A UNIFIED THEORY OF ENGINEERING DESIGN

A Thesis

by

SCOTT PATRICK DYAS

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

December 2005

Major Subject: Mechanical Engineering

A UNIFIED THEORY OF ENGINEERING DESIGN

A Thesis

by

SCOTT PATRICK DYAS

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Approved by:

Chair of Committee,	Thomas R. Lalk
Committee Members,	Jo W. Howze
	William Schneider
Head of Department,	Dennis O'Neal

December 2005

Major Subject: Mechanical Engineering

ABSTRACT

A Unified Theory of Engineering Design. (December 2005)
Scott Patrick Dyas, B.S., Texas A&M University
Chair of Advisory Committee: Dr. Thomas R. Lalk

A theoretical model of design, that is universal and has a scientific basis, was developed. By doing so, it is believed that the practice of engineering design can be significantly improved. A better system of modeling designs is the missing ingredient that needs to be developed in order to improve the practice of design in the manner suggested above. Existing methodologies were reviewed to examine the current state of engineering design. This helped in developing a set of requirements for a new methodology. The potential for a scientific methodology to improve the practice of engineering design is also discussed. Developing a scientific theory of design, and showing that it meets these requirements was done to satisfy the objective. The theory takes the form of a conceptual model of design, which relates important aspects of the problem and the solution to facilitate a truly top-down hierarchical approach. A few examples are given to show how the methodology can be applied to real world design problems.

As a result, a theoretical framework for design was created as a part of this research project. The new methodology, termed UTED (Unified Theory of Engineering Design), addresses many important aspects of design which are overlooked by other methodologies. A set of rules was developed, to guide the designer through the design, and allow a more scientific process to be used. Making design more scientific increases the likelihood of achieving a successful design. The primary conclusions are that the development of a scientific theory of design can be created that makes design processes faster and more efficient, and improves the quality of designs produced, meaning there is a strong potential for such a methodology to have a positive impact on the field of engineering design.

DEDICATION

I would like to dedicate this thesis to my parents, Fred and Carol Dyas. Thank you for your constant encouragement and support, your love and friendship, and the excellent genes.

ACKNOWLEDGEMENTS

I would first like to thank my advisor Thomas R. Lalk. Without your inspiration and guidance this thesis would not have been possible. You are the only person with the knowledge, experience, and temperament to challenge me in the ways I needed to be challenged.

I would also like to thank William Schneider and Jo W. Howze, my other thesis committee members. Your questions and comments improved not just my thesis, but also the design methodology I developed. The two of you have also had a large and positive impact on my overall professional development.

There several other faculty members who have been involved in developing my methodology, and provided tremendous personal and professional guidance over the last four years. In alphabetical order: Charles Bollfrass, Charles Culp, Ted Hartwig, Ed Marotta, Make McDermott, Arun Srinivasa, Steve Suh.

Lastly, I would like to thank my students for allowing me to hone both my methodology and my skills as an engineer, manager, and teacher. I hope you have learned as much from me as I have from you.

TABLE OF CONTENTS

Page

Page

LIST OF FIGURES

1. INTRODUCTION

As engineers, we are always trying to find new and better ways of doing things. It is hoped that by improving design processes, we will be able to improve the quality of designs produced, just as improving manufacturing processes improves the quality of products. Engineering design has traditionally been considered an art rather than a science. Many human endeavors have been greatly improved by making their practice and study more scientific: agriculture, economics, manufacturing, etc. All of these subjects were considered art, but are now carried out in a scientific manner. Even chemistry, which is treated very scientifically today, used to be known as the "art of alchemy". Making design scientific should also make it easier to teach and to conduct design research, thus leading to improvements in its practice. Nam Suh, an MIT professor who is a strong advocate of making design more scientific posed the following question:

> **Can the field of design be more scientific?** The ultimate goal of axiomatic design is to establish a scientific basis for design and improve design activities by providing the designer with a theoretical foundation based on logical and rational thought processes and tools. The Goal of Axiomatic design is manifold: to make human designers more creative, to reduce the random search process, to minimize the iterative trial and error process, to determine the best design among those proposed, and to endow the computer with creative power through the creation of a scientific base for the design field. [1]

<u>What can be done to improve engineering design?</u>

1. Making the design process more scientific improves the likelihood of producing a successful design. [1]
2. Establishing a set of rules is a way to make design more scientific. [2]

This thesis follows the style of *Society of Automotive Engineers.*

3. Creating a formal method of describing and organizing information is necessary to apply rules. [2]
4. A model is a way to organize information about a design in a formal manner. [2]
5. Rules can be applied to the information in a model, and used during the construction of the model. [2]
6. Increasing the amount of information in a model increases the potential for rules to be developed and used, to deal with this new information, thereby improving the design. [2]
7. Developing a more formal and inclusive way to describe designs is a step in making a process more scientific. *From lines 2-6*
8. Therefore, by building better models, we may be able to achieve better designs. *From lines 1,7*

A better system of modeling designs is the missing ingredient that needs to be developed in order to improve the practice of design in the manner suggested above. **The objective of this project is to develop a theoretical model of design, which is universal and has a scientific basis.** It is believed that these two criteria will improve the quality of designs which are produced, and make the design process faster and more efficient.

Overview of the study

The next section of the thesis will begin with a background section and literature review to provide important information about scientific theory and discuss existing design methodologies. In fitting with the methodology proposed in this paper, the problem must be well defined before we can find a solution, so the third section of the paper lays out the requirements for the new methodology. Once we have established what we need from the new methodology, we can develop a conceptual model of designs, which can be used to identify and describe important aspects of the problem and solution. An introduction to the new methodology will be given to acquaint the reader with all of the basic concepts will be presented before we get into the details of how the methodology is used. The section on “Application of the Methodology” will go into greater depth, and illustrate a specific process, which allows the design to be dispatched in an efficient manner. A few brief examples will then be presented to give the reader a better understanding of how models can be used to improve designs. In the summary, we will examine whether the new methodology meets the requirements laid out in earlier sections of the paper, and lastly the conclusions will be presented.

2. BACKGROUND / LITERATURE REVIEW

This section of this thesis includes background information to introduce the general topic of design and specifically the need for a cohesive design methodology, that is, one theory or model that can be used for any design activity. The results of a literature review will follow. Existing methodologies are discussed in this section, in order to give the reader an idea of other work in the field.

2.1 MAKING DESIGN SCIENTIFIC

To develop a scientific theory of design, we must first discuss what makes a design methodology "scientific". In essence, an activity can be considered scientific if it is based on a set of rules which meet specific criteria. In order to make design more scientific, a set of rules or principles must be developed, which guide its practice in order to improve the results of the process.

Definition of scientific method

> Scientific method, the method employed in exact science and consisting of:
> (a) Careful and abundant observation and experiment.
> (b) Generalization of the results into formulated "Laws" and statements. [1]
>
> Basically, there are two quite different sets of criteria that must both be satisfied for a rule to even have a chance of being scientific. The first pertains to the properties of the rule itself, while the second has to do with the way that the rule is arrived at. In regard to the first type of criterion, here is a checklist of characteristics that tend to separate the scientific rules from the pretenders.
>
> - Explicit
> - Public
> - Reliable

---

[1] Definition of *scientific method* from dictionary.com

> - Objective
>
> The second criterion is based on the way in which the rule is arrived at. [2]

Essentially, Casti says that an answer must be based on a proper set of rules in order to be considered scientific. The criteria above accurately describe the way in which this methodology was developed.

2.2 MODELING

We want a generic model that can capture all of the important aspects of both the problem and the solution. In order to treat design scientifically, we must first be able to describe it scientifically.

What is a model? - A model shares common characteristics with the entity being modeled to allow prediction of what will happen in the future or explain what has happened in the past.

Example: In chemistry, we use theoretical models to predict what will happen if we mix two chemicals together. This allows us to calculate and predict what will happen before we actually do it.

<u>Using models to predict what will happen</u>

> In science, the chosen way to paint a picture of reality is to build a model, often expressed in the compact language of mathematics. We try to encode our experiences of the real world into the symbols and rules of mathematical formalism, and then make use of this formalism to generate predictions of what will transpire in the future. So from a scientific point of view, surprise can arise only as a consequence of models that are unfaithful to nature. [2]

Of course, we want to avoid surprise in engineering, because it often results in the failure of the design. The more complete, inclusive, and correct the model is, the less chance there is of being surprised. Because we can't make models perfect, engineers must decide which attributes of reality must be considered to achieve the desired results. A design methodology may only be a set of rules for developing (theoretical) models rather than developing designs, but better models lead to better designs. The model may be quite complex, so we are going to need a framework to organize and structure all of the information. Once it is organized, then we can see the relationships between the 'bits' of information in the model. We have to be able to define the problem and the solution and the relationship between them before we can develop a process for getting from the problem to the solution. The literature review in the next section will examine previous attempts at creating modeling designs and the design process.

2.3 LITERATURE REVIEW

The literature review is intended to give the reader an overall understanding of the current state of engineering design methodology. One way to describe the various methodologies is to group them into two categories: Bottom-Up and Top-Down (Systems Engineering) methodologies. A general discussion of these two categories is presented, rather than trying to address the innumerable variations of these methodologies that have been published. Axiomatic Design will be discussed separately because, while nominally a top-down methodology, it approaches the subject of design in a different manner.

2.3.1 Bottom-Up Design Methods

Traditional engineering design methods are based on a Bottom-up approach. Beginning with a set of known elements, design engineers create the product or system by a combination of system elements. However, it is unlikely that the Functional need will be met on the first attempt unless the system is simple. After determining the product's performance and deviation from what is required, the elements and their combination are altered and the performance determined again. The Bottom-up process is iterative, with the number of iterations (and

> design process efficiency) determined by the experience and creativity of the designer, and the complexity of the product or system. [3]

Formal Bottom-up design methodologies, commonly referred to as detailed design methodologies, usually focus on improving the performance of individual system elements, rather than the performance of the system as a whole. Failures often occur at the interfaces, because there is no systematic process to ensure compatibility between the elements of the system. Like other methodologies, they may have a set of rules or guidelines associated with them, but these are usually limited in scope or in the types of designs to which they apply. A new methodology is needed, which is more inclusive, and applies to any engineering design.

2.3.2 Systems Engineering Methodologies

Systems engineering methodologies are more directed, and based on a top-down approach to design. The key idea is that large complicated problems can be reduced to a set of smaller problems that are easier to solve. Requirements at the interfaces ensure that the solutions to these smaller problems will form a coherent solution. While some iteration may inevitably occur, there should be less than is typical for bottom up methodologies. Systems Engineering also considers the entire life cycle of the design. Manufacturing, sale, distribution, service, maintenance, and finally disposal must be considered in addition to the actual use of the product.

