Proceedings of DETC99
1999 ASME Design Engineering Technical Conferences
September 12-15, 1999 – Las Vegas, Nevada

DETC99/DTM-8764

PRODUCT ARCHITECTURE DEVELOPMENT WITH QUANTITATIVE FUNCTIONAL MODELS

Robert B. Stone†
Basic Engineering Department
University of Missouri-Rolla
Rolla, MO 65409-0210
rstone@umr.edu

Kristin L. Wood
Department of Mechanical Engineering
The University of Texas at Austin
Austin, TX 78712
wood@mail.utexas.edu

Richard H. Crawford
Department of Mechanical Engineering
The University of Texas at Austin
Austin, TX 78712
rhc@mail.utexas.edu

ABSTRACT

A key phase in product design and development processes is the establishment of product architectures. During this phase, functional models are transformed into alternative product layouts. In this paper, we introduce a methodology for representing a functional model of a product in a quantitative manner. The quantitative functional model captures product functionality and customer need information. Repositories can be created which house product design knowledge for a vast number of products using this novel representation. Numerical manipulations of such a repository assist in developing product architectures. In particular, *product families* and aggregate customer need ratings for modules are easily computed. Also, the quantitative functional model provides a mechanism to archive and transmit design knowledge across time and space. Prior to presentation of the methodology, a review of customer needs gathering techniques and module identification methods is given. Results from a repository of 70 consumer products are presented to illustrate the utility of the quantitative functional model.

Keywords: Functional modeling, product architecture, modular design, customer needs.

1. INTRODUCTION

Functional modeling is an important step in product design, particularly when defining product architecture. Several systematic methods guide the development of functional models and produce graphical representations such as function structures (Pahl and Beitz, 1988; Ulrich and Eppinger, 1995; Ullman, 1997, Otto and Wood, 1997). However, functional modeling is rarely considered a quantitative activity. In this paper we introduce the concept of a *quantitative* functional model – a vector representation of a product in which the number of distinct sub-functions determine the dimension of the vector and the customer need rankings of each sub-function determine the magnitude. By aggregating these models into a repository of products (a matrix of products and functions), the quantitative functional model represents a powerful new manner of archiving and transmitting product design knowledge. This research complements other recent work which pursues varying methods of storing and extracting design knowledge from existing products (Murdock *et al*., 1997; Altshuller, 1984).

In the current work, the product repository is used to group products together based on functionality and customer needs. This provides a tool for product architecture development. For example, an early functional model of a product (i.e. a function structure) can be transformed to its quantitative equivalent and used to find other functionally similar products. Additionally, by identifying products that are similar in function and customer needs, it sets the stage for modular product architecture development. Modular product architectures, in particular, offer distinct manufacturing advantages as well as opportunities for automation in the design process. Automated design techniques for combining components (modules, in essence) quickly generate alternatives, and, in some cases, optimize some product features (Bradley and Agogino, 1994; Ward, 1989; Ward and Seering, 1993; Schmidt and Cagan, 1997a & b; Vadde et al., 1992).

For modular products, the repository of quantitative functional models provides another benefit – the ability to associate a customer need rating with a module. Using the initial customer need ratings of the repository, numerical manipulations compute an aggregate customer need rating for a group of sub-functions defining a module. This provides a unique justification of a modular product architecture by customer need.

† Corresponding author: 102A Basic Engineering Building, University of Missouri-Rolla, Rolla MO 65409-0210.

Since the quantitative functional model incorporates customer need data, a review of systematic techniques to gather customer needs follows the nomenclature section. A modular product architecture method for identifying modules from a functional model is next. In Section 4, the methodology for developing and using a product repository of quantitative functional models is presented.

2. GLOSSARY OF TERMS AND NOMENCLATURE

Several terms and variables are introduced in the development of the quantitative functional model methodology. These terms are defined below.

Module: a physical product sub-structure having a one-to-one correspondence with a subset of a product's functional elements.

Product family: a group of products that are similar in function and customer needs.

Product vector (ϕ): a vector indicating a product's functionality and customer need ratings for each sub-function.

Product-function (Φ) matrix: a collection of product vectors which forms a *m*x*n* matrix, where *m* is the total number of distinct sub-functions present in the product set and *n* is the number of products.

Normalized product-function (**N**) matrix: the product-function matrix after being normalized for differing customer need ratings and product complexity.

Product-product (Λ) matrix: a symmetric matrix (computed from **N**) that compares the similarity of all products in the repository based on function and customer needs.

Aggregate customer need ranking: a derived customer need ranking for a group of sub-functions which forms a module.

3. REVIEW

Gathering Customer Needs

The *customer* is a statistical concept; there are numerous potential purchasers of a new product being designed. The *customer population* is the set of persons whom we as designers want to be purchasers of a new product. Typically the customer population is varied. There are various strategies that might be adopted for dealing with such diversity, from offering a single product and forcing customers to be happy with that offering, to permitting customization of features on the product, to offering a portfolio of different offerings, each tailored to a market niche. To drive this decision making task, the customer population must be understood in terms of their criteria for evaluating the product offering.

When comparing methods to gather customer needs, conducting interviews is the method that provides the most information per quantity of effort. Griffin and Hauser (1993) report that focus groups are more costly to attain the same amount of information. They also report that interviewing nine customers for one hour each will obtain over 90% of the customer needs that would be uncovered when interviewing 60 customers, as observed on a single function product (industrial lunch pails). Such a rule is not entirely valid for multi-function products, implying the sample size of customers must grow as the number of product functions increases.

Thus, for the purpose of this research, customer needs are gathered through interview methods. If a product is being evolved, it is assumed that the customer uses the current generation of the product during the interview. Anywhere between 10 and 30 interviews are conducted for a product, depending on its possible uses and complexities. The statements provided by the customers are recorded in their voice, subsequently interpreted, aggregated, and verified by the customers, and then weights are assigned by a range of customers through questionnaires or follow up interviews. Recorded responses include likes, dislikes, and latent needs, as observed by the interviewers. Likewise, the weights are recorded as averages and standard deviations (if needed). Based on this process, a list of categorized customer needs is obtained, with associated weights on a scale of 1-5 or 1-10 (the higher the number, the more important the need).

