. Popper has called the latter procedure (refutation) conjecture and refutation (C-R). The basic idea of the H-D method is that scientists begin with a phenomena (anomaly, experimental or conceptual problem) that requires explanation. Having developed a hypotheses (for the Positivists how hypotheses were arrived at was not a matter for logical inquiry), the task was to test and discover whether the hypotheses was true. If it was; it could provide the theory needed to explain the phenomena. The hypothesis is a general statement and so could be tested by considering initial hypothesis, and deriving predictions about what would happen under these hypothesis. If these predictions turn out to be true, the initial hypothesis would be confirmed; if the predictions turn out false, the hypothesis would be disconfirmed. In either case, a new problem (phenomena) would have been generated and the cycle begins once more. The Positivists thought that at least positive tests of a hypothesis could give support to that hypothesis. Popper [86] contended that this assumption was false; observation statements ("this swan is white") can never logically imply theories ("all swans are white") but they can logically refute them (by providing a counterexample of just one black swan). He proposed instead that scientists should begin by making conjectures about how the world is and then seek to disprove them. If the hypothesis is disproved, then it should be discarded. If, on the other hand, a scientist tries diligently to disprove a hypothesis, and fails, the hypothesis gains a tentative or conjectural stature. Although failure to disprove does not amount to confirmation of the hypothesis and does not show that it is true or even likely to be true, Popper speaks of such an hypothesis as corroborated. The virtue of a corroborated hypothesis is that it is at least a candidate for being a true theory, whereas hypotheses that have been disproved are not even candidates. Popper terms this the process of conjectures and refutations (C-R).

3. *Crisis*: A crisis stage of varying duration precipitated by the discovery of natural phenomena that "violate the paradigm-induced (research programme or tradition) expectations that govern normal science" and marked by the invention of new theories (still within the prevailing paradigm) designed to take account of the anomalous facts.

4. *Revolution and Resolution*: A relatively abrupt transition to a new paradigm (research programme or research tradition) that defines and exemplifies a new conceptual and methodological framework incommensurable with the old; and continuation of normal science within the new paradigm (research programme or tradition).

In addition to the foregoing properties, let us state the following important features:

- Conjectures often develop by adjusting to new problems by discovering the discrepancy between their predecessors and the phenomena (recall Newell and Simons' General Problem Solver scheme). This process led, for example, to better understanding of Euler's formula and the concept of "polyhedron" [see 62];
- When an anomaly is discovered in a theory, the outcome is often an adjustment [62] of the theory rather than a total dismissal of the existing paradigm (which defines the character and structure of the original theory), research programme or tradition. Most theories evolve through a continual and incremental activity

(recall the concept of 'incrementalism' given by [69] in the context of design);

- Although corroboration does not give us a logical basis for increasing our confidence that the hypothesis will not be falsified on the next test, it does serve to limit us to an ever narrowing set of hypotheses that might be true. Popper has compared this process to Darwinian natural selection, for both nonadapted organisms and false theories are weeded out, leaving the stronger to continue in the competition. Evolution is often gradual, taking years or decades. Other theorists have pursued the idea that theory development may be parallel to the process of evolution by natural selection [see, for example, 58, 59, 16, 118];
- As pointed out above, for the Positivists how hypotheses were arrived at was not a matter for logical inquiry, since discovery was assumed to be nonrational process. Hanson [40] was one of the first to urge philosophers to redirect attention to discovery (extending the context of 'scientific justification'). His proposal was to pursue what the 19th century American Pragmatist, Charles Peirce, called abductive inference which is similar to the H-D method. The alternative to deductive reasoning is generally taken to be induction. One of the things that has brought about renewed interest in discovery is the recognition, partly motivated by work in empirical psychology, that human reasoning involves additional modes of reasoning than deductive logic and enumerative induction such as "common sense" knowledge, gestalt-like perception, analogical reasoning or by sheer trial and error. Because scientific reasoning is simply an extension of ordinary human reasoning, there is reason to think that such strategies figure also in science. Newell and Simon [79] popularized the idea that in solving complex problems we rely on heuristic principles that simplify the process through which we search for a solution. Recently there has been considerable interest by both philosophers and those in Artificial Intelligence in using AI as a tool for studying scientific reasoning [64];

Following our previous discussions on descriptive properties of design, especially the adaptive and evolutionary properties discussed in Section 2.3.2, and the prescriptive role of design paradigms, it is plausible to believe the hypothesis that there is a direct and striking resemblance between the structure of design processes and the foregoing structure of problem solving of scientific communities is corroborated (following Popper). Figure 2.11 illustrates the parallel relationship between inquiry that is associated with science and that which is associated with engineering design. This correspondence is summarized as follows:

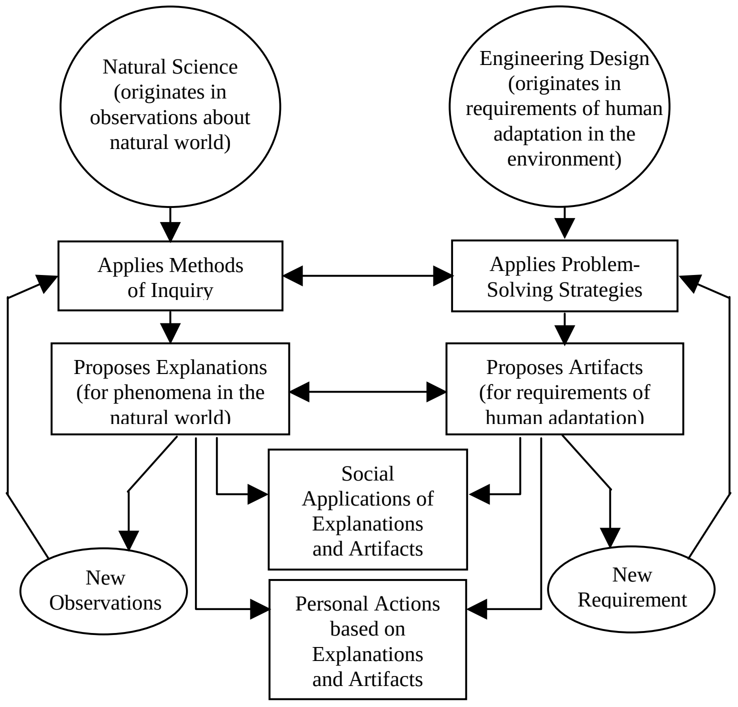

Natural Science (originates in observations about natural world)
Engineering Design (originates in requirements of human adaptation in the environment)
Applies Methods of Inquiry
Applies Problem-Solving Strategies
Proposes Explanations (for phenomena in the natural world)
Proposes Artifacts (for requirements of human adaptation)
Social Applications of Explanations and Artifacts
New Observations
New Requirement
Personal Actions based on Explanations and Artifacts