> In the top-down approach, the requirements are always satisfied through every step of the design process because it is an inherent part of the methodology, whereas in the bottom-up approach the methodology provides no assurance that the requirements are always satisfied. [3]

Two main characteristics of the top down process

1. First, the process is applicable to any part of the system. Starting with the system as a whole, repeated application of this process will result in decomposition of the system into smaller and smaller elements.
2. Second, the process is self-consistent. External properties of the total system, as described by the inputs and outputs and relations between parts, must be reproduced by the external properties of the set of interacting elements, because they have been developed from the total system and are traceable back to the top. [3]

The Systems Engineering Design Process

1. Need analysis
2. Functional decomposition
3. Conceptual design
4. Evaluation and downselection
5. Preliminary design
6. Final design [3]

Coordinating the breakdown of the problem into smaller problems and ensuring the compatibility of the solutions requires a great deal of work. Defining system integration up front, rather than after the system elements are designed can greatly reduce the amount of iteration that is required. The goal is to improve the design process and the system being designed, by improving the integration process rather than trying to improve the performance of individual elements in the system. Small Systems are generally easier to integrate than large ones, because there are fewer interfaces, and therefore systems design methodologies are most beneficial when the design is very large and complex. Systems Engineering Methodologies usually have a detailed way to describe the problem, but lack a systematic way to describe the solution. A proper description of both is needed in order to relate the two to each other.

By using a systems engineering approach the relationships and interactions between the system elements can be better determined, understood and appreciated. The top-level need and requirements of the problem can be more easily identified, leading to an overall solution rather than finding independent (and possibly incompatible) solutions for each individual requirement. Coordination of the efforts by the various elements of the system will improve efficiency and effectiveness, and allow unnecessary or redundant functions to be eliminated.

The use of a hierarchical structure allows the designer to focus on small segments of the design at a time, without having to worry whether decisions they make will affect other portions of the system. Breaking down the problem systematically allows the designer to repeatedly narrow the scope of their focus, reducing the amount of information that has to be dealt with at one time, while still ensuring a cohesive design. It also allows design work to be done in parallel by multiple designers for these same reasons, making it more conducive to teaming and management of the process. The use of hierarchy also gives traceability to the requirements, and makes documentation easier.

While system design methodologies examine the hierarchy of requirements within the system, they neglect the hierarchy of the parameters that determine whether performance requirements are met. The systems methodology doesn’t force the compatibility of requirements between levels in the hierarchy. In order to do simultaneous design of subsystems, these details need to be determined early in the design process or increased iteration may be the result.

According to Blanchard and Fabrycky, systems engineering (top-down) is not going to completely replace bottom-up methods. The book shows several different process models, but these really describe phases of the design rather than a particular process [3]. The UTED methodology allows the design to be carried out in a top down fashion throughout all levels in the hierarchy rather than having to switch part way through as is suggested in the Systems Engineering Methodology. They are correct in stating that their systems engineering process can't really be done in a top down fashion throughout the entire process, because you can't break Functional Requirements down to their lowest levels without specifying how the higher-level ones will be fulfilled. The Systems Engineering methodology doesn't recognize the solution domain, much less the fact that it is hierarchical in nature. Only the very top-level performance requirements can be specified at the beginning using the Systems Engineering process. The Systems Engineering only applies Performance Requirements to the bottom level FRs.

2.3.3 Axiomatic Design

Axiomatic Design is a methodology, which is claimed to be universal and to combine both system level and detailed design methodologies into one system. The premise behind Axiomatic Design is that it allows designs to be analyzed, evaluated, and improved through the use of a mathematical framework and a set of general rules (axioms) which guide the designer in making design decisions. The framework is also supposed to help the designer identify and address key aspects of the design, which will determine whether the needs are met. The design process takes place in several "domains", and by mapping between the domains we get from "what we want to achieve" to "how the needs are met". The framework is used to relate the information in the different domains. Suh asserts that his framework can be used for large complex systems, but this has yet to be demonstrated. All of the examples in his book are simple, easy to solve problems. The framework appears incomplete when you get beyond these simplistic types of examples.

Two perceived shortcomings associated with the Axiomatic Design methodology, in its current form, limit its use on systems that are complex or have several layers in their function structures. The first is failure to differentiate between the Concept Domain and Quantitative Domain. While some efforts are made to ensure the compatibility and cohesiveness of requirements within each layer of the function structure, and between requirements at different levels in the structure, this cannot be fully achieved unless the conceptual and quantitative aspects of the design are separated.

While trying to apply axiomatic design principles it was found that, unless the conceptual and quantitative aspects of the design are separated two major problems occur:

1. The compatibility and cohesiveness of requirements within each layer of the function structure can't be assured
2. The compatibility and cohesiveness of requirements at different levels in the structure can't be assured.

The second issue is that there isn't any need for a distinct process domain. When you develop a method of producing a product, you are really designing a manufacturing or assembly process. This is a new design to which the other domains apply. Of course the design of the system and the design of the production methods are obviously tied together, but the two can't be fully related in a useful manner unless the creation of manufacturing processes is considered a design itself. If the design of the product and the design of the process are linked properly, then they can be created simultaneously in parallel.

The process discussed in the book is relatively loose and based on zig-zagging in order to simultaneously decompose the problem and the solution in a top down fashion (A process called Parallel Decomposition). It doesn't appear that Axiomatic Design in the form described in Suh's book can be done in a fully top down manner, because the conceptual and quantitative domains are not clearly defined.

3. REQUIREMENTS FOR THE NEW METHODOLOGY

The goals of other design methodologies and their shortcomings were a source of inspiration in the development of requirements for the new methodology. Suh established the need for a scientific basis for design, but his theoretical model, known as Axiomatic Design, is incomplete and hasn't been shown to be a universal and scientific basis for engineering design. The framework used in axiomatic design along with those used in other methodologies, are not a satisfactory basis for engineering design.

At the highest level, we are trying to improve the quality of the designs that are produced, and to make the design process faster and more efficient. More specifically, a unified theoretical basis or model is needed to make design more scientific. The term "unified" implies that the methodology can be used for any design-related activity, from the initial formulation of the problem through production and testing. The methodology must be applicable throughout the entire design process, and work for large complex system designs as well as small simple designs. The process must foster innovation, improve our ability to predict the behavior of the system, and reduce the iteration and effort that is required to produce a successful design. The improvements above will be realized in the new methodology by making design more scientific.

4. CONCEPTUAL MODEL FOR THE UNIFIED THEORY OF ENGINEERING DESIGN (UTED)

Previous sections of this thesis discuss the current state of engineering and the need for a new methodology. At this point we begin the new material, which was developed as a part of this research. The first topic presented will be the conceptual basis or model for the methodology, followed by a discussion of how the methodology is applied and examples in later sections. This methodology allows a more complete picture of the design to be created, compared to other methodologies.

A set of overlapping Design Domains are the starting point for the conceptual model. Each overlapping region of these domains is associated with a Defining Design Characteristics, which can be used to define and describe the important aspects of the design. These are important attributes of reality (the problem and the design), that are incorporated into the model. Hierarchy is an integral part of the methodology, and so basic hierarchical theory will be presented as well.. A brief discussion of processes will be made to show some of advantages the UTED methodology's conceptual framework provides over other methodologies.

The viewpoint of large multi-team designs inspired much of the methodology. Generally, any process that works for complex systems will work for more simple ones, but processes that work for simple systems don't always work for the complex ones. The rationale for each important aspect of the methodology will be discussed throughout this section of the paper, as each topic is addressed.

4.1 DESIGN DOMAINS

Like some other design methodologies, the concept of domains is used to explain the theory behind this methodology. Figure 1 shows the four design domains. Part of any engineering problem is conceptual and part of it is quantitative. The solution will have both conceptual and quantitative aspects as well. This means that there is some overlap between the domains as seen in figure1. The Concept Domain deals with Functional Requirements and the Design Features, which are the conceptual or non-quantifiable aspects of the design. The Quantitative Domain is concerned with quantifiable characteristics of the design including Performance Requirements, Design Parameters, and Quantitative Constraints, which are a special type of Performance Requirement. Definitions for the Domains are listed below.

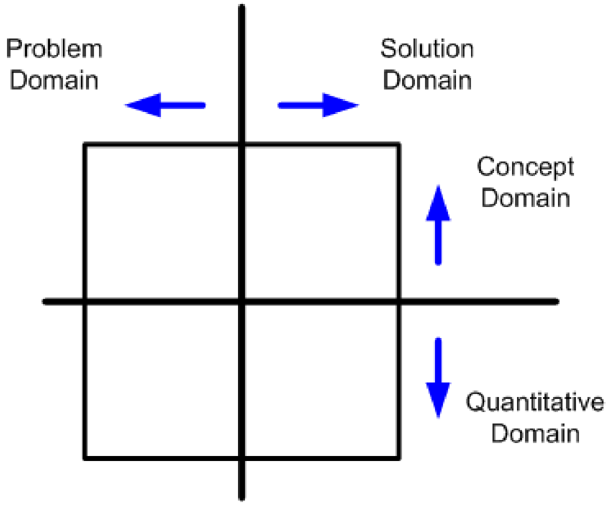


Figure 1: Design Domains

Problem Domain – A formal description of the problem in a solution independent manner, based on the Customer Requirements

Solution Domain – A formal description of the solution, which describes how requirements in the Problem Domain are met

Concept Domain – A conceptual description of the problem and solution

Quantitative Domain – A quantitative description of the problem and solution

4.2 DESIGN CHARACTERISTICS

There is a Design Characteristic associated with each overlapping section of the domains. Design Characteristics are developed based on the Customer Requirements. The Customer Requirements are developed as a part of the Need Analysis, and will be discussed further in sections 5.1 and 5.6. The other Design Characteristics will be explained in more detail in sections 5.2 through 5.5. The UTED cognitive map is shown in Figure 2, with the definitions listed below it.