Modular Product Architecture - Identifying Modules

Modular product architectures are treated in a variety of ways in recent literature. Ulrich and Tung (1991) qualitatively define modular products and identify six basic types of modular products. Matrix based methods provide the first mathematical attempt at clumping sub-functions of functional models into modules based on certain criteria such as interaction or life cycle engineering requirements (Pimmler and Eppinger, 1994; Newcomb *et al.*, 1996, Gu *et al.*, 1997; Rosen, 1996; Marks *et al.*, 1993). Others offer methods for integrating modules into a product once they are identified (Ulrich and Eppinger, 1995; Cutherell, 1996). There is also work (Gupta and Krishnan, 1997; Krishnan *et al.*, 1996) investigating the economic aspects of selecting common components for product families. Most recently, modular product architecture focuses on a combination of heuristic and quantitative efforts to develop a comprehensive modular design methodology (McAdams, Stone and Wood, 1998 & 1999; Stone, Wood and Crawford, 1998 & 1999; Allen and Carlson-Skalag, 1998). These methods identify modules from a functional model of a product, create rough geometric layouts and group products into families based on function.

Stone, Wood and Crawford (1998) develop a set of three heuristics for identification of modules from a functional description. This work provides a systematic method for clustering elements which is lacking from previous approaches to modular product architecture (Cutherell, 1996; Ulrich and Eppinger, 1995). The heuristics require a functional model in the form of a function structure, where sub-functions are then clustered based on flow (energy, material, or signal) relationships. The three heuristic propositions are stated below and shown schematically in Figs. 1-3.

> **Dominant-Flow Proposition:** The set of sub-functions which a flow passes through, from entry or initiation of the flow in the system to exit from the system or conversion of the flow within the system, define a module.

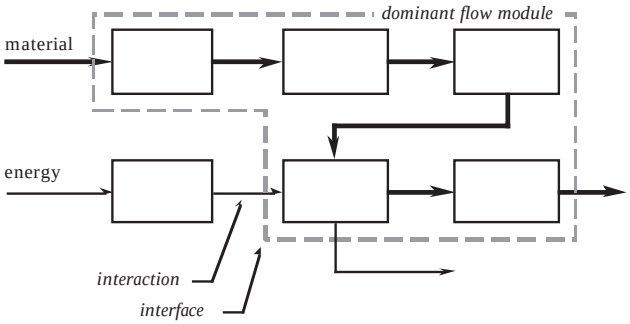

Figure 1. Dominant flow heuristic applied to a generic function structure.

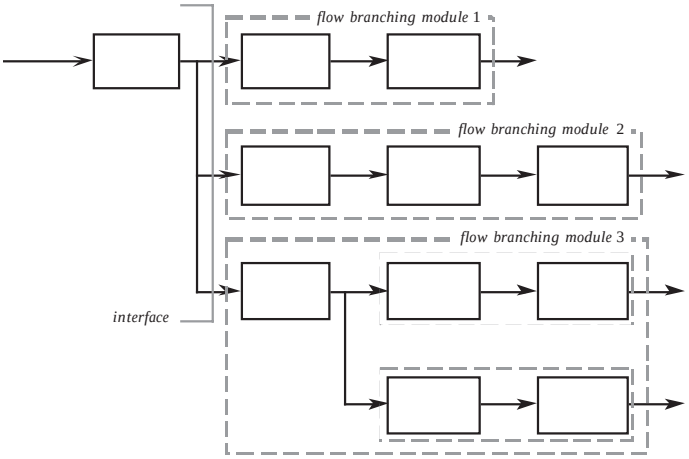

Figure 2. Flow branching heuristic applied to a generic function structure.

Branching Flow Proposition: Parallel function chains associated with a flow that branches constitute modules. Each of the modules interfaces with the remainder of the product through the flow at the branch location.

Convert-Transmit Proposition: A conversion sub-function or a conversion-transmission pair or proper chain of sub-functions constitutes a module.

Application of the heuristics represent a simple, but elegant way of identifying modules at a functional level. As an example of the heuristic methods, consider the function structure of a power screwdriver shown in Fig. 4. The dominant flow module identifies three modules: *coupling*, *supply electricity* and *torque transmission*. The branching flow module identifies modules base on flows that branch. Here, the flows *bit, hand* and *human force* branch to identify the modules *coupling/decoupling*, *decoupling*, *manual use*, *rotational lock* and *positioning*. Note that different heuristics may identify overlapping modules or modules that are subsets of others. Finally, the convert-transmit heuristic identifies the *electricity to torque* module. The identified modules suggest groups of sub-functions that can be combined into a single component or independent set of grouped components. The next

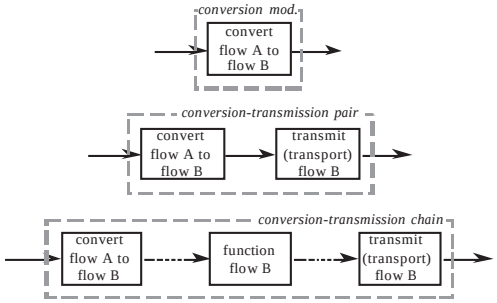

Figure 3. Conversion-transmission applied to a generic set of sub-functions.

question, though, asks if there are certain modules that are more valuable to implement than others. We answer that question with the quantitative assessment method in the next section.

4. METHODOLOGY: GENERATING A PRODUCT REPOSITORY, PRODUCT FAMILIES AND AGGREGATE CUSTOMER NEED RANKINGS

The quantitative functional model methodology for developing product architectures is part of the overall product design process for new and existing products outlined in Fig. 5. Specifically, the quantitative functional model methodology presented in this section has three phases, also shown in Fig. 5. Inputs and outputs of each phase are indicated by arrows. The first phase produces a repository of both functional and customer needs information about product designs in a compact, numerical form. Its steps include the key correlation of customer need ratings to sub-functions which forms product vectors and the aggregation of product vectors into a product-function matrix. The second phase manipulates the repository to identify groups of similar products, which can share similar product architectures. In the third phase, the product repository is used to compute aggregate customer need ratings for modules. This allows original customer needs ratings to be used to support product architecture choices even though those issues were not directly addressed in the interview process.

The Product Repository

The first phase of the methodology creates the product-function matrix (which is the basic information forming the product repository) and assumes that a number of products are being studied. For each product, a function structure and a list of rated customer needs are required. The function structure should be expressed in a common form, i.e. a common vocabulary of functions and flows spanning different levels of detail from which sub-function descriptions are chosen. Note that all sub-function descriptions must be expressed at a consistent level of detail, though the level of detail chosen is at the discretion of the designer. One such common form is detailed in Little *et al*. (1997) and Stone and Wood (1999), and is followed in this article. The customer needs are compiled as discussed in Section 3.