Figure 2.11 The Interrelationships between the Science and Engineering Design

- The counterpart of the Kuhnian paradigm or Laudan's research tradition is the designer's knowledge-base needed to generate the set of design solutions. The designer's knowledge-base is a coherent tradition of knowledge and practice, embodying theories (such as stress analysis, thermodynamics analysis, information on materials and the like), design paradigms, applications, heuristics (transformation operators), instrumentation, and law (such as existing zoning regulations). The most versatile designers will represent much of their knowledge about their environments declaratively. The set of knowledge underlies the designer's conceptual assumptions and world view and values serving as best approximation to what the design domain actually is; so, we really should be talking about the designer's beliefs rather than the designer's knowledge. But, following the tradition established by the phrase "knowledge-based systems", we will speak of the designer's knowledge;
- The counterpart of a set of phenomena, events or problems are design problems that are entirely characterized by and generated as a result of measurable and non-measurable requirements. Laudan distinguishes two kinds of problems that can confront a research tradition - empirical inadequacies of the current theories

and conceptual problems with the theories comprising the tradition. It is with a direct correspondence to the distinction (see Section 2.2.4) between empirical, measurable or well-defined requirements (which specifies externally observable or empirically determinable qualities for an artifact); and ill-defined requirements (conceptual);

- The counterpart of a scientific theory (set of hypotheses) is the tentative design/form (constitutes a systematic representation of functional relationships of the components; see II-*H*) serving (much the same as scientific theories) as a vehicle for the designer to capture her thoughts, as a plan for implementation, and as a vehicle for reflecting the evolutionary history that led to the emergence of the final form/design, thus facilitating the inspection, analysis and redesign (change) of the artifact;
- Scientific discovery follows the hypothetico-deductive (H-D) method, or the more justifiable procedure (following Popper) of conjecture and refutation (C-R). Moreover, Popper [as well as 58, 59, 16, 118] has compared the process of scientific discovery to Darwinian natural selection. It is with a direct correspondence with the evolutionary nature of design processes (see Section 2.3.2): as the design process develops (due to bounded rationality), the designer test (to corroborate or disprove the hypotheses) the tentative design against the requirements. If the test fails, the designer modifies either the tentative design or requirements (the counterpart of theory adjustment), so as to remove the discrepancy (recall how Euler's formula was adjusted to a new "counterexample"; see [62]) between them, and converge or establish a fit between the two parts. Testing involves a wide range of reasoning types, such as classical and approximate logical systems (theorem provers, qualitative reasoning); experiments (e.g., simulations); knowledge accumulated from previous experience; and common sense reasoning (heuristics and 'rules of thumb'). If a problem is found as the result of testing, it also becomes a new problem to be solved in another design cycle;
- Besides the embodiment of ontogenetic design evolution within the model of scientific discovery, Kuhn's model of scientific progress can also explain phylogenetic design evolution (see Section 2.3.2). Incremental redesign activity corresponds to the continual and incremental evolvement of scientific theories within a normal science, whereas innovative redesign activity corresponds to a transition to a new paradigm (conceptual or paradigm-shift);
- Comparison of theories in terms of "degree of falsifiability", a concept most fully developed by Popper [86] and Newtonian world-view of "reductionism", cast considerable light on the axiomatic theory of design [113]. Popper concludes that the best reason for entertaining a theory is that it is testable (more accurately "falsifiable") - i.e., that it makes strong predictions that are readily capable of being refuted by evidence if the theory is, in fact, false. Let us, following Popper, restate the criterion in a number of other forms. Theory T_1 (e.g., "All heavenly bodies move in circles") is decidedly stronger than Theory T_2 ("All planets move in ellipses") if it is more universal and precise. Since in a theory it is desirable to obtain the maximum of prediction from the minimum of

assumptions, the more universal and precise a theory, hence the more falsifiable, the better. In the case of two theories related as T_1 and T_2 we will entertain the weaker, T_2, only if the stronger, T_1, is falsified. Another form is that of simplicity. “Simple” theories are generally thought preferable to “complex” theories. Consequently, for the same design problem, a computer time-sharing system design, T_1, which was accompanied by larger amounts of experimental work is stronger than a computer time-sharing system accompanied by poor amounts of experimental work. Similarly, as previously noted, Suh’s information axiom and derivations (such as the minimization theorem and corollaries, see [113]) are derived from Popper’s (and Occam’s razor) simplicity principle.

What we propose to do now is to briefly indicate how Suh’s independence axiom and derivations, as well as various hierarchical decomposition principles of design problems represent a direct embodiment (from its paradigmatic aspect) of the 17th-century Newtonian mechanics. The essential point of the Newtonian reductionistic language is that, in the Newtonian picture, the categories of causation are isolated into independent mathematical elements of the total dynamics. Indeed, the independence axiom and the principle of a nearly decomposable system (A hierarchical system of components $C_1,...,C_n$ where each C_i is itself an aggregate of more primitive entities such that the interactions between the entities within the C_i’s are appreciably stronger than those between the C_i’s is called a nearly decomposable system, see [105]) are nothing but a paraphrase of the Newtonian language, adapted to inherently non-mechanical situation.

