

DETC2006/DAC-99469

A METHOD TO IMPROVE PLATFORM LEVERAGING IN A MARKET SEGMENTATION GRID FOR AN EXISTING PRODUCT LINE

Ronald S. Farrell¹ and Timothy W. Simpson^{2*}
Department of Mechanical & Nuclear Engineering
The Pennsylvania State University
University Park, PA 16802, USA

ABSTRACT

This paper describes a method for improving commonality in a highly customized low volume product line whose members were originally developed one-at-a-time to meet specific customer requirements. Rather than focusing on redesign of the entire product line, which can often be cost prohibitive, the method is part of a strategy to redesign a limited set of component parts that have the highest potential for cost savings. The method involves a four-step process: (1) determine an optimal component solution for each member artifact of an existing market segment grid, (2) test the feasibility of using each optimal component as a platform for the other artifacts, (3) formulate an optimization problem around the feasibility statistics whose solution is a product platform portfolio, and (4) solve the optimization problem for the minimum number of platforms that can adequately span the existing market segment grid. The proposed method is applied to an example involving the redesign of actuator mounting yokes for an existing set of valves that are used in nuclear power plants. The method shows promise for determining a product platform mix that maximizes commonality yet meets performance requirements.

Keywords: product platform, product variety, market segmentation, commonality, redesign

NOMENCLATURE

- a = Yoke Leg Cross-Section Inside Radius
- A = Yoke Leg Cross-Section Area
- b = Yoke Leg Cross-Section Outside Radius
- B_i = Boundary Constraints for Grid Member i
- c = Yoke Leg Cross-Section Angle
- C_j = Candidate Platforms for Grid Member i

- f_1, f_2 = Valve Primary Mode Natural Frequencies
- f_{MIN} = Minimum Allowed Natural frequency
- F_i^* = Optimal Objective Function for Grid Member i
- i = Market Segment Grid Member Ordinal
- j = Optimal Component Ordinal for Grid Member i
- n = Number of Market Segment Grid Members
- N = Number of Used Platforms
- N^* = Minimum Required Number of Used Platforms
- P_i = Performance Constraints for Grid Member i
- S = Market Segment Grid Ordinals Array
- S_A = Allowable Stress
- T = Tradeoff Metric for Platform Portfolio Optimization
- X_i = Design Variables for Grid Member i
- X_i^* = Optimal Component Solution for Grid Member i
- Y = Platform Portfolio Design Variables
- Y^* = Optimal Platform Portfolio Design Variables
- α = Tradeoff Metric Proportionality Constant
- σ_1, σ_2 = Yoke Leg Critical Stresses

1. INTRODUCTION

Commonality is recognized as an effective way to achieve economies of scale and scope, and product families have been successfully employed by various companies such as Sony [1], Black & Decker [2], Lutron [3], and Airbus [4] to address the challenge of providing variety for the marketplace while maintaining commonality between products. A product family is a group of related products that share common features, components, and subsystems; yet satisfy a variety of market niches. The set of common parameters, features, or components that remain constant from product to product within a given product family is referred to as the product platform. The product platform provides the basis for the product family, which is derived through the addition, substitution, or exclusion of one or more modules from the platform [5-9] or by scaling the platform in one or more dimensions [10-12].

¹ Graduate Research Assistant.

^{2*} Associate Professor, Member ASME, and Corresponding Author. 329 Leonhard Building, Penn State University, University Park, PA 16802 USA, Phone/fax: (814) 863-7136/4745, Email: tw8@psu.edu.

In product platform design, it is common to map overall design requirements into an appropriate *market segmentation grid* [2], as shown in Figure 1. The grid allows for identification of potential leveraging opportunities for the product platform to satisfy effectively a variety of market segments. It is traditional to employ horizontal, vertical, and beachhead approaches as illustrated in Figure 1 to enable effective platform leveraging both within and across different market segments.