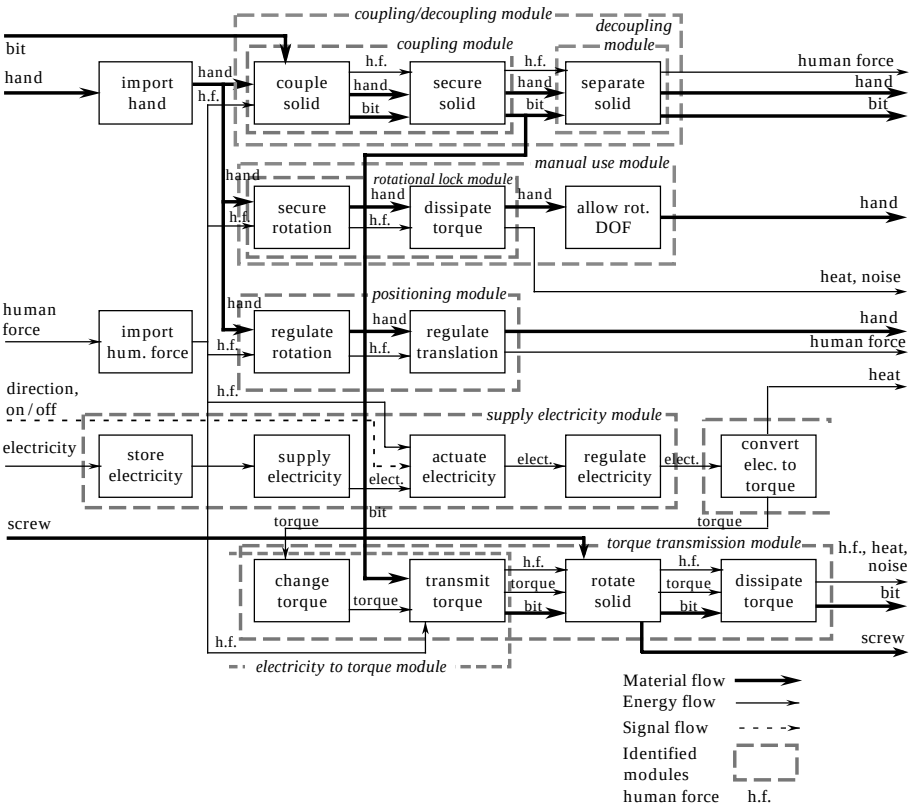

Figure 4. Function structure of a SKIL power screwdriver with modules identified by the heuristic methods.

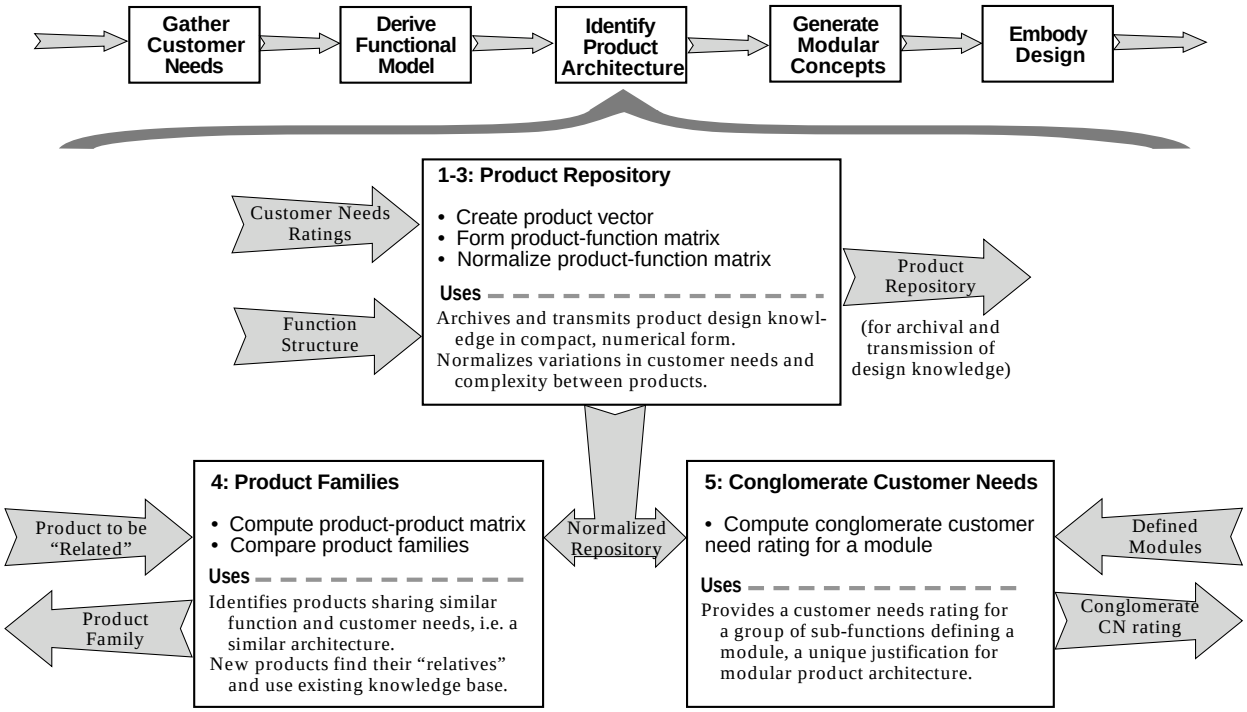

Figure 5 Overview of the methodology to use quantitative functional models for developing product architectures.

Customer need	Scaled cust. need rating (1 to 5)	Associated flow(s)	Associated sub-function(s)
Powerful	5	electricity, torque	convert elect. to torque, change torque
Fast	4	electricity, torque	convert elect. to torque
Long lasting battery	5	electricity	store electricity
Short charge time	4	electricity	store electricity
Manual use capability	4	hand, human force	import human hand, import human force, secure rotation
Reversible (screw and unscrew)	4	electricity, torque	actuate electricity
Lightweight	4	human force, electricity	import human force, convert elect. to torque
Weight balance	2	human force	regulate rotation, regulate translation
Small size	3	hand, electricity	import human hand
Comfortable handle	2	hand	import human hand
Automatic shut-off	3	electricity	actuate electricity
Interchangeable tips	4	bit	secure solid
Maintenance free	1	torque, bit, human force	dissipate torque, store solid
Variable velocity	3	electricity	

Table 1 Step 1: Generating a product vector. Task 1 involves correlating customer needs to flows and sub-functions for the SKIL power screwdriver example.

Step 1: Create the product vector

This step begins with two inputs: 1) customer need ratings scaled to the range of 1 (optional) to 5 (must have) (Otto and Wood, 1996 & 1997; Ulrich and Eppinger, 1995) and 2) a function structure expressed in a standard form (Little, 1997; Little *et al*., 1997). The scaled customer need ratings are then mapped to the sub-functions of the product function structure to produce a product vector, ϕ. This step is broken into two tasks. First, each customer need is associated with a flow or set of flows. Then, by following each flow through the function structure, sub-functions are related to flows and receive the original customer need rating if the sub-function addresses the customer need. Sub-functions may address more than one customer need, and thus receive the cumulative value of the customer need ratings.