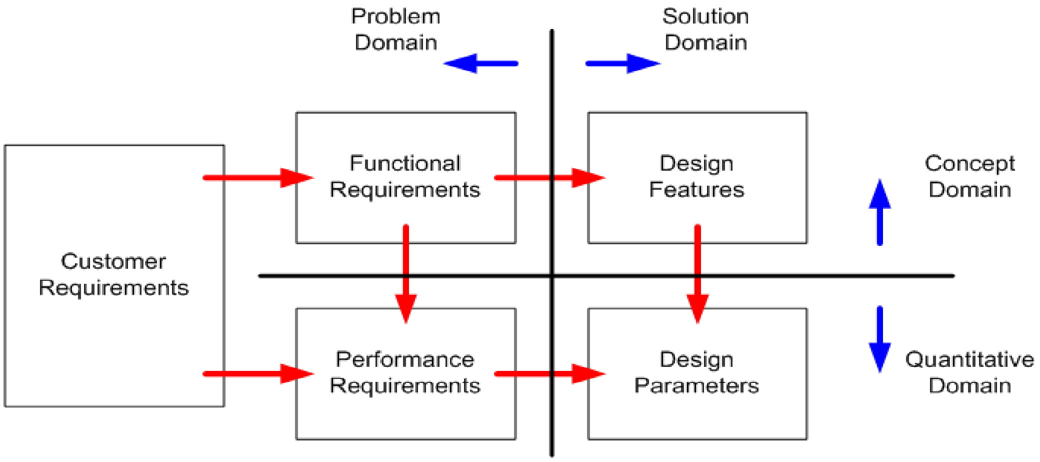


Figure 2: UTED Cognitive Map

Customer Requirements (CR) – Needs that must be met for the design to be successful, according to the customer's desires, and stated in the designer's own words.

Functional Requirements (FR) - The critical functions or tasks that must be carried out in order for the design to be successful, forming a conceptual description of the problem.

Performance Requirements (PR) – The variables which define how well the Functional Requirement must be carried out in quantifiable terms, to create a quantitative description of the problem.

Design Features (DF) - The actual physical elements in the design that carry out or facilitate the Functional Requirements, which are labeled to create a conceptual description of the solution.

Design Parameters (DP) - Quantifiable characteristics of Design Features (or environmental and operating conditions), which determine whether the Performance Requirements are met, providing a quantitative description of the solution

4.3 HIERARCHY

As with any systems based methodology, the design problem is broken down or decomposed in a hierarchical manner. All of the defining Design Characteristics (FRs, PRs, DFs, and DPs) in the framework apply at each level in the model. There may be more than one way to decompose the problem. There may be advantages or disadvantages to each of these methods, depending on the nature of the problem and the team that is trying to solve it. As long as none of the rules are broken, the arrangement of the hierarchy is generally up to the designer. There isn’t a single “right” answer, but there are definitely “wrong” answers. An answer becomes wrong when it violates one of the rules. The hierarchy should be arranged so as to minimize the complexity of the model. There is always a way to make the description of the problem more complex, but not always a way to simplify it [2]. In general, we’d like to achieve the least complex model possible while still maintaining the necessary completeness.

A complex system consists of a large group of individual components. All of the requirements could be placed at the same level, so that all of the Design Parameters would be leaves, and all of the Design Features would be components. Components often work together to fulfill a particular high-level function, so we group these together and refer to them collectively as a subsystem. By specifying the collective requirements of the subsystem and treating it as a “black box”, less information is needed to define the design at each level in the hierarchy. If a top down approach is used, and we proceed through the design one level at a time, then we reduce the amount of information that the designer has to be keep track of at one time.

The defining design characteristics (FR, PR, DF, and DP) must be consistent within each domain and level in hierarchical decompositions, between domains, and between levels. The problem and the solution should be decomposed in a top-down manner in order to ensure compatibility between these two domains. In section 7, a method of displaying the design characteristics, called Design Matrices, display the information in a manner which allows coupling or contradictions to be more easily identified during the decomposition process. Conflicts should be identified as early as possible, to reduce the amount of iteration that is required.

5. APPLICATION OF THE METHODOLOGY

The previous section presented the basic concepts behind the methodology. This section will cover the concepts in greater detail, and establish a set of rules for using these concepts. The customer requirements and defining design characteristics will be explained, to show how they can be best incorporated into the model. Lastly, a design process that exploits the potential of the new model will be presented.

5.1 CUSTOMER REQUIREMENTS

Definition - Needs that must be met for the design to be successful, according to the customer's desires, and stated in the designer's own words.

Any design process should begin by establishing the Customer Requirements, by sitting down with the customer to establish exactly what is desired. The Customer Requirements are a concise statement of the customer's needs as understood by the designer. In some cases the designer may be the "customer", so they must look at what it is they are trying to accomplish. Initially, the customer doesn't usually have a well-defined idea of what they want, and so, clearly defining the needs of the customer is usually not a trivial task. The designer then converts this information into a set of Customer Requirements that are structured and phrased in a much more formal manner. Some Customer Requirements are "needs", and some are "wants". The designer must determine, and specify which requirements are "needs" and which are "wants". There are many different methods of developing a set of customer requirements. ***It should be emphasized again that the Customer Requirements aren't necessarily what the customer says the requirements are, but are the engineer's interpretation of those requirements. The Customer Requirements are statements of the true need in the designer's own words.*** It is a negotiated set of requirements which the customer and the designer have agreed upon. More detailed processes for developing a good set of Customer Requirements will be developed as a part of future research.

5.1.1 General Application of Customer Requirements

There is a difference between Customer Requirements, Functional Requirements, and Performance Requirements. It is the job of the design engineer to convert the Customer Requirements into an appropriate set of Functional and Performance Requirements. Functional and Performance Requirements are a much more formal in nature. Some Customer Requirements may be converted directly into Functional Requirements or Performance Requirements, as long as they meet all of the rules for these design attributes. Intellectual property or technology issues can limit which solutions will be acceptable, so Customer Requirements may constrain which Design Features can be used to carry out the Functional Requirements.

5.1.2 Customer Requirement Rules

Customer Requirements may consist of:

- Tasks that the design must perform.
- Required performance the design must have.
- Any important qualities, attributes, or characteristics that the customer requires or considers desirable for the design to have.
- Legal or social requirements, such as “comply with OSHA standards”. Details of the regulations can be described more explicitly in the Functional Requirements or Performance Requirements.
- Any other constraints that the customer whishes to place on the design.

5.1.3 Common Mistakes Designers Make with Customer Requirements

The most common mistakes made by designers regarding Customer Requirements include not specifying all of the necessary requirements, listing unnecessary requirements which over constrain the design, and listing incorrect customer requirements (when the real problem or issue has not been correctly identified).

5.1.4 Additional Customer Requirement Guidelines

When establishing the Customer Requirements avoid placing unnecessary constraints that are placed on the design. Challenge the customer so you can determine which ones are absolutely required, and which ones are negotiable.

5.2 FUNCTIONAL REQUIREMENTS

Definition - The critical functions or tasks that must be carried out in order for the design to be successful, forming a conceptual description of the problem.

The Customer Requirements usually lead to a set of Functional Requirements and Performance Requirements. It is important to remember that Functional Requirements are concerned with what must be done, rather than how. Tasks can be active or passive, meaning they can either cause or prevent a particular action. A good Need Statement should reflect the top level Functional Requirement.

It is not uncommon to find functions arranged in a hierarchical fashion in other methodologies. A Function Structure is the most common way to organize and communicate the Functional Requirements, and is helpful when decomposing the problem in a hierarchical manner. The Function Structure also aids in identifying the Functional Requirements. Figure 3 shows a simple generic Function Structure. FR0 represents the top-level function. FR1, FR2, and FR3 are Functional Requirements at the next lower level. The lowest level FRs are referred to as ‘leaves’, when they can’t or won’t be broken down further. If a function can be satisfied by an off the shelf component or system, then there isn’t any need to break down the functions beyond that point.

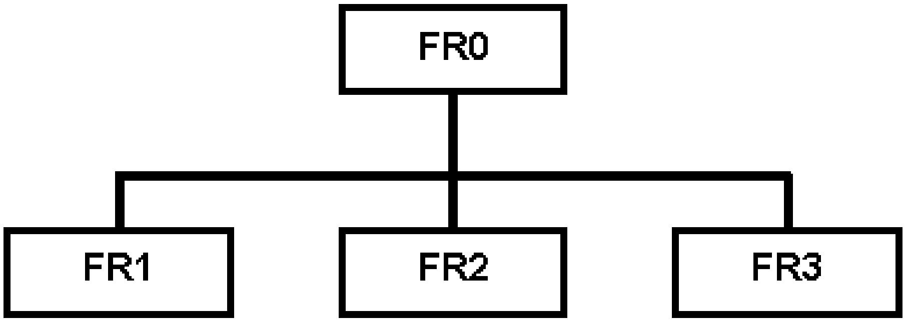


Figure 3: Function Structure

If you are designing a portion of a system, then you should begin your Function Structure at least one level above the part of the system you are designing. This is a useful step to help the designer understand how their design relates to the overall system. A compatible set of requirements should lead to a cohesive design.

5.2.1 General Application of Functional Requirements

There may be more than one Functional Requirement that can be used to satisfy a Customer Need. By selecting the Functional Requirement, you can choose different paths leading to different concepts. You should start with a general FR at the beginning, and then make it more specific, to develop different methods of solving the problem. Functional Requirements generally evolve and become more specific as the design becomes better acquainted with the intricacies of the design at each level.

Table 1: Common Functional Requirements with Example Design Features.

Functional Requirements	Design Features
Energy storage	
Store chemical energy	Fuel tank, battery
Store electrical energy	Capacitor, inductor
Store potential energy (mechanical)	Spring, lifting a weight
Store kinetic energy (mechanical)	Flywheel
Energy conversion	
Chemical to thermal	Combustor
Chemical to electrical	Battery, fuel cell
Electrical to thermal	Resistor, heating coil
Electrical to mechanical	Electric motor
Thermal to mechanical	Cylinder and piston, turbine
Mechanical transformer	
Rotational motion to rotational motion	Gearbox
Rotational motion to (Tangential) linear motion	A wheel on a vehicle
Rotational motion to (Radial) linear motion	A lobe on a camshaft
Convert linear motion to linear motion	Pulley
Mechanical displacement to volume displacement	Hydraulic pump
Gyrators (bond graph terminology)	
Convert motion into a force	Shock absorber, Air foil
Convert flow into a torque	Turbine
Constraining Functions	
Constrain motion	Bearing, fastener
Constrain energy flow	Insulation

5.2.2 Functional Requirement Rules

Below is a set of rules for Functional Requirements developed as a part of the UTED methodology to help designers better decompose design problems and thereby improve the solutions they create.

1. Placing Functional Requirements on the incorrect level in the hierarchy.
2. The Functional Requirement must be stated as an objective task.
3. The Functions on the same level of the hierarchy must represent independent tasks.
4. Low level Functional Requirements must be functions that are necessary for a higher-level function to be carried out.