- Both designer’s knowledge-base and Laudan’s research tradition have the component of a group’s shared past cases (used within the case-based design paradigm) and examples. By that it means the concrete problem-solutions and cases that students encounter from the start of their scientific and engineering education, whether in laboratories, on examinations, or at the technical problem-solutions found in the periodical literature and information about wide variety of devices (their parts, characteristics, materials, uses and behavior) found from commercial catalogs.
- The ASE design paradigm is based on a phase of thorough analysis of the requirements, followed by the actual synthesis of design, and then a phase of testing the design against the requirements, which is strikingly resemble to the one of the most influential methodologies of science, namely, inductivism. According to inductivism only those propositions can be accepted into the body of science which either describe hard facts by making observations and gathering the data (the counterpart of an analysis phase of requirements) or are infallible inductive generalizations (the counterpart of a synthesis phase of design) from them. The problem with inductivism (as well as with the ASE paradigm) is that the inductivist cannot offer a rational ‘internal’ explanation for why certain facts rather than others were selected in the first instance. Hanson [41] argued that observation itself is theory-laden, that is that what we perceive is influenced by what we know, believe, or are familiar with (which implies that

designers have some conceptual models prior to gathering requirements).

The above discussion on the interrelationship between the science and engineering design emphasizes *individual* human design problem solving. To summarize, Popper argues that theories ought to prove their mettle through several critical tests (theories should be falsified rather then confirmed). On the other hand, the approach of Kuhn, Laudan and Lakatos is historical and case-based. They argue that theories are always more than a single hypothesis. Theories are usually an interweaving of a number of law-like rule-like statements supplemented by an often larger number of auxiliary hypotheses concerning instruments used in testing them and specifications of initial conditions and experimental set-up necessary for these tests. Our analysis of design methodologies shows that these two alternative view of scientific development (logical reconstructability versus case-based approach) were utilized to gain useful insight on the parallelism between the methodologies of science and design methodology (emphasizing individual problem solving).

Thus far, we have not discussed the fact that design processes as well as scientific progress involve social factors (see Section 2.4.9 about a discussion on design as a social process); have similar mechanisms of organizing collaborative activities among multiple disciplines, groups, and groups members (e.g. negotiations); and have similar mechanism for allocating effort (such as funding structure and peer review). Indeed, the views of Kuhn, Laudan and Lakatos have been elaborated more recently by others who pointed out that social and sometimes cultural factors can be significantly involved in bringing scientific debate to a relative close and stabilizing concepts of facts [127, 138, 139, 140, 141]. These approaches have generated a vigorous program of empirical research on the process of the social construction of scientific knowledge in various sciences. See, for example, Collins [141] for scientific controversy, Collins and Pinch [142] for physics and biology, and DeMillo et al. [143] for program verification in computer science. These approaches, which we refer to as "*social constructivist*," mark an important new development in the *sociology of science*. The main characteristics of the social constructivist view is described by the "Empirical Programme of Relativism" (EPOR) as was developed in the sociology of scientific knowledge. Three stages in the explanatory aims of the EPOR can be identified [127]:

1. In the first stage (Interpretative Flexibility), it is shown that scientific findings are open to more than one interpretation. Often opposing directions of research are found between schools of thought that do not closely interact. For example, the Freudian and behaviorist schools in psychology explain human neuroses in very different ways. Neither has proved that its ideas are in all respects superior to its competitors. There is no good reason for the community as a whole to accept one theory over the other. In the absence of criterial experiments to discredit one theory or the other, these opposing ideas may coexist for a long time. Moreover, all thought happens in the context of certain assumptions. For example, we can distinguish between the *flat world* and the *round world* hypotheses by imagining two viewpoints, one for each of the theories. We would expect statements in the flat-world viewpoint to the effect that the world

has edges and the belief in the round-world viewpoint that traveling west long enough will bring you back to where you started. Research effort in opposing directions is not only found between distinct schools of thought; it can often be found in the same research group or individual.

2. When an anomaly is discarded in a theory, the outcome is often an adjustment of the theory rather than a total dismissal. This is because theories that have shown some success tend to develop a core of fundamental concepts that serve to motivate further work. The process of adjustment, then, is an effort to protect this core from being discredited. Evolution is often gradual, taking years or decades. On rare occasions it is quite rapid as the community reacts to a "breakthrough" that clearly decides an issue (controversy) previously muddled. Social mechanism that limit interpretative flexibility, and thus allow scientific controversies to be terminated are described in the *second stage* of EPOR.
3. The sociocultural and political situation of a social group shapes its norms and values, which in turn influence the meaning given to theories. A *third stage* is to relate the "closure mechanism" to the wider social-cultural milieu.

Having described the EPOR approach to the study of science, and the social constructivist approach to the study of design (as described in Section 2.4.9), we now discuss the parallels between them [127]:

1. *Interpretative Flexibility* - In design, there is flexibility in how designers interpret artifacts, and how artifacts are designed. There is not just one possible way or one best way of designing an artifact. For example, for some, the artifact air tire introduced in the bicycle, was a solution to the vibration problem of small-wheeled vehicles. For others, the air tire was a way of going faster. For yet another group of engineers, it was an ugly looking way of making the low-wheeler even less safe (because of side-slipping) then it already was [127]. Moreover, other artifacts were seen as providing a solution for the vibration problem, e.g. the saddle, the steering bar, and solutions used spring construction in the frame;
2. *Closure and Stabilization* - Closure in design involves the stabilization of an artifact and the resolving of technological controversy. Pinch and Bijker [127] identify two closure mechanisms. Rhetorical closure emerges when the relevant social groups *see* the design problem as being solved (although it might not be solved in the common sense of the word). For example, for some time, the "safety controversy" around the high-wheeler bicycles was resolved by announcing (through advertisement) that the artifact (i.e. the air tire) was perfectly safe although to engineers high-wheeled bicycles were known to have safety problems. Closure by redefinition of the problem means translating the meaning of the artifact to constitute a solution to quite another problem. For example, originally the air tire meant a solution to the vibration problem to the users of the low-wheeled bicycle. However, the group of sporting cyclists riding their high-wheelers did not accept that as a problem at all. Eventually, closure has been reached when the meaning of the air tire was translated to constitute a solution to quite another problem: the problem of how to go as fast as possible

[127].

To summarize, we argue that the social constructivist view that is prevalent within the sociology of science provides useful insights that will aid in the interpretation of design processes with respect to the social factors involved.