Some sub-functions of a product may not be directly associated with any customer need, though they are essential to the mechanics of the overall function. These sub-functions are termed *supporting functions*. The sub-functions that do have a direct correlation are called *carrier functions*. So that the supporting functions are not neglected in the arithmetic manipulation that follows, they are given a value of 1. To maintain the five-point resolution of the customer needs ranking, a value of 1 is added to the carrier functions as well. Now the scale is 1 to 6, where a value of 1 denotes a supporting function and a value of 6 denotes an essential or highly important function. Note that values greater than 6 can exist when a sub-function relates to more than one customer need.

Example: Power screwdriver product vector

An example of the customer need rating to sub-function correlation is shown in Table 1. This example examines the SKIL power screwdriver first examined in Section 3. The first and second columns list the customer needs and their scaled rating. The third column identifies the flows that directly affect the stated customer need. The relationship between sub-function and flow is often evident, but inevitably some engineering judgment is required. The final column is where engineering judgment is definitely required. By following the flows through the function structure, shown previously in Fig. 4, we identify the sub-functions that directly meet the customer need.

Consider the customer need "powerful." It is associated with the flows *electricity* and *torque*. Following the flow of *electricity* through the function structure, we identify the sub-function *convert electricity to torque* as directly impacting that need. Considering the flow *torque*, it affects the screwdriver's power through the sub-function *change torque*.

Once the associated sub-functions are identified, each receives the value of the original customer need rating. This process effectively assigns a customer need rating to those sub-functions which are viewed as directly affecting the associated customer need. These customer-need ratings are then summed to produce the value of the weighted sub-function as shown in Table 2. Note that the third column of Table 2 forms the product vector for the SKIL power screwdriver.

Step 2: Form the product-function matrix (i.e. the product repository)

Once the product vectors are formed, they are aggregated to form a product repository. This repository represents the basic mechanism for archiving product design knowledge. To form

Table 2 Step 1: Generating the product vector. Completion of Task 2 gives the weighted sub-function values for the SKIL power screwdriver product vector.

Sub-function	Associated customer need ratings	Weighted cust. need rating
actuate electricity	4, 3, 1	8
allow rot. DOF	1	1
change torque	5,1	6
convert elect. to torque	5, 4, 4, 1	14
couple solid	1	1
dissipate torque	1,1	2
import human force	4, 4, 1	9
import human hand	4, 3, 2, 1	10
regulate electricity	3, 1	4
regulate rotation	2, 1	3
regulate translation	2, 1	3
rotate solid	1	1
secure rotation	4,1	5
secure solid	4,1	5
separate solid	1	1
store electricity	5,4,1	10
supply electricity	1	1
transmit torque	1	1

the product-function matrix, each product vector is transformed to a *m*x1 vector, where *m* is the total number of different sub-functions for *n* products. Since *m* is typically greater than the original dimension of the product vector, zeros are added to indicate sub-functions that do not exist in a given product. The product vectors are then arranged into a *mxn* product-function matrix, Φ. Each element ϕ_{ij} is the cumulative customer need rating for the *i*th function of the *j*th product.

Example: Aggregating product vectors to form the product-function matrix

Consider the power screwdriver product vector from the previous example. In order to create a new product repository, we must aggregate it with other product vectors. An additional product vector for a DeWalt power sander is developed. Both the vectors are shown in Fig. 6 (a). The power screwdriver has 18 different sub-functions while the power sander has 21. Between the two products there are 32 distinct sub-functions. Therefore, the initial repository of two products is represented by a 32x2 matrix, shown in Fig. 6 (b). Note that zeros are added to the original product vectors to indicate sub-functions that are not present in the product.

Step 3: Normalize the product-function matrix

Since Φ (from Step 2) is designed to accumulate product design knowledge over time, it may have many different contributors. The systematic methods for gathering customer needs and developing functional models make the product vector generation step (Step 1) less designer-dependent, but they still generate product vectors of varying detail. To compensate for these variations, Φ is normalized across the entire product space. This normalization also accounts for cases where customer needs may affect many sub-functions and thus inflate the sub-functions importance ratings. The philosophy used to normalize the product-function matrix relies on two complementary points:

1. all products are of equal importance, and
2. products with more sub-functions are more complex; therefore the customer need ratings must be normalized to account for varying complexity.

The first point is realized by scaling the customer need rating of each sub-function such that the sum of a given product's sub-function customer need ratings equals the average sum of the customer need rating for all products. This normalization expresses each sub-function customer need rating as a fraction of the average sum of customer need ratings for all products. The complexity issue of the second point is addressed by scaling each product sub-function by the ratio of the number of product sub-functions to the average number of sub-functions per product. Thus, more complex products (ones with more sub-functions) receive a higher normalized customer need rating.

Once implemented, the normalized version of Φ, **N**, has elements

$$v_{ij} = \phi_{ij}\left(\frac{\overline{\eta}}{\eta_j}\right)\cdot\left(\frac{\mu_j}{\overline{\mu}}\right), \tag{1}$$

where the average customer need rating is

$$\overline{\eta} = \frac{1}{n}\sum_{i=1}^{m}\sum_{j=1}^{n}\phi_{ij}, \tag{2}$$

the total customer need rating for the *jth* product is

$$\eta_j = \sum_{i=1}^{m}\phi_{ij}, \tag{3}$$

the number of functions in the *jth* product is

$$\mu_j = \sum_{i=1}^{m} H\left(\phi_{ij}\right), \tag{4}$$

and the average number of functions is

$$\overline{\mu} = \frac{1}{n}\sum_{i=1}^{m}\sum_{j=1}^{n} H\left(\phi_{ij}\right)\cdot \tag{5}$$

H is a Heaviside function, *n* is the number of products and *m* is the total number of different sub-functions for all products.

Normalizing the Φ matrix provides a level playing field on which to compare products. The averaging and scaling technique defined above is an intuitive way to account for variations in customer needs and functional models.