5.2.3 Common Mistakes Designers Make with Functional Requirements

Designers often select Functional Requirements, which aren't tasks. Often times, these requirements are proper Customer Requirements, but don't fit the definition for Functional Requirements. Below are some examples of possible Customer Requirements, which are not tasks and therefore can't be Functional Requirements. They are important qualities or attributes that may be desirable for the design to have, but they are not Functional Requirements or Performance Requirements. Some of the attributes, such as "safe" or "reliable", may lead to a set of Performance Requirements that will be applied to the system, but an additional Function Requirement may, or may not be required in order to make the design fit the attribute. Designers may come up with a Functional Requirement such as "be safe". If there isn't a specific Design Feature in the system that makes it safe, then this isn't a true Functional Requirement. Safety is much more likely to be a Customer Requirement, which can be turned into a Performance Requirement (or Quantitative Constraint) which determines the safety in a quantitative manner. The customer wants the design to be safe, although there may not be anything actively done by the system to make itself safe. Below is a list of types of common Customer Requirements:

- Reliable
- Cost-effective
- Safe
- Foldable
- Flexible

- Ergonomic
- Attractive
- Reconfigurable
- Portable

5.3 PERFORMANCE REQUIREMENTS

Definition - The variables which define how well the Functional Requirement must be carried out in quantifiable terms, to create a quantitative description of the problem.

Performance requirements are not a new concept in engineering design, but the way that they are related to Design Parameters in this methodology is unique. This is the only know methodology which recognizes hierarchy in the Quantitative Domain. It is important to separate Functional Requirements and Performance Requirements, but many methodologies don't make a distinction between the two [Suh]. When a product is re-designed, the Performance Requirements are often changed while the Functional Requirements remain the same.

Figure 4 shows the hierarchy, and the interdependence of the variables (PRs and DPs), but it can become a very large and difficult to read chart. PR0a is dependent on DP1 and DP2 in order to be satisfied. Another way of saying it would be that values for DP1 and DP2 are chosen, which allow PR0a to be satisfied. DP1 becomes PR1a, which is dependent on DP5 and DP6, etc.

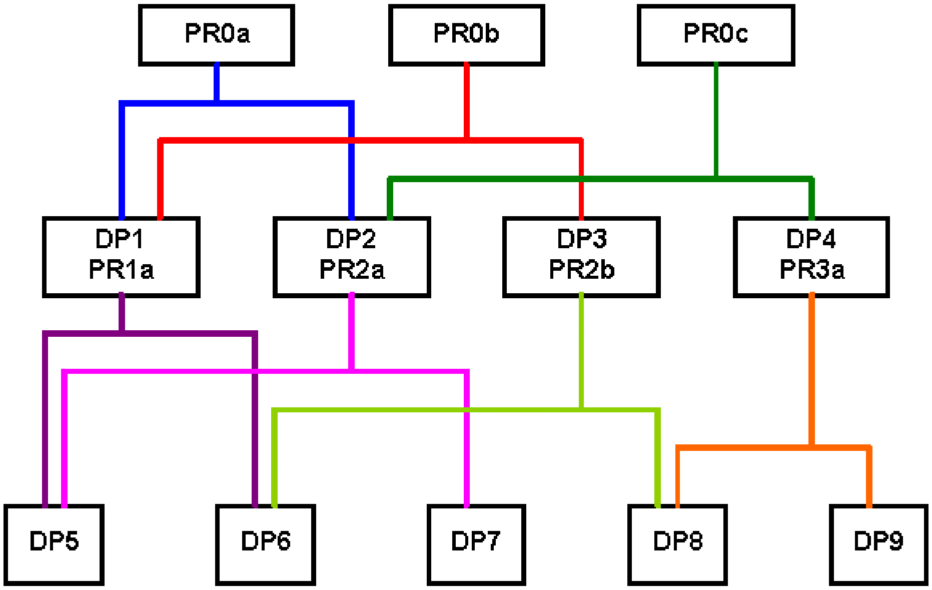


Figure 4: Performance Requirement / Design Parameter Hierarchy

5.3.1 General Application of Performance Requirements

Performance Requirements consist of a *variable*, a *value*, and a set of *operating conditions.* The *variable* should have a required *value* when operating under a particular set of *operating conditions*, or over a range of *operating conditions*. The required *value* may also be expressed in terms of the *operating conditions* (i.e. 20 degrees C above ambient temperature). The *operating conditions* are often not listed when the required *value* is the same for all *operating conditions*. Performance Requirements may have continuous or discreet values.

Achieving independence among the Design Parameters is important when we begin to optimize the system. If certain types of independence are not maintained, then contradictions will begin to appear in the hierarchy. For example, an electric motor may have a performance requirement for power. The speed at which the motor spins may be important as well, because it affects other components in the system. We can’t have both power and speed as performance requirements for the motor, because the power depends on the speed. So instead, we choose our two Performance Requirements to be speed and torque, which give us all of the necessary information we need to determine the power and the speed of the motor, while still being an independent set of parameters.

Specification of Performance Requirements can involve equalities or inequalities (=, <, >, ≠, ≤, ≥). Performance Requirements for manufacturing tolerances are often specified as a range or maximum deviation from the chosen value. There may be one or more Performance Requirements associated with each Functional Requirement.

Often times, it may be difficult for the designer to come up with Performance Requirements if the value of the requirement is not stringent or is trivial to satisfy. All of the methods to solve the problem that may immediately come to mind may already meet the Performance Requirements. Performance Requirements are mathematically dependent on the Design Parameters.

5.3.2 Performance Requirement Rules

Below is a set of rules for Performance Requirements developed as a part of the UTED methodology to help designers better quantify design problems and thereby improve the solutions they create.

1. Performance Requirements must be quantifiable
2. Performance Requirements on the same level must be independent of each other.
3. It is permissible to have more than one Performance Requirement associated with each Functional Requirement.
4. Two Performance Requirements at the same level in the hierarchy can be coupled through common Design Parameters, but they should not be functions of each other.

5.3.3 Special Cases of Performance Requirements

Unattached Performance Requirements (Quantitative Constraints)

Mass, cost, and dimensions are examples of common constraints that are placed on design. In the UTED methodology, they are called un-attached Performance Requirements because they are not directly related to the function that is being performed by the system in question. Constraints stem from a higher-level requirement, which is dependent on the performance of the system. The mass of an energy storage device such as a battery doesn’t really have an effect on the ability of the battery to store energy. The mass is a result of selecting values for design parameters, some of which will affect the amount of energy the battery can store. The battery’s mass may affect the overall system’s performance, and thus must be considered as a requirement at the battery’s level in the hierarchy, except it is tied to a specific Design Feature rather than a Functional Requirement. Quantitative Constraints can propagate down through the hierarchy, like system mass requirements are based on the masses of the subsystems.

5.3.4 Common Mistakes Designers Make with Performance Requirements

Common mistakes that designers make with Performance Requirements include selecting a non-independent set of Performance Requirements, selecting Performance Requirements which aren’t quantifiable, and matching Performance Requirements with the wrong Functional Requirement

5.3.5 Additional Guidelines for Performance Requirements

If a different Design Feature is used, then a different set of Design Parameters will likely determine whether the Performance Requirements are met. Performance Requirements consist of a variable, a value, and a set of operating conditions. Changing Performance Requirements or adding new ones at a particular level can create the need for additional Functional Requirements at lower levels, or can necessitate a change in the Design Feature that is use to satisfy the Functional Requirement.

5.4 DESIGN FEATURES

Definition - The actual physical elements in the design that carry out or facilitate the Functional Requirements, which are labeled to create a conceptual description of the solution.

Design Features are physical elements of the system. They are identified (named) in the model so they can be described in greater detail (quantitatively and hierarchically). They are linked to the functions they perform. It should also be noted that stored information (i.e. software or data tables) can be considered Design Features. Design Features are a concept that isn't present in any of the other methodologies that were examined in this study. While some methodologies talk about the features, they don't specifically label them, and they don't include them in their conceptual framework. When other methodologies actually do use the term "feature", it has a different meaning. They are using the word "feature" to describe attributes of the system rather than parts of the system. This is the only known methodology that recognizes there is a hierarchy to the Design Features.

The Design Features have a hierarchy that breaks down systems into subsystems, components, and eventually into individual features on each component. The Design Feature Tree provides a way to display the solution in a hierarchical manner. Each DF corresponds with a particular FR at the same level. See Figure 5.

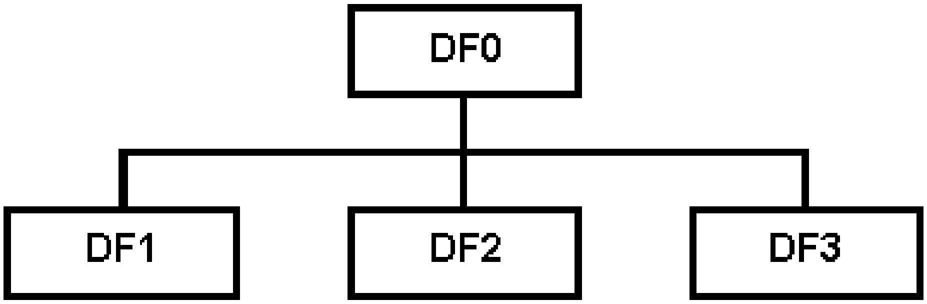


Figure 5: Design Feature Tree

5.4.1 General Application of Design Features

A Design Feature may be a system, a subsystem, a component, or a portion of a component. Depending on the level of the Functional Requirement in the hierarchy, the associated Design Feature could be an automobile, an engine, or threads on a bolt. It is possible for a single component to have multiple Design Features, with each satisfying a different Functional Requirement.

Different Design Features will have different sets of Design Parameters that will determine whether the Performance Requirements are met. Different Design Features will also have different sets of lower level Functional Requirements that must be satisfied in order to meet the overall requirements.

Bond graph terminology can be used with many design features. This is very convenient for energy based systems, because each element in the bond graph represents a design feature. Many Design Features act as transformers or gyrators, and so they. [See list of Common Functional Requirements and Design Features in the section 5.2.1]

5.4.2 Design Feature Rules

Below is a set of rules for Design Features developed as a part of the UTED methodology to help designers better describe the solutions they create.