2.6 A GENERAL DESIGN METHODOLOGY

This section presents a design methodology, based on the scientific community metaphor, by emphasizing the variational (or parametric) design part. In variational design, the dimensions of a part are calculated by solving a system of constraints or specifications (typically, nonlinear equations). Let us summarize some of the very basic features of the evolutionary design model as articulated in Chapter 6 and Chapter 17:

- In variational design, an artifact at any particular abstraction level is described in terms of part types (a group of objects which are similar but have different sizes). Every part can be described by a set of attributes. Each attribute can be described by its dimension (such as wire diameter, spring diameter, number of active coils, and modulus of elasticity).
- Specifications or constraints, at any particular abstraction level, are the various functional, behavioral, performance, reliability, aesthetic, or other characteristics or features that are to be present in the physically implemented artifact. In the case of the Gear Box design (see Chapter 17), the initial specifications were to design “a mass production device, which can be used for lifting light objects or opening a garage door in a family or small warehouse.” In variational design, closed-form constraints are usually either *Euclidean* (including distance, tangency, parallelism, and so on), or *functional* (such as mass properties, forces, stiffness, strength, rating life, and so on). A higher order constraint is a property that is satisfied by lower order constraints.
- Design proceeds as a succession of cycles. In each cycle, the objects that evolve are the design/specifications complexes; the evolution of the design/specifications complexes is towards the satisfaction of tentative specifications; and the *mechanism* employed in this evolutionary process is the attempt to verify the validity (“degree of believability”) of existing specifications. As a consequence of this mechanism, new specifications and design parameters are introduced. If the design satisfies the specifications, then there is a fit between the two, otherwise there is said to be a misfit between design and specifications. For example, the Gear Box design process is terminated when the transmission parts, casing, shaft set, and accessories are fully specified, and all user-specified requirements (duration, capacity), strength constraints and heat balance are satisfied by the current solution. This evolutionary design model follows the hypothetico-deductive (H-D) method, or the more justifiable procedure (following Popper) of conjecture and refutation

(C-R) of scientific discovery (see Section 2.5.2).

The validity of specifications (*qualitative* design specifications as well as *closed-form equations*) as encountered by the designer is determined by means of (1) relevant *knowledge* of the design domain such as *tools* and *techniques* of verification (e.g. finite-elements or finite-differences); and (2) the process *history*, which includes both the sequence of all process states (i.e., the design/specifications complexes) visited so far, together with the transformations and production rules (or inference rules) used to modify such process states. Consider the following examples:

1. Constraint C_1 ("to insure a long work life for the rope") is considered validated only if constraint C_2 ("the overbending of the rope is avoided") is validated. This in turn, is considered validated only if constraint C_3 ("the minimum drum diameter should be large enough") is validated. According to the engineering recommendation value (knowledge of the design domain), for the minimum drum diameter to be large enough, a 15 to 20 times the rope diameter should be chosen. Therefore, the designer chooses the drum diameter as $d_{dr} = 3.75$ inches (a design parameter). The *tool* used for support of constraint C_3 is simply to check if the selected drum diameter is indeed 15 to 20 times the rope diameter;
2. The constraint "select an efficient reducer", is determined by a concept selection process (the *verification tool*), which is used to measure the degree of efficiency of each alternative with respect to well-defined criteria. This in turn, is considered validated only if the design parameter ("the reducer is a worm and wormgear type") is validated (or selected). The specifications and design parameters have dependencies among them. Dependencies between the specifications and parameters are represented by *rules*, or logical relationship the validity of which implies the validity of the specification or the design parameter. The rules are specified in the first-order calculus. For example, the validity of specification C ("the shear strength and compression stress of the key must be less than the allowable shear strength and compression stress, respectively ") is determined by the logical relationship: ("the shear stress should be calculated" $\wedge$ "the compression stress should be calculated" $\wedge$ "the key's material should be specified" $\wedge$ "the shear stress must be less than the allowable shear stress of the material" $\wedge$ "the compression stress must be less than the allowable compression stress of the material"). In this case, the specification part is updated by replacing specification C by its *antecedents* (C is called *consequent*, see [149]). The specification "the key's material should be specified" derives a new design parameter DP "the material of the key is ASTM 40." In this case, the *design part* is updated by adding the new design parameter DP.

A most distinct feature of the design evolutionary model is that the design process is *constantly subject to revision*. For example, in the Gear Box design process, if the shear strength and compression stress specifications are not satisfied by the key, the designer has either to change the key material (DP_1), or change the shaft diameter DP_2 (the key size is associated with the diameter of the shaft). In that case, all specifications that dependent on DP_1 and/or DP_2 will have to be *revised* in the light of this new design state. The evolutionary design process is, thus, *non-monotonic* in nature in accordance with Kuhn's model of scientific progress (see Section 2.5.2).

2.7 SUMMARY

To conclude this chapter it is useful to restate that design contains a wide range of concepts. Design begins with the acknowledgment of needs and dissatisfaction with the current sate of affairs and realization that some action must take place in order to solve the problem. Design science is a collection of many different logically connected knowledge and disciplines constituting miscellaneous design paradigms. Although there is no single paradigm that can provide a complete definition of the design process, there are common characteristics that form the framework within which various paradigms are utilized.

We maintained that the demarcation of the natural sciences from engineering design, as a result of the differences in their respective aims, has led to the fictitious attitude that the methodologies of science and engineering design are fundamentally different. Alternatively, we display the parallelism between the natural sciences and engineering design. In summary we believe that the scientific community metaphor (as captured by a number of philosophers, sociologists and historians of science) can supply important insights that will aid in the interpretation (a descriptive role) and construction (a normative role) of design problem solving systems. The resulting framework has been useful in guiding the development of general purpose design process meta-tools as shown in Parts III and IV of the book. Finally, we may realize that design theorists can expect no easier fate than that which befell scientists in other disciplines.