SKIL power screwdriver	
actuate electricity	8
allow rotational DOF	1
change rotation	6
convert elect. to rotation	14
couple solid	1
dissipate rotation	2
import human force	9
import human hand	10
regulate electricity	4
regulate rotation	3
regulate translation	3
rotate solid	1
secure rotation	5
secure solid	5
separate solid	1
store electricity	10
supply electricity	1
transmit rotation	1

&

De walt sander	
actuate electricity	3
convert elect. to rot.	9
convert rot. to pneum.	9
dissipate sound	4
dissipate heat	6
dissipate vibrations	4
export solid	1
guide gas	6
import electricity	1
import human force	7
import human hand	15
import solid	1
indicate status	1
position solid	8
regulate electricity	1
remove solid	6
secure solid	13
separate solid	6
store solids	3
transmit human force	7
transport solid	3

(a)

⇒

Sub-function\ Product	SKIL power screwdriver	De walt sander
actuate electricity	8	3
allow rotational DOF	1	0
change rotation	6	0
convert elect. to rot.	14	9
convert rot. to pneum.	0	9
couple solid	1	0
dissipate sound	0	4
dissipate heat	0	6
dissipate rotation	2	0
dissipate vibrations	0	4
export solid	0	1
guide gas	0	6
import electricity	0	1
import human force	9	7
import human hand	10	15
import solid	0	1
indicate status	0	1
position solid	0	8
regulate electricity	4	1
regulate rotation	3	0
regulate translation	3	0
remove solid	0	6
rotate solid	1	0
secure rotation	5	0
secure solid	5	13
separate solid	1	6
store electricity	10	0
store solids	0	3
supply electricity	1	0
transmit human force	0	7
transmit rotation	1	0
transport solid	0	3

(b)

Figure 6 Two product vectors (a) aggregated to create a product-function matrix (b).

Example: Normalizing the product-function matrix

Continuing with the prior example, we apply the normalization procedure outlined above to the product repository to account for any variation in customer need and complexity of the functional model. Using equation 3, the total customer need ratings for the screwdriver and sander are 85 and 114, respectively. Likewise, from equation 4, the total numbers of functions are 18 and 21, respectively. Computing the averages (from equations 2 and 5) gives $\overline{\eta} = 99.5$ and $\overline{\mu} = 19.5$ for the repository. Now, these values are substituted into equation 1 to produce **N**, which is shown in Fig. 7. As expected, the normalization process increases the relatively lower values of the screwdriver and reduces the values of the sander. While this example is a simple application, this technique becomes crucial as the repository grows and receives contributions from different sources.

Product Families

The first three steps of the product architecture methodology create a product repository from quantitative functional models. Now the product repository can be manipulated to identify groups of products sharing similar function and customer needs, known as product families. In terms of product architecture, this technique offers many opportunities. For instance, product families lend themselves to a modular architecture, allowing a family to use a set of core components. Also, a new product's functional model can be used to find its product "relatives" and use existing knowledge to guide its development. The next step of the methodology explains how to do this.

Sub-function \ Product	SKIL power screwdriver	De walt sander
actuate electricity	8.644	2.820
allow DOF of solid	1.081	0.000
change rotation	6.483	0.000
convert elect. to rot.	15.128	8.460
convert rot. to pneum.	0.000	8.460
couple solid	1.081	0.000
dissipate sound	0.000	3.760
dissipate heat	0.000	5.640
dissipate translation	2.161	0.000
dissipate vibrations	0.000	3.760
export solid	0.000	0.940
guide gas	0.000	5.640
import electricity	0.000	0.940
import human force	9.725	6.580
import human hand	10.805	14.099
import solid	0.000	0.940
indicate status	0.000	0.940
position solid	0.000	7.520
regulate electricity	4.322	0.940
regulate rotation	3.242	0.000
regulate translation	3.242	0.000
remove solid	0.000	5.640
rotate solid	1.081	0.000
secure rotation	5.403	0.000
secure solid	5.403	12.219
separate solid	1.081	5.640
store electricity	10.805	0.000
store solids	0.000	2.820
supply electricity	1.081	0.000
transmit human force	0.000	6.580
transmit rotation	1.081	0.000
transport solid	0.000	2.820

Figure 7 The normalized product-function matrix for the example repository.

Step 4: Compute the product-product matrix

Product families are generated based on customer weighted sub-function similarity across a set of products. The product similarity method uses the product-function matrix **N**, which is a collection of product vectors in sub-function space. The product vectors that result from Eq. 1 are renormalized such that their norm is 1. The renormalization is strictly for convenience, as shown later. To create the product family, one product is chosen as the generating product. This choice is the product against which all other products will be compared. Then the inner product is calculated between the normalized generating product vector and each of the remaining normalized product vectors. Forming the inner product between the generating product *a* and another product *b*, $a \circ b$, gives the projection of the product *b* on the generating product *a*. The inner product of a product with itself (the completely similar product) is 1 and the inner product of a product with one that shares no sub-functions in common is zero.

Matrix multiplication allows all the projections for all possible generating products to be obtained in a compact form. This matrix of projections is

$$\Lambda = \hat{N}^T \hat{N}, \tag{6}$$

where $\hat{N}$ is the matrix of unity normalized product vectors (each vector of **N** is renormalized to one), similar to **N**. Each element of Λ, λ_{ij}, is the projection of the *i*th product on the *j*th product. Λ is known as the product-product matrix.

The product-product matrix is customer need based. Another method to determine similarity is to examine sub-function commonality without regard to customer needs. Here we use a binary form of **N**, called **B**, where each component of the matrix receives a 1 if the product has the sub-function or a zero if it does not. Substituting **B** for $\hat{N}$ in Eq. 6 produces a matrix with elements that list the percentage of sub-functions the two products have in common.

Therefore, Λ and its binary counterpart **B** produce different product family rankings. Λ includes customer needs in its formulation. Thus, it not only identifies common sub-functions, but also products with similar levels of customer needs. The binary similarity ranking is also shown in the example that follows for comparison.

Example: Computing a product family

The SKIL power screwdriver of the previous examples is used to generate a product family. Here, a 70-product repository (Stone, 1997; McAdams, Stone and Wood, 1998) is used to generate the product family. The repository is not repeated here, as it is a 147 sub-function x 70 product matrix. Following the outlined procedure, **N** is renormalized such that the product vectors each have a norm of 1. Then Λ is formed, and the column corresponding to the SKIL power screwdriver is sorted in descending order. The top ten products most similar to the SKIL screwdriver are listed in Table 3.

There are several things to note from Table 3. First, as expected, the similarity index for the SKIL power screwdriver compared with itself is 1. Also, agreeing with our intuition, the Black & Decker cordless screwdriver is the most similar product to the SKIL power screwdriver. This result makes sense as the two products compete for the same customer market and appear physically similar. The subsequent products' similarity, though, is increasingly less intuitive. For instance, would an observer think

Table 3 Product family with SKIL power screwdriver as the generating product.