1. A Design Feature must be identifiable as a noun.
2. A Design Feature can be a System, a Subsystem, a Component, or a portion of a Component.
3. Sub-features must be a portion of the higher level Design Feature.
4. Each Design Feature must be traceable to a specific functional requirement that it satisfies (see repeated features in section 5.4.3)

5.4.3 Special Cases of Design Features

Repeated Design Features

In some cases repeated Design Features are used to fulfill a single Functional Requirement. Using symmetry in the positioning of these Design Features is going to make the process simpler. When using repeated Design Features, the quantity used must also be a Design Parameter. If the repeated Design Features are placed asymmetrically, then additional Design Parameters to determine the location of each Design Feature will be required. A designer might want to distribute repeated Design Features asymmetrically to match the asymmetry of loads or other requirements. Examples of repeated Design Features are listed below:

1. Multiple bolts on a flange head
2. Columns supporting a structure
3. Multiple bricks in a wall
4. Multiple cylinders in an internal combustion engine
5. Individual cells within a fuel cell power unit

Reconfigurable Design Features

Some Design Features may be reconfigurable in order to meet different requirements, or operate in different conditions. When systems are reconfigured, it sometimes means switching between lower level features, which have repeated features with different values. In a multi-speed (stepped) gearbox, a different ratio may be required at different times during the system’s operation. Different Design Features (gears) with different Design Parameters (radii) are engaged when the Performance Requirement (drive ratio) of the system (gearbox) change over time. The different gear pairs serve the same function, but they carry out the function at different times. Engaging or disengaging each feature also becomes a new function. Besides the extra gear pairs, there is an additional Design Feature or Design Features (like a clutch or some other type of coupling) or that must be added in order to allow switching between the features (gearsets). In a continuously variable transmission, some of the Design Parameters are able to change with time. The Design Feature doesn't change, as was the case in the stepped transmission. Rather than changing between different DFs (gearsets) to achieve changes in DPs, DFs with variable DPs are used [See Design Parameters in section 5.5].

5.4.4 Common Mistakes Designers Make with Design Features

Designers tend to make mistakes in defining the requirements more often than they do in defining the solution, because engineers tend to be solution oriented. Selecting Design Features is often a check on the Functional Requirements. A common mistake is to choose a Design Feature which isn't a noun. Often times, a designer will try to list some attribute of the design, which is not actually a noun. Sometimes, theses attributes are actually the Design Parameters that describe the Design Feature. If a specific Design Feature, which carries out a particular Functional Requirement can't be identified, then it may not be a true Functional Requirement. [See common mistakes for FRs, section 5.2.2]

5.5 DESIGN PARAMETERS

Definition - Quantifiable characteristics of Design Features or operating conditions, which determine whether the Performance Requirements are met, to provide a quantitative description of the solution.

It is important to remember that Design Parameters must be quantifiable. They may have discrete or continuous values. These are physical attributes of the system that are specified while the system is being designed. When broken down to the very lowest level, they consist of properties and dimensions. At higher levels, they don't necessarily fall into one of these two categories. In many cases the Design Parameters at one level become the Performance Requirements associated with the Functional Requirements at the next lower level. [See Design Parameter / Performance Requirement hierarchy in Figure 4, section 5.3]

5.5.1 General Application of Design Parameters

It is important that Design Parameters must be independent. Factors of safety and tolerance stack-up can also be included as a part of the parameter analysis. Each Design Feature may have more than one Design Parameter associated with it. For a given design, any number of parameters could be defined, which characterize some aspect of the final system. The key is to find which ones have an important effect on the system performance. Often, a Design Parameter can be broken down into multiple subparameters. The mass of an object can be broken down in terms of the object's physical dimensions, and material properties. Parameters must be broken down to the point where they can be specified independently. If an off the shelf component is going to be selected, only the required macroscopic properties of the component or system need to be determined. There isn't any need to break down the lower level requirements of the off the shelf component or system unless it is going to be modified.

If every Design Parameter was broken down to its very lowest level, there could easily be an unmanageable number of parameters that the designer would have to concern themselves with.

Whenever engineers design, they use a simplified model (often theoretical or analytical) of the system to predict how it will perform. Simplifying assumptions are made to reduce the amount of information that is needed to calculate the performance. There is no way to take *everything* into account. It is up to the designer to decide which approximations are appropriate.

Like Performance Requirements, Design Parameters may have continuous or discreet values. On engineering drawings, each dimension that is specified should correspond with a Design Parameter. When the Design Parameters are finalized, they become Design Specifications.

5.5.2 Design Parameter Rules

Below is a set of rules for Design Parameters, developed as a part of the UTED methodology to help designers better quantify design problems and thereby improve the solutions they create.

1. Design Parameters on the same level must be independent of each other. Two parameters may both be a function of a common lower level Design Parameter, but they must not be functions of each other.
2. Design Parameters must be quantifiable. They can have continuous or discreet values.
3. Design Parameters must describe a quantifiable characteristic of a Design Feature unless they are Environmental parameters or Operating/Control Parameters (see below).

5.5.3 Special Cases of Design Parameters

Splitting of Design Parameters

Some Performance Requirement may be based on the power generated by a motor, leading to the selection of the motor's power as a Design Parameter. Let's say the designer discovers that another Performance Requirement depends on the motor's rotating speed. The Design Parameter "power" can be split into two parameters, torque and speed. Because the Design Parameters must be independent, speed and power shouldn't both be used as parameters on the same level.

Reparameterization

A designer may find that using a different set of parameters may make the problem easier to solve. By switching from Polar to Cartesian coordinate systems, a distance and an angle can be used to quantify a design rather than a pair of distances. This is useful when the performance of a system is more directly quantifiable in terms of one set of independent parameters than with another.

Special types of Design Parameters

Environmental Design Parameters – These are parameters that will affect the performance of the system, but the designer can't control. By writing them down, we are trying to anticipate the values, rather than specifying them, in order to predict the system's performance.

Operational/Control Design Parameters - These are parameters that can be varied during the operation of the system to achieve the desired performance. These may come in the form of inputs to the system, which can be specified by the designer or operator. These can serve to add an extra degree of freedom, and may be varied in order to help achieve the performance requirements. Boolean variables with yes/no or true/false can be used as Control Parameters, because they can be redefined as numbers i.e. 1/0. In this way they can be made quantifiable to meet the definition for Design Parameters.

5.5.4 Common Mistakes with Design Parameters

The mistakes commonly made with Design Parameters are similar to those made with Performance Requirements.

5.5.5 Additional Guidelines for Design Parameters

The lowest level Design Parameters (leaves) are often dimensions or properties of matter (material properties). Tolerances and factors of safety can be included in the design as independent parameters or requirements. The layout and location of the features should be specified by design parameters (for both single Design Features and repeated ones). The designer should try to minimize the number of Design Parameters that have to be dealt with at each level.

6. THE UTED DESIGN PROCESS

The process begins with a detailed need analysis, like any good systems engineering design methodology. Afterwards, the activities in the UTED Methodology are the same, but they are carried out in a different order. The Parallel Decomposition / Design Process allows the same process to be followed at each level in the design.

6.1 NEED ANALYSIS

The Need Analysis is the process by which the designer establishes the Customer Requirements. The process is iterative, and may vary from situation to situation, but the end product should be the same. In written form, the Need Analysis consists of four parts:

- Need Statement – A concise statement of the overall problem
- Definition of Terms – Important terms are defined to eliminate any confusion
- Background – Background information is collected, which is helpful for the designer to understand the context in which the design will be used.
- Definition of Customer Requirements – A formal description of the Customer Requirements as discussed in section 5.1

6.2 PARALLEL DECOMPOSITION / DESIGN

After the overall problem is well defined, then the Problem Decomposition / Design phase can begin. The other activities in typical systems methodologies (development of FRs and PRs, concept generation, downselection, final design) are still carried out, but they done at each level in the hierarchy. Functional decomposition is done in addition to a parametric decomposition of the design to aid in optimization. The system is also broken down into a set of Design Features which carryout all of the Functional Requirements. The line between the Preliminary design and final design is blurred. After the needs of the customer have been established, the modeling of the design can begin by simultaneously decomposing the problem and the solution. In the UTED methodology, the designer can go through all of the steps in the process at each level, because the decomposition of the problem and the creation of the solution are can be accomplished in parallel. Downselection between competing concepts can be done at any time during the process, as it is determined which ones will best meet the customer's needs. Just like solving any other engineering problem, the engineer must make decisions about what simplifying assumptions should be made when using this design methodology. The smaller the acceptable performance envelope of the design, the fewer assumptions you can make.

6.2.1 Advantages of Parallel Decomposition and Design

The basic framework is flexible so that top down or bottom up methods may be used, but parallel decomposition is usually the most efficient process due to the advantages listed below. The first advantage to this is that it makes parallel or simultaneous engineering easier to coordinate, by improving communication of requirements between design teams. Knowing the Performance Requirements helps the selection of the Design Features. Downselection and evaluation of concepts can be done at each level, to reduce the amount of time spent developing unfeasible concepts. It also reduces the amount of iteration required, and makes it easier to create a modular design.

The advantage to specifying values for the Performance Requirements and Design Parameters as you go is that it is much easier to do the appropriate downselection at each level. Including the Quantitative information is also useful, because you may realize that you need to add an additional FR in order to satisfy all of the requirements. A gearbox is a good example of this. The engine in a vehicle works best over a certain range of rotating speeds. These engine speeds may not match with other DPs in order to meet the vehicles speed requirements. While it may be possible to adjust other parameters such as the wheel diameter, this may not be the case. An obvious solution would be some type of mechanical transformer such as a gearbox. The designer won't know whether this Functional Requirement needs to be fulfilled unless the PRs and DPs can be met without it.

The combined design tree in Figure 6 can be used to display the information in the Function Structure, the Design Feature tree, and the Performance Requirement / Design Parameter tree in one chart. The elements in Figures 3, 4, and 5 can all be included in one chart, although some of the relationships between the elements are omitted, such as the relationships between Design Parameters shown in Figure 5. A major limitation is the size of the chart for complex systems, especially when you begin writing the names of the requirements in addition to the numbers. Because the information is different for each conceptual design, each one will have different design trees and design matrices. The Design Tables and Design Matrices serve as a much more compact way of displaying the information, and will be discussed later in the report.