REFERENCES

1. Agassi, J., *Technology: Philosophical and Social Aspects*. Tel-Aviv: Open University,1985.
2. Akin, O., "An Exploration of the Design Process," *Design Methods and Theory*," Vol. 13 (3-4), pp. 115-119, 1979.
3. Alagic, S. and Arbib, M.A, *The Design of Well-Structured and Correct Programs*. Berlin: Springer-Verlag, 1978.
4. Alexander, C., *Notes on the Synthesis of the Form*. Cambridge MA: Harvard University Press, 1964.
5. Altshuller, G.S., *Creativity as an Exact Science*. New York: Gordon and Breach Publishers, 1984.
6. Antonsson, E.K., "Development and Testing of Hypotheses in Engineering Design Research," *Journal of Mechanisms, Transmissions, and Automation in Design*, Vol. 109, pp. 153-154, 1987
7. Asimow, W., In *Emerging Methods in Environment Design and Planning* (ed. Moore G.T.). Cambridge. MA: MIT Press, pp. 285-307, 1962.
8. Bell, C.G. and Newell, A., *Computer Structures: Reading and Examples*. New York: McGraw-Hill, 1971.
9. Bezier, P.E., "CAD/CAM: Past, Requirements, Trends," In *Proc. CAD*, Brighton, pp. 1-11, 1984.
10. Billington, D.P., *Robert Maillart's Bridges: The Art of Engineering*. Princeton. NJ: Princeton

University Press, 1979.
11. Bobrow, D.G., *Qualitative Reasoning About Physical Systems: An Introduction*." Cambridge. MA: MIT Press, pp. 1-5, 1985.
12. Boothroyd G. and Dewhurst P., *Product Design for Assembly*. Wakefield, RI: Boothroyd & Dewhurst Inc, 1987.
13. Braha, D. and Maimon, O., "A Mathematical Theory of Design: Modeling the Design Process (Part II)," *International Journal of General Systems*, Vol. 25 (3), 1997.
14. Brown, D.C. and Chandrasekaran, B., "Expert Systems for a Class of Mechanical Design Activity." In [33], pp. 259-282, 1985.
15. Brown, D.C. and Chandrasekaran, B., "Knowledge and Control for a Mechanical Design Expert System," *Computer*, Vol. 19 (7), July, pp. 92-100, 1986.
16. Campbell, D.T., "Evolutionary Epistemology." In *The Philosophy of Karl Popper*, P. Schlipp (ed.), LaSalle. IL: Open Court, 1974.
17. Chandrasekaran, B., "Design Problem Solving: A Task Analysis," *AI Magazine*, Winter, 1990.
18. Charniak, E. and McDermott, D., *Introduction to Artificial Intelligence*. Reading. MA: Addison-Wesley, 1985.
19. Churchman, C. W., "The Philosophy of Design," *ICPDT*, Boston, Mass, August., pp. 17-20, 1987.
20. Coyne, R.D., Rosenman, M.A., Radford, A.D., Balachandran, M. and Gero, J.S., *Knowledge-Based Design Systems*. Reading, MA: Addison-Wesley, 1990.
21. Cross, N. (ed.), *Development in Design Methodology*. New York: John Wiley, 1984.
22. Dasgupta, S., "The Structure of Design Processes," In *Advances in Computers*, Vol. 28, M.C. Yovits (ed.). New York: Academic Press, pp. 1-67, 1989.
23. Diaz, A. R., "A Strategy for Optimal Design of Hierarchical System Using Fuzzy Sets," In *NSF Engineering Design Research Conference*, University of MASS., Amherst June, pp. 11-14, 1989.
24. Dieter, G.E., *Engineering Design: A Materials and Processing Approach*. New York: McGraw-Hill, 1983.
25. Dixon, J.R., *AI EDAM*, Vol. 1 (3), pp. 145-157, 1987.
26. Dong, Z., "Evaluating Design Alternatives and Fuzzy Operations," *International Conference on Engineering Design*, Boston, MASS, August, pp. 17-20, pp. 322-329, 1987.
27. Fenves, S.J., Flemming, U., Hendrickson, C., Mehar, M.L. and Schmitt, G., "Integrated Software Environment for Building Design and Construction," *Computer Aided Design*, Vol. 22 (1), pp. 27-35, 1990.
28. Fetzer, J.H., "Program Verification: The Very Idea," *Comm. ACM*, Vol. 31 (9), September, pp. 1048-1063, 1988.
29. *FMS Handbook*, CSDL-R-1599 U.S Army Tank Automotive Command under contract No. DAAE07-82-C-4040, 1983.
30. Freeman, P. and Newell, A., "A Model for Functional Reasoning in Design," In *Proc. of the 2nd Int. Joint Conf. on Artificial Intelligence*, pp. 621-633. 1971.
31. Garey, M.R. and Johnson, D.S., *Computers and Intractability: A guide to the Theory of NP-Completeness*. San Francisco: W. H. Freeman and Company, 1979.
32. Gero, J.S., "Prototypes: A New Schema for Knowledge Based Design," *Technical Report*, Architectural Computing Unit, Department of Architectural Science, 1987.
33. Gero, J.S. (ed.), *Knowledge Engineering in Computer-Aided Design*. Amsterdam: North-Holland, 1985.
34. Gero, J.S. and Coyne, R.D., "Knowledge-Based Planning as a Design Paradigm," In *Design Theory for CAD*, H. Yoshikawa and E.A.Warman (eds.). Amsterdam: Elsevier Science Publishers, 1987.
35. Giloi, W.K. and Shriver, B.D. (eds.), *Methodologies for Computer Systems Design*." Amsterdam: North-Holland, 1985.
36. Glegg, G.L., *The Science of Design*. Cambridge, England: Cambridge University Press, 1973.
37. Goel, V. and Pirolli, P., "Motivating the Notion of Generic Design with Information-Processing Theory: The Design Problem Space," *AI Magazine*, Vol. 10 (1), pp. 18-38, 1989.
38. Gould, S. J., *Ontogeny and Phylogeny*. Cambridge. MASS: Belknap Press of the Harvard University Press, 1977.
39. Gregory, S.A., "The boundaries and Internals of Expert Systems in Engineering Design," *Proceeding of the Second IFIP Workshop on Intelligent CAD*, pp. 7-9, 1988.
40. Hanson, N.R., "An Anatomy of Discovery," *The Journal of Philosophy*, Vol. 64, pp. 321-352, 1967.
41. Hanson, N.R., *Patterns of Discovery*. Cambridge, England: Cambridge University Press, 1958.