Product	Similarity index, λ_{ij}	% of sub-functions in common
SKIL power screwdriver	1.0000	100
B&D cordless screwdriver	0.7652	95
Braun hand blender	0.7430	42
Durabuilt hand vacuum	0.6615	53
B&D weed trimmer	0.6469	37
Krups cheese grater	0.6460	47
Mini Pro hair dryer	0.6354	32
DeWalt sander	0.5816	42
Radio controlled truck	0.5796	37
Battery operated toothbrush	0.5668	53

that the SKIL screwdriver is similar to a cheese grater? Indeed they are, sharing 64% of the same customer need weighted sub-functions and approximately half of the actual sub-functions. How is knowledge of the cheese grater's similarity useful? Possibly a solution principle in the cheese grater could be used in the power screwdriver to improve its performance, or spur another creative solution. More generally, though, product families lead designers to consider products and their solutions that are not necessarily intuitive or that would have been otherwise ignored. This could prompt the adoption of a surprising technology in the product.

Note that the percentage of sub-functions in common does not decrease as the similarity index. Coupling functionality and customer needs provides a clearer picture of how products relate to each other.

Recall the dominant flow heuristic identified an electrical supply module for the power screwdriver. This product family presents a list of potential products that could utilize such a module. The universal electrical supply module could provide the electricity and actuation of electricity to drive units that function as screwdrivers, blenders, vacuums, graters, small sanders, trimmers, toys or toothbrushes. The method provides greater insight into the relationships between different products.

Aggregate Customer Need Ratings

In this phase of the methodology, modular product architectures are assessed to see how well a module meets customer needs. A functional definition of a module, such as the one reviewed in Section 3, is required as input to this step, along with a normalized product-function matrix.

Step 5: Compute the aggregate customer need rating

The strategy of Step 5 is to use the customer need ratings for the sub-functions to compute a overall, or aggregate, customer need rating for a group of sub-functions defining a module. To do this, we first compute the sum of the products of the customer need ratings (for the sub-functions of the module under consideration) for each product in the normalized product-function matrix. This takes the form

$$s'_j = \sum_{p=1}^{n}\left(\prod_{i=1}^{f_j} v_{ip}\right), \tag{7}$$

where n is the number of products, g is the number of modules in the product under investigation, $j = 1..g$ indicates the module to which the value corresponds and f_j is the number of sub-functions in module j. Here, v_{ip} is the element of **N** corresponding to the ith sub-function of module j in the pth product. A non-zero contribution to the function importance s'_j relies on the existence of all sub-functions in a product.

To maintain the customer need scale of 6, the f_jth root of the multiplicative product $\prod_{i=1}^{f_j} v_{ip}$ is taken. The sum is divided by the number of products, n. Equation 7 becomes

$$s_j = \frac{1}{n}\sum_{p=1}^{n}\sqrt[f_j]{\prod_{i=1}^{f_j} v_{ip}}\,. \tag{8}$$

The value s_j the aggregate customer need rating for module j. The measure of s_j is still on a 6 point scale. For example, if s_2 has a value of 6, the combination of sub-functions in module 2 is a "must" for all products in the repository.

The resolution of s_j

Customer need ratings initially have a resolution of 1, on a 6 point integer scale. Equations 1-8 generate values of s that are not integers. The sensitivity of the aggregate customer need ratings to a change in initial customer need rating of one point is evident in at least the second decimal place (McAdams et al., 1998).

Choice of repository

A brief clarification of the type of repository used for this step is necessary. The equations developed above in Step 5 are applicable to an entire product repository. However, in practice, it is more meaningful to operate on a subset of the product repository as defined by a product family set. This produces values of s that are less diluted; showing differences well above the resolution limit discussed above. To do this, the product family subset is renormalized (using their original product vectors) following the procedure of Step 3.

Example: Computing the aggregate customer need ratings for a modular product

Consider the following scenario. Company A wants to enter the power tools market with a power screwdriver product. They

Sub-function \ Product	Radio Controlled Truck	Mini Pro Hair Dryer	De walt sander	Krups Cheese Grater	SKIL Power Screwdriver	Durabuilt Hand Vacuum	B&D Weed Trimmer	B&D Cordless	Screwdriver	Battery Op. Toothbrush
actuate electricity	2.119	1.298	2.582	0.822	7.915	1.253	9.796	1.044	1.117	5.367
allow DOF of solid	0	0	0	0	0.989	0	0	1.044	0	0
assemble product	0	0	0	0	0	1.253	0	0	0	0
change electricity	0	0	0	0	0	0	0	0	0	1.073
change rotation	11.3	0	0	3.288	5.936	0	0	19.83	8.938	10.73
clean product	0	0	0	4.932	0	0	0	0	1.117	4.293
convert elect. to rotation	13.42	19.47	7.746	11.51	13.85	15.04	10.78	16.7	10.06	7.513
convert electricity to heat	0	3.893	0	0	0	0	0	0	0	0
convert rotation to pneum.	0	5.191	7.746	0	0	11.28	0	0	0	0
convert rot. to vibration	0	0	0	0	0	0	0	0	1.117	0
couple solid	0	0	0	0	0.989	0	0	6.263	0	0
disassemble product	0	0	0	1.644	0	0	0	0	0	0
dissipate sound	0	7.787	3.443	2.466	0	0	0	0	3.352	3.22
dissipate heat	0	0	5.164	0	0	3.76	0	1.044	0	0
dissipate rotation	3.531	6.489	0	0	1.979	3.76	1.959	3.131	5.586	6.44
dissipate vibration	7.062	0	3.443	0	0	5.014	0	1.044	3.352	4.293
distribute gas	0	6.489	0	0	0	0	0	0	0	0
export solid	0	0	0.861	5.754	0	5.014	0.98	0	0	0
guide gas	0	6.489	5.164	0	0	1.253	0	0	0	0
guide solid	0	0	0	8.219	0	0	1.959	0	0	0
guide translation	8.475	0	0	0	0	0	0	0	0	0
import electricity	0	1.298	0.861	0	0	0	0	0	0	1.073
import gas	0	1.298	0	0	0	0	0	0	0	0
import human force	3.531	5.191	6.024	10.69	8.904	7.521	5.878	4.175	12.29	9.66
import human hand	0	7.787	12.91	4.932	9.894	7.521	0	4.175	5.586	9.66
import liquid	0	0	0	0	0	0	0	0	1.117	0
import signal	3.531	0	0	0	0	0	0	0	0	0
import solid	4.237	0	0.861	0.822	0	6.267	0.98	0	1.117	0
indicate status	0	0	0.861	0	0	5.014	0	0	0	0
maintain device	0	0	0	1.644	0	0	0	0	4.469	3.22
mix liquid and solid	0.706	0	0	0	0	0	0	0	0	0
position product	0	1.298	0	0	0	0	8.817	0	4.469	2.147
position solid	0	0	6.885	4.932	0	0	0	0	0	0
refine gas	0	1.298	0	0	0	1.253	0	0	0	0
regulate electricity	1.412	6.489	0.861	0	3.957	1.253	0	1.044	0	4.293
regulate rotation	0	0	0	0	2.968	0	0	1.044	5.586	0
regulate translation	0	0	0	0	2.968	0	0	1.044	0	0
remove solid	0	0	5.164	11.51	0	0	5.878	0	12.29	0
rotate solid	0	0	0	0	0.989	0	0	1.044	1.117	0
secure rotation	0	0	0	0	4.947	0	0	4.175	0	0
secure solid	0.706	0	11.19	4.932	4.947	0	0.98	5.219	0	0
sense control	0	6.489	0	0	0	0	0	0	5.586	4.293
separate signal	0.706	0	0	0	0	0	0	0	0	0
separate solid	0	0	5.164	0	0.989	2.507	0	3.131	0	0
stop chemical energy	0	0	0	0	0	1.253	0	0	0	0
stop gas	0	0	0	0	0	1.253	0	0	0	0
stop liquid	0	0	0	0	0	0	0	0	11.17	0
stop rotation	0	0	0	0	0	0	0	5.219	0	0
stop solid	0	1.298	0	0	0	0	0	0	0	0
stop heat	0	1.298	0	0	0	0	0	0	0	0
store electricity	0	0	0	8.219	9.894	15.04	5.878	13.57	0	0
store product	0	1.298	0	0	0	3.76	0	0	0	1.073
store solids	0	0	2.582	0.822	0	1.253	0	0	0	0
supply electricity	0	0	0	0.822	0.989	1.253	0	2.088	1.117	0
transmit electricity	0	0	0	0.822	0	0	0	0	1.117	0
transmit human force	0	0	6.024	0	0	0	0.98	0	0	0
transmit rotation	0	0	0	0	0.989	0	5.878	2.088	1.117	1.073
transmit heat	0	1.298	0	0	0	0	0	0	0	0
transport solid	0	0	2.582	0	0	0	0	0	0	0