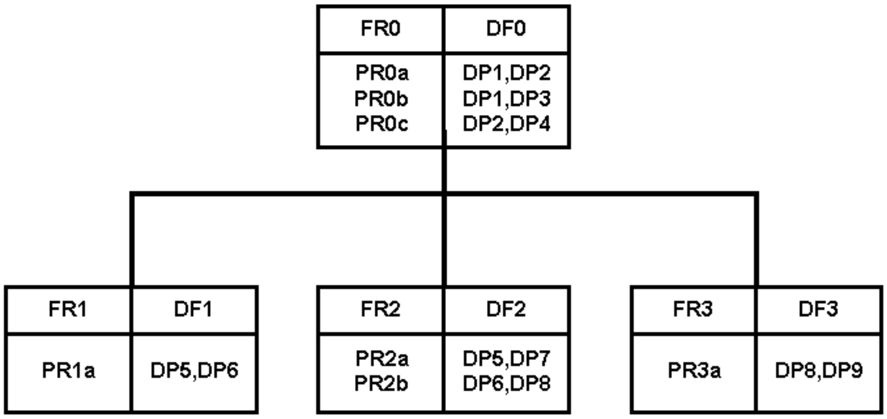


Figure 6: Combined Hierarchy

6.2.2 Decomposition / Design Process

In the following section yellow indicates the Design Characteristic(s) that is being determined, and green indicates Design Characteristics that have already been determined.

Step 1:

The top level Functional Requirement is determined based on the Customer Requirements. See Figure 7.

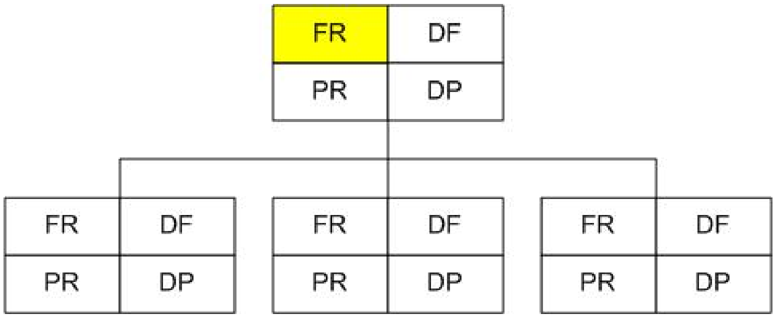


Figure 7: Parallel Decomposition Step 1

Step 2:

Top Level Performance Requirements are determined, which describes the top level FR quantitatively, by saying how well the task must be done in quantifiable terms. It is also based on the Customer Requirements. Of course, there may be more than one Performance Requirement associated with the top level Functional Requirement. The Performance Requirements will help the designer determine which Design Features will work best. See Figure 8.

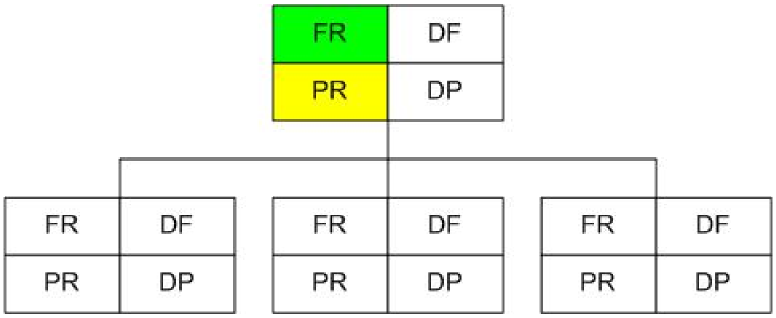


Figure 8: Parallel Decomposition Step 2

Step 3:

Top Level Design Feature is selected based on the FR to be performed and the PRs associated with it. The design feature at the top level will most likely be a system rather than an individual component. The DP variables at the top level will depend on the top level DF that is chosen. See Figure 9.

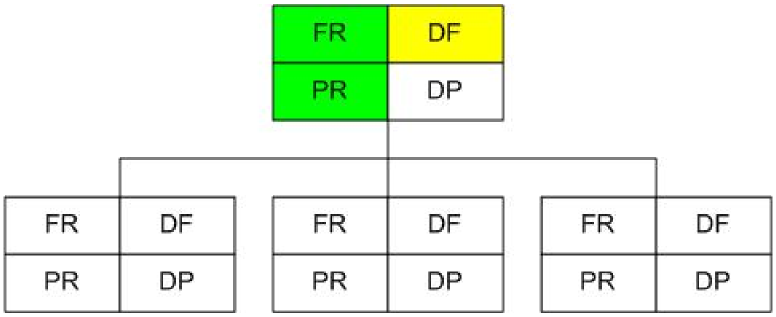


Figure 9: Parallel Decomposition Step 3

Step 4:

Lower Level Functional Requirements are determined, which are the Subfunctions that are required to make the top-level system work. These will be the functions that the subsystems will have to perform. See Figure 10.

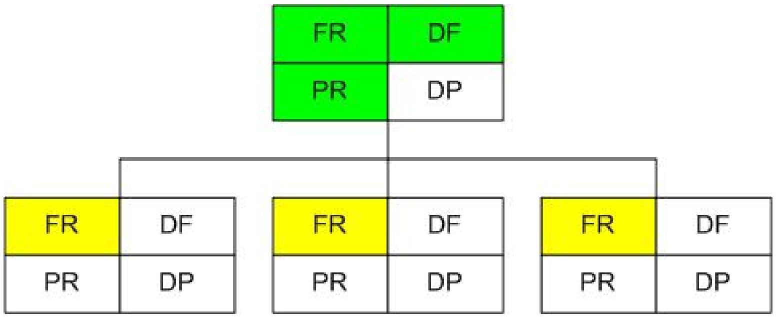


Figure 10: Parallel Decomposition Step 4

Step 5:

Top Level Design Parameters and Lower Level Performance Requirements are determined. Because most of the top level DPs will become PRs at the lower level, it is best to determine them at the same time. Doing analysis to make sure that the PRs at the top level can be met will let you know if any additional features (i.e. a transformer) are required at the lower level. As the DPs and PRs are selected in this step, it may become apparent that additional FRs must be fulfilled for the overall system to work. See Figure11.

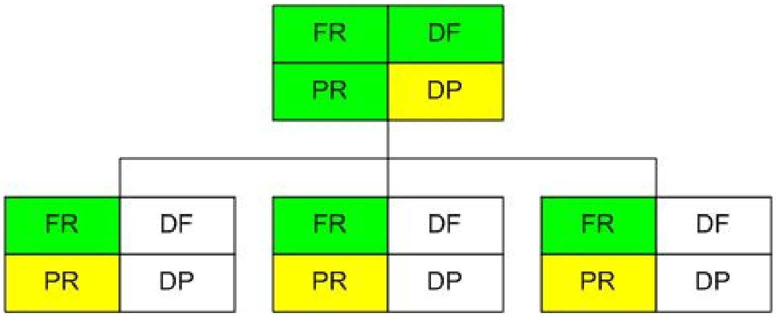


Figure 11: Parallel Decomposition Step 5

Step 6:

Lower level Design Features are determined. The DFs at the lower level are chosen to satisfy the lower level FRs. The DFs are selected on their ability to meet the PRs at that level. If there isn't any coupling then these can be selected independently. See Figure 12.

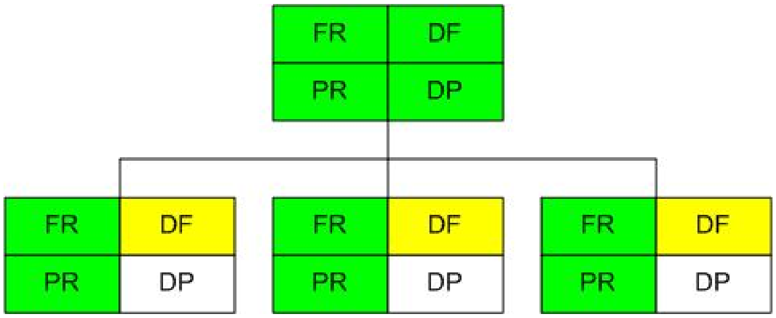


Figure 12: Parallel Decomposition Step 6

Step 7:

Lower level Design Parameters are determined. If this is the bottom level, then the DPs can be selected. If there are going to be levels beneath the second one, then the FRs at the third level will have to be determined first. See Figure 13.

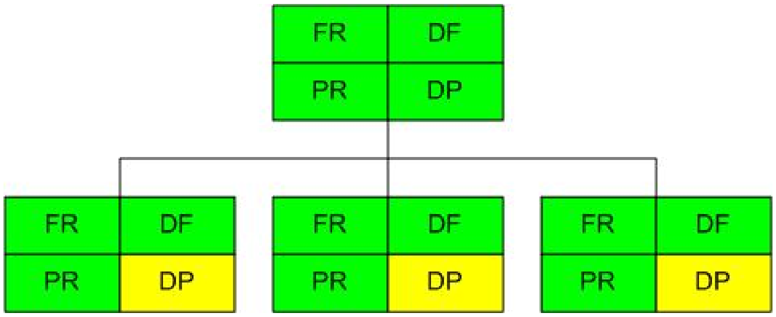


Figure 13: Parallel Decomposition Step 7

6.3 DOWNSELECTION

Using the UTED methodology allows decisions to be made between competing concepts based on both mathematical estimates for the concepts performance and axiomatic design principles. Subjective downselect procedures only evaluate concepts relative to each other, but don't provide any absolute evaluation of the concepts feasibility. If none of the existing conceptual designs can meet the requirements, then the new methodology will hopefully make this apparent during the conceptual design phase. This way, additional concepts can be generated before you move to the next level in the hierarchy in the design. The branching method of concept generation shown in the next section will allow the designer to look at the possible combination in an organized fashion.

7. EXAMPLES

7.1 SINGLE FUNCTIONAL REQUIREMENT DESIGN

A design is needed for a system to transmit electrical power from one point to another. This design is going to be a part of a system, but for the moment, let's focus on a single Functional Requirement. In this example, a metal wire was chosen as the Design Feature to satisfy the Functional Requirement. The wire will be strung between poles placed at a predetermined interval. Power loss is obviously to be minimized, so the resistance of the power line will be important. The cost of the poles is going to be based on the amount of mass that they need to carry, so the overall mass of the wires is important. The wires must support themselves, so the strength to density ratio will determine how far apart the poles can be spaced. Of course, the material costs of the power line itself will contribute to the overall cost. Figure 14 contains this information in a table format, but it doesn't show the coupling of the Design Parameters. Figure 15 shows this information, and the DPs that will affect the performance, in a convenient format called a design table. This is an example set of requirements. Depending on the situation, different sets of requirements may be chosen. The Performance Requirements were chosen to be independent of length, so the design could have the same performance regardless of the distance that the electricity is transmitted.