42. Harrisberger, L., *Engineersmanship*. Belmont, California: Brooks/Cole Publishing, 1966.
43. Henderson, P., *Functional Programming: Application and Implementation*. Englewood Cliffs, NJ: Prentice-Hall International, 1980.
44. Holton, G., *Introduction to Concepts and Theories in Physical Science*. Reading. MA: Addison-Wesley, 1952.
45. Hubka, V., *Principles of Engineering Design*. London: Butterworth Scientific, 1982.
46. Hubka, V. and Eder, W.E., *Theory of Technical Systems: A Total Concept Theory For Engineering Design*. Berlin: Springer-Verlag, 1988.
47. Huhns, M. H. and Acosta, R. D., "Argo: An Analogical Reasoning System for Solving Design Problems," *Technical Report AI/CAD-092-87*, Microelectronic and Computer Technology Corporation, March, 1987.
48. Ishida, T., Minowa, H. and Nakajima, N., "Detection of Unanticipated Functions of Machines," In *Proc. of the Int. Symp. of Design and Synthesis*, Tokyo, pp. 21-26. 1987.
49. Jaques, R. and Powell, J.A. (eds.), *Design: Science: Method*. Guildford, England: Westbury House, 1980.
50. Johnson-Laird, P.N., *The Computer and the Mind*. Cambridge, MA: Harvard University Press, 1988.
51. Jones, J.C. 1963, "A Method of Systematic Design," In *Conference on Design Methods* (J.C. Jones and D. Thornley, eds.), pp. 10-31, Pergamon, Oxford. Reprinted in [21].
52. Jones, J.C., *Design Methods: Seeds of Human Futures* (2nd Edition). New York: John Wiley, 1980.
53. Kant, E. and Newell, A., "Problem Solving Techniques for the Design of Algorithms," *Information Processing and Management*, Vol. 20 (1-2), pp. 97-118, 1984.
54. Kolodner, J. L., Simpson, R. L. and Sycara, K., "A Process Model of Case-Based Reasoning in Problem Solving," *Proceedings of IJCAI-85*, Los Angeles, pp. 284-290, 1985.
55. Kornfeld, A.W. and Hewitt, E.C., "The Scientific Community Metaphor," *IEEE Transactions on Systems, Man and Cybernetics*, SMC-11 (1), 1981.
56. Krishnamoorthy, C.S., Shivakumar, H., Rajeev S. and Suresh, S., "A Knowledge-Based Systems with Generic Tools for Structural Engineering," *Structural Engineering Review*. Vol. 5 (1), 1993.
57. Kuhn, T.S., *The Structure of Scientific Revolutions*. Chicago, IL: University of Chicago Press, 1962.
58. Kuhn, T.S., Postscript - 1969. In The Structure of Scientific Revolutions. Chicago, IL: University of Chicago Press. Enlarged 2nd Edition, pp. 174-210, 1970.
59. Kuhn, T.S., "Reflections on My Critics." In [63], pp. 231-278, 1970.
60. Kuhn, T.S., "Second Thoughts on Paradigms," Reprinted in T.S. Kuhn, *The Essential Tension*. Chicago, IL:University of Chicago Press, 1977.
61. Lakatos, I., "Falsification and the Methodology of Scientific Research Programmes," In [63], pp. 91-196, 1970.
62. Lakatos, I., *Proofs and Refutations*. Cambridge, U.K: Cambridge University Press, 1976.
63. Lakatos, I. and Musgrave, A. (eds.), *Criticism and the Growth of Knowledge*. Cambridge, U.K: Cambridge University Press, 1970.
64. Langley, P., Simon, H.A., Bradshaw, G.L. and Zytkow, J.M., *Scientific Discovery: Computational Explorations of the Creative Process*. Cambridge, MA: MIT Press, 1987.
65. Lansdown, J., *Design Studies*, Vol. 8 (2), pp.76-81, 1987.
66. Laudan, L., *Progress and Its Problems*. Los Angeles: University of California Press, 1977.
67. Lawson, B., *How Designers Think: The Design Process Demystified*. London: Architectural Press, 1980.
68. Lewin, K., Dembo, T., Festinger, L., and Sears, P.S., "Levels of Aspiration." In *Personality and the Behavior Disorder*, Hunt J.M. (ed.). New York: The Ronald Press, 1944.
69. Lindblom, C.E., "The Science of Muddling Through," *Public Administration Review*, Vol. 9, pp. 79-88, 1959.
70. Luckman, J., *Operational Research Quarterly*, Vol. 18 (4), pp. 345-358, 1967.
71. Maher, M.L., "A Knowledge-Based Approach to Preliminary Design Synthesis," *Report EDRC-12-14-87*, Carnegie Mellon University Engineering Design Research Center, 1987.
72. Maimon, O. and Braha, D., "A Mathematical Theory of Design: Representation of Design Knowledge (Part I)," *International Journal of General Systems*, Vol. 25 (3), 1997.
73. Maimon O. and D. Braha, "On the Complexity of the Design Synthesis Problem," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 26 (1), 1996.
74. Manton, S.M., "Engineering for Quality," In *Taguchi Methods*, Bendell, Disney and Pridmore (eds.), IFS publications, 1989.