Figure 8 Normalized product-function matrix, N, for the power screwdriver product family.

Table 4 Aggregate customer need ratings for the SKIL power screwdriver family.

Modules (from heuristics): *functional description*	Aggregate CN rating	
	Top 10 products	Top 5 products
electricity to torque: *convert electricity to torque, change torque, transmit torque*	2.225	3.565
coupling/decoupling: *couple solid, secure solid, separate solid*	0.619	1.238
supply electricity: *store electricity, supply electricity, actuate electricity, regulate electricity*	0.835	1.67
torque transmission: *change torque, transmit torque, rotate solid, dissipate torque*	0.764	1.529
manual use: *secure rotation, dissipate torque, allow rotational DOF*	0.444	0.889
positioning: *regulate rotation, regulate translation*	0.4	0.8

develop a functional model of their power screwdriver (the function structure in Fig. 4) and apply the module heuristics to identify modules. The design team has done its homework – collected customer need ratings, created a product repository and computed a product family (as shown in the Step 4 example) based on its functional model of the power screwdriver. Now they want to know which of the identified modules best meet their customers' needs. Following Step 5, here is what they find.

First a subset of the original repository identified by the top ten products of the product family is renormalized to form a smaller product family repository. This repository is shown in Fig. 8. Then equation 8 is used to compute the aggregate customer need ratings for the six modules identified of the power screwdriver (shown in Table 4).

Table 4 shows that for the top ten product subset, the *electricity to torque* module has the highest aggregate customer need rating. In fact, this computation is supported by power screwdrivers currently on the market. These three sub-functions are typically found as a module, utilizing standard components and having minimal interactions with other parts of the product. For company A, the aggregate customer need rating directs their development of modules. In other words, it says to Company A that the *electricity to torque* module could be used in several different products. In fact, it should guide designers to see if such a module is available off the shelf or if they should develop the module and sell it to other manufacturers. Along the same lines, the aggregate customer need rating could support the development of a portfolio of products based on one or more common core modules. The next highest aggregate customer need value is for the *supply electricity* module. Though its value indicates it is on the same level as a supporting function, it offers the next highest payoff in terms of customer needs.

Aggregate customer need ratings also offer input to decisions on overlapping modules. In the case where the heuristics have identified overlapping modules, the aggregrate rating indicates which module best addresses customer needs. Specifically, Table 4 shows that of the overlapping modules *electricity to torque* and *torque transmission*, the former best meets customer needs for this family of products. One possible outcome here is that Company A makes a modular power screwdriver that consists of an *electricity to torque* drive unit and a *supply electricity* power supply unit. This creates a novel power screwdriver design that is supported by customer needs.

As a comparison, *s* is computed for the top five *functionally* similar products to the power screwdriver (recall the calculation shown in the Step 4 example). This narrower repository augments the values since the products share more sub-functions in common. Notice now that four modules are considered more important than supporting.

Reality Check

The method of quantitative functional models is based on a body of empirical data from customer needs and functional descriptions of actual products. The method allows designers to make inferences about new products based on this data. While techniques for costing components are well developed, this method offers a novel way to choose modules based on customer needs and product function.

There are limitations, however. Functional models and thorough sets of customer needs are needed to form the repository – a non-trivial exercise. Also, the functional models must be expressed in a consistent language (i.e., their sub-function description) to make their manipulation meaningful. The correlation of customer need to sub-function is still not precise, and it is possible that different designers may produce different product vectors. However, identifying the functionality should be greatly improved with a consistent language of functions and flows.

5. SUMMARY

This article presents a methodology for transforming customer need rankings and function structures into quantitative functional models. The quantitative functional models are arranged in a product repository used to compute product families and aggregate customer need ratings for modules. It also represents a novel way to archive and communicate product design knowledge.

The quantitative functional model method is presented as it applies to product architecture. The product-function matrix provides a normalized look at a set of products with different functions and different customer needs. This approach, in essence, allows customer needs to be gathered by many individuals and organized into a cohesive repository. The product-product matrix produces sets of products that are closely related in terms of

functionality and customer need. These sets are called product families and are noted to offer opportunities for various types of modularity, especially component sharing and component swapping. Within the product family, the aggregate customer need rating quantifies how the combination of the sub-functions defining a module satisfies customer needs. The module heuristic methods described in Section 3 are used to define modules.

Taken together, the heuristic method and the quantitative functional model method provide a way to identify modules, assign a customer need rating to those modules and identifying other similar products. This analysis tells the designer which sub-functions make sense to combine into modules and, then, which ones to focus on to meet the customer needs. These two methodologies provide a unique justification for a modular product architecture development.