#	FR	DF	PR		DPs
1	Transmit electrical power	Metal wire	1a	Resistance / L	D_wire, Resistivity
			1b	Mass / L	D_wire, Density
			1c	Specific Strength	Density, Strength_Yield
			1d	Cost / L	D_wire, Density, $ / kg

Figure 14: Single FR Design Table

Figure 15 shows the same information from Figure 14 in a Design Matrix. This format has the added advantage of showing the coupling or interdependence between the Design Parameters. The letter 'X' denotes that the Performance Requirement to the left is mathematically a function of the Design Parameter above. A '0' means that the

Performance Requirement isn't dependent on that particular Design Parameter. This format can serve as a basis for doing axiomatic design analysis, as will be shown in future work.

#	FR	DF	PR		DPs				
					D_wire	Resistivity	Density	Strength_Yield	$ / kg
1	Carry electrical power	Power line	1a	Resistance / L	X	X	0	0	0
			1b	Mass / L	X	0	0	0	0
			1c	Strength / Mass	0	0	X	X	0
			1d	Cost / L	X	0	X	0	X

Figure 15: Single FR Design Matrix

Figure 16 allows the designer to list the values of the Performance Requirements and the Design Parameters, as they are determined. When the values are selected, they are placed to the right of the Performance Requirements and beneath the Design Parameters.

#	FR	DF	PR			DPs				
			#	Variable	Value	D_wire	Resistivity	Density	Strength_Yield	$ / kg
					Value->					
1	Carry electrical power	Power line	1a	Resistance / L		X	X	0	0	0
			1b	Mass / L		X	0	0	0	0
			1c	Strength / Mass		0	0	X	X	0
			1d	Cost / L		X	0	X	0	X

Figure 16: Single FR Design Matrix with Values

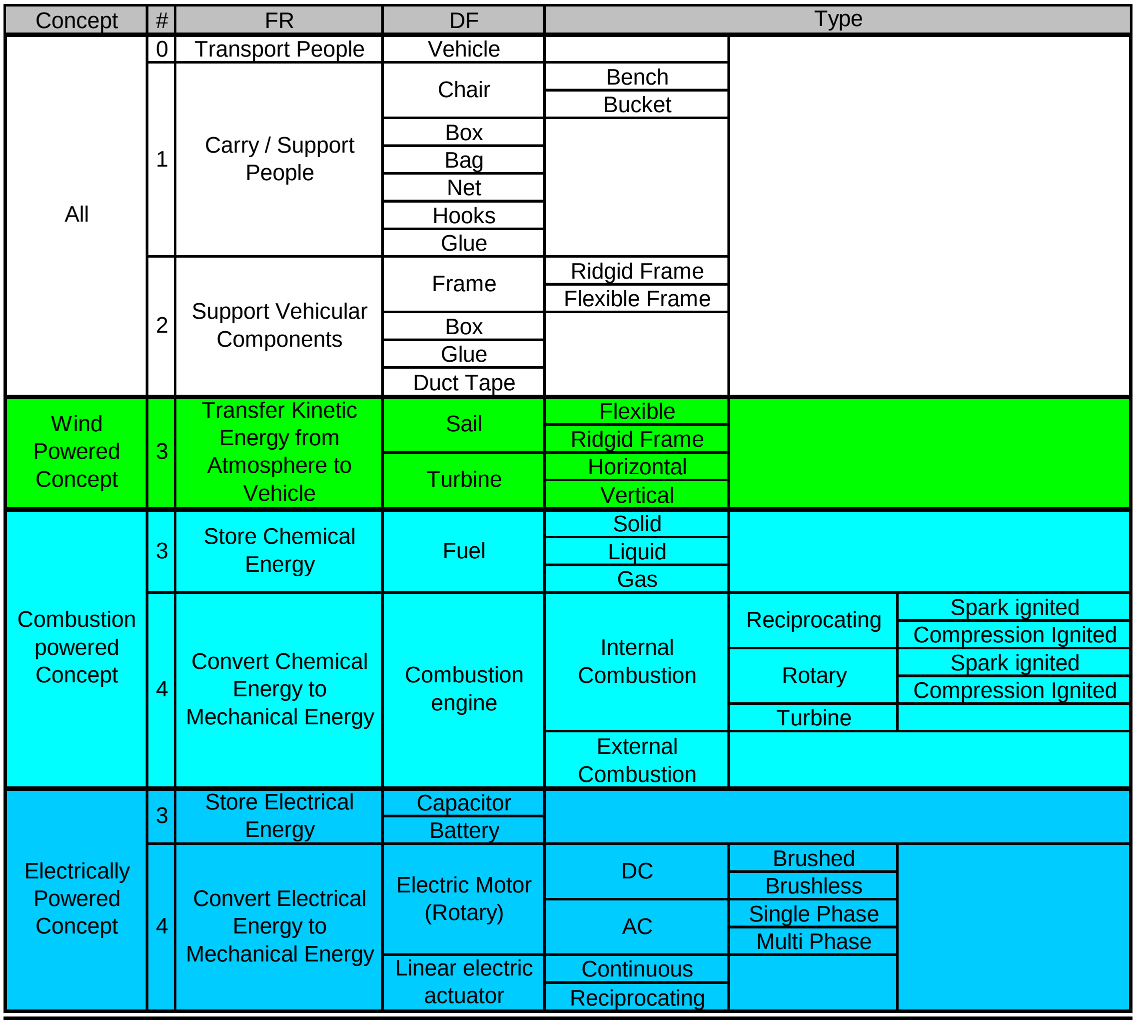
7.2 CONCEPTUAL DESIGN OF A VEHICLE

Your customer asks you to design a method of transporting people from one location to another. There are all sorts of possible Performance requirements that can be assigned to this design problem. If the Performance Requirements aren't stringent, then rather then using a vehicle to transport the people, they could just be dragged or even flung with a catapult (much to their dismay). This is where coming up with a good set of customer needs is important to direct and constrain the solution to something that the customer will be happy with (especially if the customers are the ones being transported in this case!) Before we can decide which concepts should be developed, we must first consider what performance requirements we should look at. Possible Customer Requirements include:

- Distance
- Travel Time
- Number of passengers
- Operating conditions – weather, terrain, etc.
- Passenger comfort and safety

The need statement should give a good overall summary of the customer needs. Specific needs should be presented in a list, with descriptions and explanations of each Customer Need. The designs will be differentiated first by the Functional Requirements that are chosen, and then by the Design Features that are selected to fulfill the Functional Requirements. We will assume that the people will be transported a long distance, in relative comfort and safety. This leads us to select a vehicle of some sort.

Three possible concepts (Design Features) which can satisfy the Functional Requirements are: a Wind Powered Vehicle, a Combustion Engine Powered Vehicle, or an Electrically Powered Vehicle. Of course, these aren't the only possibilities. The concept outline in Figure 17 shows the concept of branching in this problem.



Concept	#	FR	DF	Type		
All	0	Transport People	Vehicle			
	1	Carry / Support People	Chair	Bench		
				Bucket		
			Box			
			Bag			
			Net			
			Hooks			
			Glue			
	2	Support Vehicular Components	Frame	Ridgid Frame		
				Flexible Frame		
			Box			
			Glue			
			Duct Tape			
Wind Powered Concept	3	Transfer Kinetic Energy from Atmosphere to Vehicle	Sail	Flexible		
				Ridgid Frame		
			Turbine	Horizontal		
				Vertical		
Combustion powered Concept	3	Store Chemical Energy	Fuel	Solid		
				Liquid		
				Gas		
	4	Convert Chemical Energy to Mechanical Energy	Combustion engine	Internal Combustion	Reciprocating	Spark ignited
						Compression Ignited
					Rotary	Spark ignited
						Compression Ignited
					Turbine	
				External Combustion		
Electrically Powered Concept	3	Store Electrical Energy	Capacitor			
			Battery			
	4	Convert Electrical Energy to Mechanical Energy	Electric Motor (Rotary)	DC	Brushed	
					Brushless	
				AC	Single Phase	
					Multi Phase	
			Linear electric actuator	Continuous		
				Reciprocating		

Figure 17: Concept Outline

While two of the FRs are common to all of the designs, some FRs are specific to a particular conceptual design. By selecting different sets of Functional Requirements, we can affect the resulting solution greatly. All of the concepts represented in Figure 17 relate to Functional Requirements and Design Features at the second level of the hierarchy. The designer can continue to branch these concepts and develop all of them further. There may be an almost infinite number of combinations. Therefore, it may be in the designer’s best interest to eliminate certain concepts when it becomes apparent that they are infeasible, or are otherwise inferior to the rest. Depending on the time available, the scheduling of concept elimination may be varied. If there is a rush to complete the design, a single branch may be chosen at each level, leading to the development of a single concept.

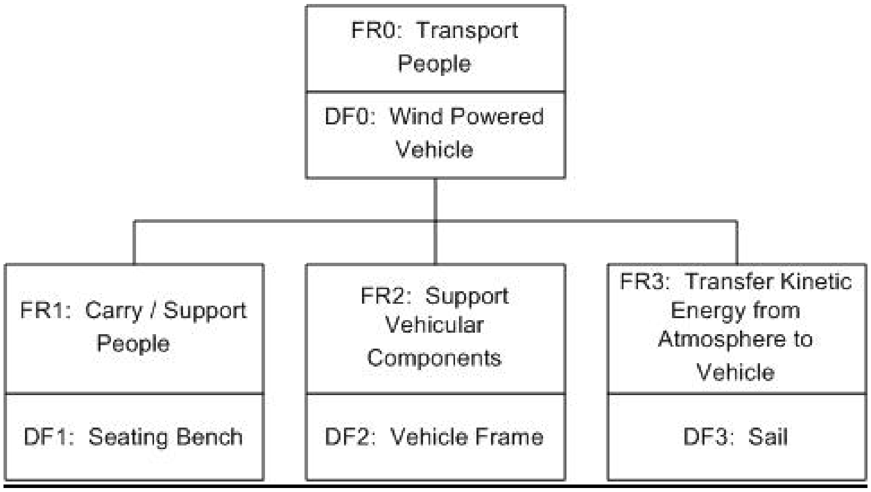


Figure 18: Wind Powered Vehicle Concept

There are two functions that should be common to all of these designs (FR1, FR2), once we have narrowed down the concepts to a vehicle. The third one is specific to this concept. In general, we knew that we needed some source of energy to move the vehicle. On this concept, the Functional Requirement was made more refined – *FR3: Transfer Kinetic Energy from the Atmosphere to Vehicle* as in Figure 18. There are of course, a couple of ways of accomplishing this function. The concepts must have compatible sets of FRs and DFs. A gasoline tank and an electric motor together, without anything else in between them, won't get the passengers or the vehicle anywhere. Figure 19 shows the FRs and DPs for a vehicle powered by an IC engine.