75. Masterman, M., "The Nature of a Paradigm," In [63], pp. 59-90, 1970.
76. Mostow, J., "Toward Better Models of the Design Process," *The AI Magazine*, Spring, pp. 44-57, 1985.
77. Mueller, R.A. and Varghese, J., "Knowledge-Based Code Selection in Retargetable Microcode Synthesis," *IEEE Design and Test*, Vol. 2 (3), pp. 44-55, 1985.
78. Murthy, S.S. and Addanki, A. 1987, "PROMPT: An Innovative Design Tool," In *Proc. of the 6th Nat. Conf. on Artificial Intelligence*, Seattle, WA.
79. Newell, A. and Simon, H. A., *Human Problem Solving*. Englewood Cliffs, NJ: Prentice-Hall, 1972.
80. Newell, A. and Simon, H. A., "Computer Science as Empirical Enquiry: Symbols and Search," (ACM Turing Award Lecture), *Comm. ACM*, Vol. 19 (3), March, pp. 113-126, 1976.
81. Nilsson, J.N., "Logic and Artificial Intelligence," *Artificial Intelligence*, Vol. 47, pp. 31-56, 1991.
82. Osborn, A.F., *Applied Imagination*. New York: Charles Scribner's Sons. 1963.
83. Paynter, H.M., *Analysis and Design of Engineering Systems*. Cambridge, MA: MIT Press, 1961.
84. Penberthy, J.S., *Incremental Analysis and the Graph of Models: A First Step Towards Analysis in the Plumber's World*, S.M. Thesis, MIT Department of Electrical Engineering and Computer Science, 1987.
85. Pahl, G. and Beitz, W., *Engineering Design*. Berlin: Springer-Verlag, 1988.
86. Popper, K.R., *The Logic of Discovery*. London: Hutchinson (originally published, 1935), 1959.
87. Pugh, S., *Total Design*. New York: Addison-Wesley, 1990.
88. Pugh, S. and Smith, D.G., "CAD in the Context of Engineering Design: The Designers Viewpoint," In *Proceedings CAD*, London, pp. 193-198, 1976.
89. Pugh, S. and Morley, I.E., *Toward a Theory of Total Design*, University of Strathclyde, Design Division, 1988.
90. Rehg, J., Elfes, S., Talukdar, S., Woodbury, R., Eisenberger, M. and Edahl, R., "CASE: Computer-Aided Simultaneous Engineering," *AI in Engineering Design*, Gero, J.S. (ed.), Springer-Verlag, Berlin, pp. 339-360, 1990.
91. Ressler, A.L., "A Circuit Grammar for Operational Amplifier Design," *Technical Report 807MIT*, Artificial Intelligence Laboratory, 1984.
92. Rieger, C. and Grinberg, M., "The Declarative Representation and Procedural Simulation of Causality in Physical Mechanisms," In *Proc. of the 5th Int. Joint Conf. on Artificial Intelligence*, pp. 250, 1977.
93. Rinderle, J.R., *Measures of Functional Coupling in Design*, PhD dissertation, MIT, 1982.
94. Rinderle, J.R., "Function and Form Relationships: A basis for Preliminary Design," *Report EDRC-24-05-87*, Carnegie Mellon University Engineering Design Research Center, Pittsburgh. PA, 1987.
95. Rittel, H.W. and Webber, M.M., "Planning Problems are Wicked Problems," *Policy Sciences*, Vol. 4, pp. 155-169, Reprinted in [21], pp. 135-144, 1973.
96. Roylance, G., "A simple Model of Circuit Design," *Technical Report 703*, MIT Artificial Intelligence Laboratory, 1983.
97. Rychener, M. (ed.), *Expert Systems for engineering design*. New York: Academic Press, 1988.
98. Schön, D.A., *The Reflective Practitioner*. New York: Basic Books, 1983.
99. Serbanati, L.D., *IEEE 9th International Conference of Software Engineering*, pp. 190-197, 1987.
100. Shina G.S., *Concurrent Engineering and Design for Manufacture of Electronics Products*. Van Nostrand Reinhold, 1991.
101. Siddall, J.N., *Optimal Engineering Design: Principles and Applications*. New York: Dekker, M., 1982.
102. Simon, H.A., "The Structure of Ill Structured Problems," *Artificial Intelligence*, Vol. 4, pp. 181-200, Reprinted in [21], pp. 145-165, 1973.
103. Simon, H.A., "Style in Design," In *Spatial Synthesis in Computer Aided Building Design. Eastman*, C.M. (ed.), John Wiley Sons, New York, 1975.
104. Simon, H.A., *Administrative Behavior*, 3rd Edition. New York: The Free Press, 1976.
105. Simon, H.A., *The Science of the Artificial*. Cambridge. MA: MIT Press, 1981.
106. Simon, H.A., *Models of Bounded Rationality*, Vol. 2. Cambridge, MA: MIT Press, 1982.
107. Simoudis, E. and Miller, J.S., "The Application of CBR to Help Desk Applications," *Proceeding of the 1991 Case-Based Reasoning Workshop*, Washington, DARPA, 1991.
108. Spillers, W.R. (ed.)., *Basic Questions of Design Theory*. Amsterdam: North-Holland, 1972.
109. Sriram, D. and Cheong, K., “Engineering Design Cycle: A Case Study and Implications for CAE,” In *Knowledge Aided Design*, Academic Press, New York, 1990.