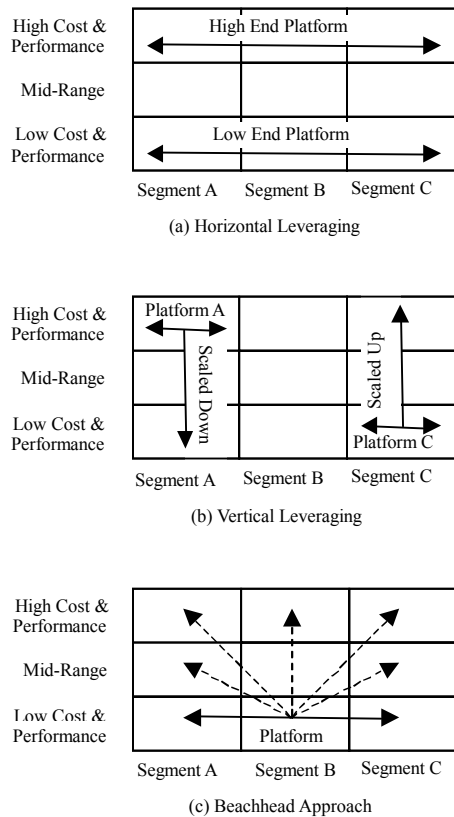


Figure 1. Market Segmentation Grid and Platform Leveraging Strategies [2]

A company’s product line could be based totally or in part on a *product platform portfolio*, which is a collection of product platforms that covers the entire market segment grid. A design process centered on a product platform portfolio could result in a cost-effective product development system that can provide the variety that the market demands. An agile manufacturing system combined with a well-designed portfolio can efficiently and proactively change to meet future demand.

In most research, the product platform portfolio and the design and platform variables are established prior to detailed design; however, there has been research involving optimization prior to detailed platform design [13]. Some research endeavors to distinguish between design variables, which change with each family member, and platform variables, which are constant within the family [11,14,15]. For instance,

D’Souza and Simpson [14] employ a non-dominated sorting Genetic Algorithm (GA) to optimize the balance between commonality and performance. Effectively, this optimizes the extent to which the design variables cover the targeted market segments. De Weck, et al. [16] present a method to determine the optimal number of product platforms to maximize overall product family profit with simplifying assumptions. The method is demonstrated for a hypothetical automotive vehicle line that is required to fill seven market segments. The portfolio can vary from one to seven platforms: the seven-platform case corresponds to no leveraging while the single platform case corresponds to the maximum leveraging possible. The method computes the profit for each portfolio from the set of seven, and chooses the one that yields the highest profit as the optimal one; a three-platform portfolio is determined to be optimal for their example. Fujita and Yoshida [17] advocate the simultaneous design of multiple products. Assuming a modular architecture, they propose a simultaneous optimization method for both module combination and module attributes of multiple products. The method considers cost, profit, commonality, and similarity, and hybridizes a genetic algorithm, a mixed-integer programming method with a branch-and-bound technique, and a constrained nonlinear programming method. This is an extension of other work [18] where optimization of module combination and module attributes are treated separately. In other research, product family selection is optimized based on a performance loss function [19], or optimization is based on combining business and engineering decisions [20]. In a paper involving modular architecture artifacts, a GA-based method is used to optimize module sharing and creation of new modules [6]. In work by Hernandez, et al. [21], the goal is to minimize the impact of commonality on performance using the concept of space of customization.

Development of the method proposed in this paper is motivated by the need to improve commonality in a highly customized low volume product line whose members were originally developed one-at-a-time to meet specific customer requirements, as it can be difficult to achieve and maintain commonality. When a product is unique, it results in high development and production costs that are difficult to predict, and in long and uncertain production times. A manufacturer of these products may eventually develop a quasi-standard product line, but since the line is designed one custom product at a time, the full spectrum of product offerings is rarely reviewed to ensure that it is optimal for the business [22]. Focusing on custom products can result in “a failure to embrace commonality, compatibility, or standardization” [23], leading to a proliferation of products and parts with increasing costs and overhead. The failure potential increases by degree for highly customized product lines and is even greater for small firms.

Redesign of a ‘one-at-a-time’ product line can be cost prohibitive, especially for a small firm. However, what may be justifiable is a strategic redesign of a limited set of component parts that have the highest potential for cost saving. A component part redesign effort can employ the product platform

approach, and when applied across the market segment, a *component product platform portfolio* results.

This paper presents a method for optimizing a component product platform portfolio so that market segment grid platform leveraging is maximized. It assumes a product line already exists that fits into an existing market segment grid, and the objective is to redesign individual components that are common to all members of the product line but lack commonality between the members. The method does not rely on the traditional leveraging strategies (horizontal, vertical, and beachhead), as no constraints are imposed on the leveraging strategy. The goal is to minimize manufacturing cost by minimizing the number of component product platforms required to span the market segment grid without sacrificing product performance or customer perceived variety, but this can be challenging because it involves a tradeoff between minimum cost and maximum performance. An important precursor is a design method that can determine component design parameters that optimize performance for a single artifact from the market segment grid. The proposed method is described in detail in Section 2 as a four-step process. In Section 3, an example involving the design of yokes on nuclear grade valves is presented that follows the four steps. Section 4 gives closing remarks and discusses the example problem results and potential future work.