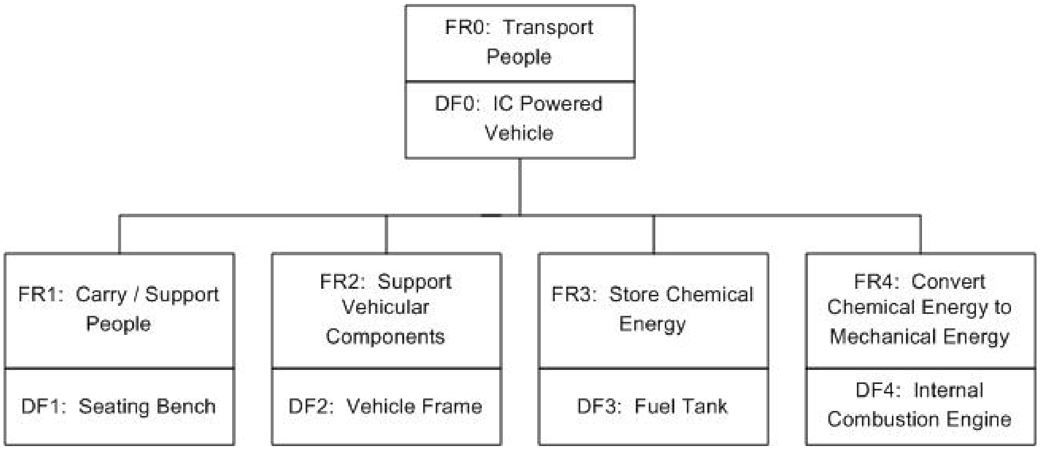


Figure 19: IC Powered Vehicle Concept

At this point there is a lot of room for innovation. Any fuel that can be burned is a possibility at this point. We need information about the expected performance to make a decision. Any type of internal combustion engine can be chosen at this point, and again, Performance Requirements will have to be established in order to select Design Features that will meet the Customer Needs. Figure 20 shows the FRs and DPs for an electrically powered vehicle.

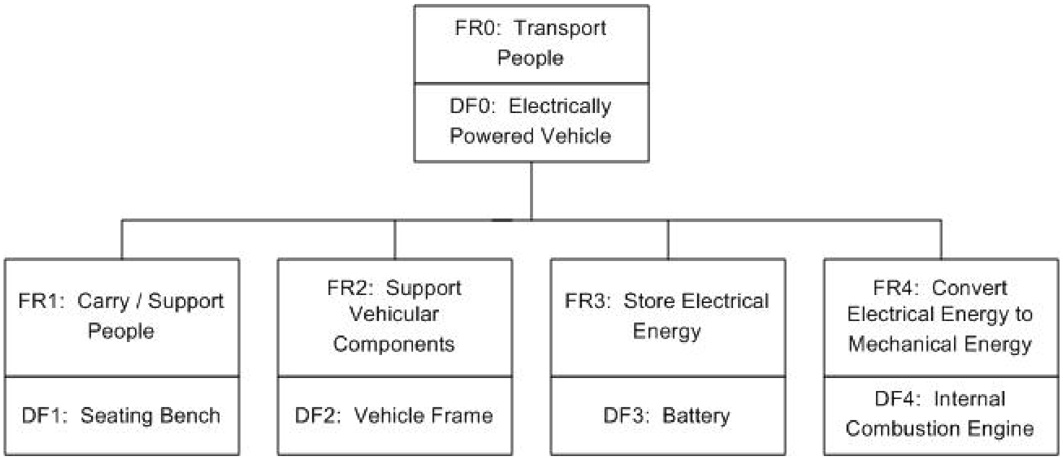


Figure 20: Electrically Powered Vehicle Concept

8. SUMMARY

8.1 DISCUSSION

In many ways, the design theory and methodology presented in this report are similar to game theory. They both take an activity that was traditionally considered more of an art than a science, and try to make it more scientific. The UTED methodology has some similarities to both the Systems Engineering Methodology and Axiomatic Design. The major similarity to axiomatic design is the division of the process into domains, although a different set of domains is used. The UTED methodology resolves many shortcomings of current methodologies, and serves as a useful tool throughout the design process. While it serves as a platform for doing axiomatic design analysis, this new methodology is substantially different from Suh's basic framework.

The UTED methodology requires new strategies to define the important characteristics of the design. The new methodology does provide the designer with a great deal more information, and shows more opportunities for improving the design. The methodology's formal structure forces the designer to select a good set of requirements, and makes poor choices more apparent. When specific requirements are difficult to satisfy, the methodology shows how changing other parts of the system can be modified to solve the problem. This is especially true in situations when requirements are dependent on a shared set of parameters.

The UTED methodology allows the practice of engineering to be greatly improved by creating a set of rules and principles, which guide the designer through the process. The UTED Methodology meets the definition of a "systems" methodology.

8.2 A REVIEW OF THE REQUIREMENTS FOR THE METHODOLOGY

Below is a list of a review of the requirements for the UTED methodology laid out in section 3, along with a brief description of how each one was satisfied. It was shown that if these requirements are met, then we have an increased likelihood of achieving a successful design.

To make design more scientific, a theoretical model of design was developed, complete with a set of rules to guide its practice. The rules allow us to set up the conceptual model of the design properly, and then proceed through the design process in an organized fashion.

One of the main goals for the methodology was to improve our ability to predict the behavior of the system. The formal and hierarchical method of defining the problem and solution provide an organized way to ensure that a full set of requirements is developed, and that these requirements are met. This will also reduce iteration, because potential problems can be eliminated before prototypes are built.

We would also like to make design processes faster and more efficient. The parallel decomposition process facilitates simultaneous design, which can speed up the design process. Also, the parallel decomposition of conceptual and quantitative aspects of the design allows evaluation of competing concepts to occur more quickly, reducing the time spent developing infeasible design alternatives and contributing to design process efficiency.

The new methodology should foster innovation. It has been shown that the use of abstraction in stating Functional Requirements promotes innovation. The branching technique demonstrated in this paper provides an organized and directed method of concept generation, which encourages innovation at every level in the hierarchy.

The new methodology must have a scientific basis. This methodology includes information that isn't a part of the other methodologies. The UTED methodology is the only know methodology which recognizes and defines all four of the overlapping design domains (problem, solution, conceptual, quantitative). Recognizing these domains allowed additional rules to be developed to define the process, and therefore make it more scientific, based on the arguments in the introduction of this thesis.

8.3 NOTEWORTHY CHARACTERISTICS OF THE UTED METHODOLOGY

The UTED methodology provides endless possibilities for improving design processes. Below are a few of the many characteristics or attributes, which set it apart from methodologies that are currently in use.

1. The inclusion of Design Features in the basic framework allows conceptual aspects of the solution to be related to the problem, and the quantitative aspects of the solution.

2. A single hierarchy that extends through all of the Design Domains, and ensures the compatibility of requirements within each level and between levels, between elements of the solution, and between the problem and the solution. Individual Design Features in design to be traced to the requirements they fulfill, and all requirements can be traced back to the original need

3. The Parallel Decomposition process allows the entire design to be accomplished in a top down fashion. The first advantage to this is that it makes parallel or simultaneous engineering easier to coordinate, by improving communication of requirements between design teams. Knowing the Performance Requirements helps the selection of the Design Features. Downselection and evaluation of concepts can be done at each level, to reduce the amount of time spent developing unfeasible concepts. It reduces the amount of iteration required, and makes it easier to create a modular design.

4. A novel method of organizing and displaying all of the important information about the design, called Design Matrices.

8.4 FUTURE DEVELOPMENTS

Below is a list of related topics which will be studied in future research:

- Design of manufacturing / assembly processes
- Design of experiments
- A variety of design procedures, which can be tailored to fit specific designs, including modified versions of pre-existing processes, to guide the designer through each phase of the process.
- A detailed procedure for conducting Failure Modes and Effects Analysis – By the time the design matrices have been completed, you are about halfway done with the FMEA. FMEA information can be incorporated into or linked to the design matrices.
- Automation of design processes by computers
- Theory and quantification of complexity
- Use of axiomatic design and updated versions of Suh’s axioms.
- Process for defining interfaces
- Business principles showing why it is effective - Lean Design (from lean manufacturing theory)

9. CONCLUSION

The objective of this project was to develop a theoretical model of design, which is universal and has a scientific basis. In analyzing the needs of design engineers, it was determined that a better system of modeling designs was the missing ingredient that needed to be developed in order to improve the practice of design. A scientific theory of design was developed and shown to meet these criteria. The UTED methodology covers many important aspects of design, which are overlooked by other methodologies. This allowed a set of rules to be developed, which guide the designer through the design, and allow a more scientific process to be used. As discussed in the paper, making design more scientific increases the likelihood of achieving a successful design, and the universal model of design presented in this thesis paves the way for design process improvements on numerous fronts.

REFERENCES

1. Suh N., “Axiomatic Design, Advances and Applications," Oxford University Press, 2001, ISBN 0-19-513466
2. Casti J., “Complexification,” Harper Collins Publishers, Inc., 1994, ISBN 0060925876
3. Blanchard B. and Fabrycky S., “Systems Engineering and Analysis,” Prentice Hall, 1998, ISBN 0131350471

VITA

Scott Patrick Dyas

College Address
503 Cherry, Apt. 106
College Station TX, 77840
Phone: (210) 833-0308

Permanent Address
5863 Woodridge Oaks
San Antonio TX, 78249
Email: sdyas@tamu.edu

Education

Texas A&M University, College Station Texas 1996-2001
B.S. in Mechanical Engineering. (Minor in Political Science)

Texas A&M University - College Station Texas 2001-2005
M.S. in Mechanical Engineering

Work Experience

Department of Mechanical Engineering- Texas A&M University
Instructor, Senior Design Courses 2002-2005
Graduate Research Assistant 2001-2002

Schlumberger – Sugar Land, TX 2004, 2005
Duties: Designed high cycle hydraulic actuation system for down-hole tools. Developed manufacturing processes and designed equipment for insulating wire splices.

Southwest Research Institute – San Antonio, TX 2002
Duties: Led design of new automotive gaseous emissions sampling and high flow particulate collection systems.

Areas of Professional Interest

Design Methodology / Systems Engineering
Automotive Engineering
Engineering Education