110. Sriram, S., Stephanopouls, G., Logcher, R. et al., "Knowledge-Based System Applications in Engineering Design: Research at MIT," *AI Magazine*, Vol. 10 (3), pp. 79-96, 1989.
111. Steinberg, L.I., "Design as Refinement Plus Constraint Propagation: The VEXED Experience," *Proceedings of the Sixth National Conference on Artificial Intelligence*, pp. 830-835, 1987.
112. Suh, N.P., The Principles of Design. New York: Oxford University Press, 1990.
113. Suh, N.P., Bell, A.C. and Gossard, D.C., "On an Axiomatic Approach to Manufacturing and Manufacturing Systems," *Journal of Engineering for Industry*, Vol. 100 (5), pp. 127-130, 1978.
114. Swartout, W. and Balzer, R., "On the Inevitable Intertwining of Specification and Implementation," *COMM. ACM*, Vol. 25 (7), July, pp. 438-440, 1982.
115. Sycara, K. and Navinchandra, D., "Integrating Case-Based Reasoning and Qualitative Reasoning in Design," In Gero, J. (ed.). Ashurst, Computational Mechanics Publishing, 1989.
116. Tong, C., *Knowledge-Based Circuit Design*, PhD dissertation, Stanford University, 1986.
117. Tong, C. and Sriram, D. (eds*.), Artificial Intelligence in Engineering Design*. Boston, Mass: Academic Press, 1992.
118. Toulmin, S., *Human Understanding*. Princeton, NJ: Princeton University Press, 1972.
119. Ullman, D., Stauffer, L. and Dietterich, T., "Preliminary Results of Experimental Study of the Mechanical Design Process," *Technical Report 86-30-9*, Oregon State University, C.S. Dept., 1986.
120. Ulrich, K.T., "Computation and Pre-Parametric Design," *Technical Report 1043*, MIT Artificial Intelligence Laboratory, 1988.
121. Vollbracht, G. T., "The Time for CAEDM is Now," *Computer-Aided Engineering*, CAD/CAM section, Vol. 7 (Oct.), pp. 28, 1988.
122. Winston, P.H., et. al., "Learning Physical Descriptions From Functional Definitions, Examples and Precedents," *Memo 679*, MIT, Artificial Intelligence Laboratory, 1983.
123. Wood, K.L. and Antonsson, E.K., "Engineering Design-Computational Tools in the SYNTHESIS Domain," *The Study of the Design Process*, A Workshop, Ohio State University, February, pp. 8-10, 1987.
124. Yoshikawa, H., "General Design Theory and a CAD system," *Man-Machine Communications in CAD/CAM*, Proceedings, *IFIP W.G* 5.2, *Tokyo*, North-Holland, Amsterdam, pp. 35-38, 1982.
125. Konda, S., Monarch, I., Sargent, P., and Subrahmanian, E., "Shared Memory in Design: A Unifying Theme for Research and Practice," *Research in Engineering Design*, Vol. 4 (1), pp. 23-42, 1992.
126. Subrahmanian, E., Konda, S., Levy, S., Reich, Y., Westerberg, A., and Monarch, I., "Equations Aren't Enough: Informal Modeling in Design," *AI EDAM*, Vol. 7 (4), pp. 257-274, 1993.
127. Pinch, T. K., and Bijker, W. E., "Social Construction of Facts and Artifacts," In Bijker, W. E., Hughes, T. P., and Pinch, T. W., *Social Construction of Technological Systems*, MIT Press, Cambridge, 1989.
128. Sriram, D., Wong, A., and Logcher, R., "Shared Workspaces in Computer Aided Collaborative Product Development," Technical report, *Intelligent Engineering Systems Laboratory*, 1991.
129. Sargent, P. M., Subrahmanian, E., Downs, M., Greene, R., and Rishel, D., "Materials' Information and Conceptual Data Modeling," *Computerization and Networking of Materials Databases: Third Volume*, *ASTM STP 1140*, Thomas I. Barry and Keith W. Reynard, editors, American Society for Testing and Materials, Philadelphia, 1992.
130. Danko, G. and Printz, F., "A Historical Analysis of the Traditional Product Development Process as a Basis for an Alternative Process Model," *EDRC report*, Carnegie Mellon University, 1989.
131. Clark, K., and Fujimoto, T., *Product Development Performance*, Harvard Business Press, 1991.
132. Woodforde, J., *The Story of the Bicycle*, Routledge and Kegan Paul, London, 1970.
133. Westerberg, A., "Distributed and Collaborative Computer-Aided Environments in Process Engineering Design," *EDRC report*, Carnegie Mellon University, 1996.
134. Bjork, B. C., "Basic Structure of a Proposed Building Product Model," *Computer Aided Design*, Vol. 12 (2), 1988.
135. Eastman, C., Bond, A., and Chase, S., "A Formal Approach for Product Model Information," *Research in Engineering Design*, Vol. 2, pp. 65-80, 1991.
136. Zamanian, K., Fenves, S. J., and Gursoz, E., "Representing Spatial Abstractions of Constructed Facilities, *EDRC report*, Carnegie Mellon University, 1991.
137. Hauser, J. H., and Clausing, D., "The House of Quality," *Harvard Business Review*, pp. 63-73, May-June 1988.
138. Bloor, D., "Wittgenstein and Manheim on the Sociology of Mathematics," *Studies in the History and Philosophy of Science*, Vol. 4, pp. 173-191.

139. Mulkay, M. J., “Knowledge and Utility: Implications for the Sociology of Knowledge,” *Social Studies of Science*, Vol. 9, pp. 63-80, 1979.
140. Collins, H. M., “An Empirical Relativist Programme in the Sociology of Scientific Knowledge,” in *Science Observed: Perspectives on the Social Study of Science,* K. D. Knorr-Cetina and M. J. Mulkay, eds., Sage, Beverly-Hills, pp. 85-113, 1983.
141. Collins, H. M., “The Seven Sexes: A Study in the Sociology of a Phenomenon, or the Replication of Experiments in Physics,” *Sociology*, Vol. 9, pp. 205-224, 1975.
142. Collins, H. M., and Pinch, T. J., *Frames of Meaning: The Social Construction of Extraordinary Science*, Routledge and Kegan Paul, London, 1982.
143. DeMillo, R. A., Lipton, R. J., and Perlis, A. J., “Social Processes and Proofs of Theorems and Programs,” *Communications of the ACM*, Vol. 22, pp. 271-280, 1979.
144. Nam P. Suh, “Design and Operation of Large Systems”, *Journal of Manufacturing Systems*, Vol. 14, no. 3, pp 203-213, 1995.
145. Ertas, A. and Jones, J. C., *The Engineering Design Process*, John Wiley & Sons, New York, 1995.
146. Dasgupta, S., *Design Theory and Computer Science*, Cambridge University Press, 1994.
147. Ferguson, E. S., *Engineering and the Mind’s Eye*, MIT Press, Cambridge, MA, 1992.
148. Sipper, M. et. al., “A Phylogenetic, Ontogenetic, and Epigenetic View of Bio-Inspired Hardware Systems,” *IEEE Transactions on Evolutionary Programming*, Vol. 1, No. 1, pp. 83- 97, 1997.
149. Ramsay, A., *Formal Methods in Artificial Intelligence*, Cambridge University Press, 1988.
150. Maimon, O. and Horowitz, R., “Creative Design Methodology and the SIT Method,” Proceedings of DETCC97/DTM-3865, 9th International ASME Design Engineering Theory and Methodology Conference, 1997. Awarded Best Conference Paper.
151. Maimon, O. and Horowitz, R., “Sufficient Conditions for Design Inventions”, to appear in *IEEE Systems Man and Cybernetics*, 1998.