ACKNOWLEDGMENTS

The research reported in this paper was partially supported by an Engineering Doctoral Fellowship through the College of Engineering and by a Continuing Fellowship at The University of Texas at Austin and the Department of Basic Engineering at University of Missouri-Rolla. In addition, this work is supported by the National Science Foundation under a NSF Young Investigator Award, Ford Motor Company, Desktop Manufacturing Corporation, Texas Instruments, W.M. Keck Foundation, and the June and Gene Gillis Endowed Faculty Fellow in Manufacturing. Any opinions or findings of this work is the responsibility of the authors, and does not necessarily reflect the views of the sponsors or collaborators.

REFERENCES

Allen, K. and Carlson-Skalak, S., 1998, "Defining Product Architecture During Conceptual Design," Accepted to Proceedings of DETC98, DETC98/DTM-5650, Atlanta, GA.

Altshuller, G., 1984, *Creativity as an Exact Science*, Gordon and Branch Publishers.

Bradley, S. and Agogino, A., 1994, "An Intelligent Real Time Design Methodology for Component Selection: An Approach to Managing Uncertainty," *Journal of Mechanical Design*, 116:980-988.

Cutherell, D., 1996, "Chapter 16: Product Architecture," *The PDMA Handbook of New Product Development*, M. Rosenau Jr., et al., ed., John Wiley and Sons.

Griffen, A. and Hauser, J., 1993, "The Voice of the Customer," *Marketing Science*, 12(1):1-27.

Gu, P., Hashemian, S. and Sosale, S., 1997, "An Integrated Modular Design Methodology for Life-Cycle Engineering," *Annals of the CIRP*, 46(1):71-74.

Gupta, S. and Krishnan, K., 1997, "Product Family-Based Integrated Component and Vendor Selection," Department of Management Working Paper, The University of Texas at Austin.

Krishnan, V., Singh, R., and Tirupati, D., 1996, "A Model-Based Approach for Planning and Developing a Family of Technology-Based Products," Department of Management Working Paper, The University of Texas at Austin.

Little, A., 1997, *A Reverse Engineering Toolbox for Functional Product Measurement*, Master Thesis, The University of Texas at Austin.

Little, A., Wood, K., and McAdams, D., 1997, "Functional Analysis: A Fundamental Empirical Study for Reverse Engineering, Benchmarking and Redesign," *Proceedings of the 1997 Design Engineering Technical Conferences*, 97-DETC/DTM-3879, Sacramento, CA.

Marks, M., Eubanks, C., and Ishii, K., 1993, "Life-Cycle Clumping of Product Designs for Ownership and Retirement," *Proceedings of the ASME Design Theory and Methodology Conference*, DE-Vol. 53.

McAdams, D., Stone, R., and Wood, K., 1998, "Product Similarity Based on Customer Needs," *Proceedings of DETC98*, DETC98/DTM-5660, Atlanta, GA.

McAdams, D., Stone, R., and Wood, K., 1999, "Functional Interdependance and Product Similarity based on Customer Needs," To Appear *Research in Engineering Design*, 11(1).

Murdock, J., Szykman, S. and Sriram, R., 1997, "An Information Modeling Framework to Support Design Databases and Repositories," *Proceedings of DETC'97*, DETC97/DFM-4373, Sacramento, CA.

Newcomb, P., Bras, B., and Rosen, D., 1996, "Implications of Modularity on Product Design for the Life Cycle," *Proceedings of the 1996 ASME Design Theory and Methodology Conference*, 96-DETC/DTM-1516, Irvine, CA.

Otto, K. and Wood, K., 1996, "A Reverse Engineering and Redesign Methodology for Product Evolution," *Proceedings of the 1996 ASME Design Theory and Methodology Conference*, 96-DETC/DTM-1523, Irvine, CA.

Otto, K. and Wood, K., 1997, "Conceptual and Configuration Design of Products and Assemblies," *ASM Handbook, Materials Selection and Design*, Vol. 20, ASM International.

Pahl, G. and Beitz, W., 1988, *Engineering Design: A Systematic Approach*, Springer-Verlag.

Pimmler, T. and Eppinger, S., 1994, "Integration Analysis of Product Decompositions," *Proceedings of the ASME Design Theory and Methodology Conference*, DE-Vol. 68.

Rosen, D., 1996, "Design of Modular Product Architectures in Discrete Design Spaces Subject to Life Cycle Issues," *Proceedings of the 1996 ASME Design Engineering Technical Conferences*, 96-DETC/DAC-1485, Irvine, CA.

Schmidt, L. and Cagan, J., accepted 1997a, "Optimal Configuration Design: An Integrated Approach Using Grammars," To appear in *Journal of Mechanical Design*.

Schmidt, L. and Cagan, J., 1997b, "GGREADA: A Graph Grammar-Based Machine Design Algorithm," *Research in Engineering Design*, 9(4):195-213.

Stone, R., 1997, *Towards a Theory of Modular Design*, Doctoral Thesis, The University of Texas at Austin.

Stone, R., Wood, K., Crawford, R., 1998, "A Heuristic Method to Identify Modules from a Functional Description of a Product," *Proceedings of DETC98*, DETC98/DTM-5642, Atlanta, GA.

Stone, R., Wood, K., Crawford, R., 1999, "A Heuristic Method for Identifying Modules for Product Architectures," To Appear *Design Studies*.

Stone, R. and Wood, K., 1999, "Development of a Functional Basis for Design," *Proceedings of DETC99*, DETC99/DTM-8765, Las Vegas, NV.

Ullman, D., 1997, *The Mechanical Design Process 2nd ed.*, McGraw-Hill.

Ulrich, K. and Eppinger, S., 1995, *Product Design and Development*, McGraw-Hill.

Ulrich, K. and Tung, K., 1991, "Fundamentals of Product Modularity," *Proceedings of the 1991 Winter Annual Meeting*, DE-Vol. 39, Atlanta, GA.

Vadde, S., Allen, J., and Mistree, F., 1992, "Catalog Design: Design Using Available Assets," *Proceedings of the 1992 ASME Design Technical Conferences, Advances in Design Automation*, DE-Vol. 44-1, Scottsdale, AZ.

Ward, A., 1989, *A Theory of Quantitative Inference Applied to a Mechanical Design Compiler*, Doctoral Thesis, Massachusetts Institute of Technology.

Ward, A. and Seering, W., 1993, "Quantitative Inference in a Mechanical Design 'Compiler'," *Journal of Mechanical Design*, 115:29-35.