2. PROPOSED METHOD

The pre-existence of design artifacts that make up a market segment grid is assumed. The artifacts should have a common component, each of unique design that is to be designed or redesigned with the goal of maximizing commonality throughout the market segment. Improved commonality is achieved through the design of component product platforms, where each platform forms the basis for common component design for a group of artifacts. Then, several platforms are required to address all components in the grid, and this collection of platforms is defined as the component product platform portfolio. The ultimate goal is to determine the portfolio that optimizes the commonality and performance of the family in the market segmentation grid.

As a precursor, we assign an ordinal (i) to each artifact in the market segment grid. Then, the grid is symbolized by an array (S) with n members, and the elements of S contain sequential numbers from 1 to i to n . At a minimum, a design strategy must exist that can be implemented on artifact i to determine optimal critical component design parameters (X_i^*) which yield an optimal objective function (F_i^*). Description of the four steps in the proposed method follow.

Step1: Determine an optimal component solution for each member (i) of the market segment grid.

Each solution consists of an optimal objective function (F_i^*), an array of optimal design variables (X_i^*), an array of design value constraints associated with performance (P_i), and an array of constraints (B_i) associated with the upper and lower

bounds on X_i . The values of F_i^* , X_i^* , and B_i are saved for subsequent steps. The method assumes that the optimal product platform portfolio consists of a subset of the resulting optimal designs.

Step 2: For each optimal component, test the feasibility of using it as a platform on each market segment member.

Given the n members, $(n^2 - n)$ tests are required, and $n-1$ tests are associated with each optimal component. Notice that testing a component against its own member is not required as its feasibility is assured in advance. These $n - 1$ tests require applying the component's optimal solution X^* to each market segment member, and assessing whether the member's constraints (B_i and P_i) are satisfied.

In order to save computing effort, we perform the testing in two phases. In Phase 1, we test only the bounds constraints (B_i), and in Phase 2, we test the remaining constraints (P_i). The two-phase testing procedure can save effort because Phase 1 does not require reformulation of the optimization problem since only upper-lower bounds are involved, and most likely fewer tests are required during the more computationally expensive Phase 2 because the bounds test can eliminate many candidate component-member combinations.

Step 3: Formulate an optimization problem, whose solution is a product platform portfolio.

From the previous step's result, we construct arrays of candidate platforms (C_j) for each optimal component (j), then each C_j contains the grid member ordinals (i) for which the component (j) satisfies constraints. Then, the design variables (Y) consist of n elements, one for each component (j). The value of a design variable (Y_j) equals the index into the component's C_j array. Each element of Y is bounded from one to the size of C_j . The design variables define which component is used as a platform for each grid member such that the n used platforms are given by $C(Y)$.

The main optimization objective is to minimize the number (N) of required component product platforms. There can be other sub-objectives such as minimizing the cost of portfolio implementation, maximizing performance not captured during component optimization, or constraining leveraging. The formulation assumes these sub-objectives are captured by a single tradeoff metric (T). Satisfying the sub-objective may require a tradeoff with the primary objective in that any improvement in T may require an increase in N and vice-versa. N and generally T are functions of $C(Y)$, and N equals the number of unique ordinals in $C(Y)$. For instance, if $C(Y)$ equals $\{1, 2, 1, 3, 2\}$, then N equals 3. The optimization problem is thus stated as follows:

$$\text{Minimize } \{N + \alpha T\} \quad (1)$$

Subject to the upper and lower bounds on Y .

where α is a scaling factor to adjust the weight on T .

Step 4: Solve the optimization problem.

The solution requires a zero-order algorithm such as the Simulated Annealing (SA) Algorithm or the Genetic Algorithm (GA), which is capable of addressing integer design variables. Given the optimal solution (N^*), the components to use as platforms are given by the unique ordinals in $C(Y^*)$, and the platforms to use for each grid member is given by $C(Y^*)$.

3. EXAMPLE PROBLEM

Implementation of the four-step process is demonstrated using an example involving the design of yokes in a market segment of nuclear grade valves. Before demonstrating the portfolio optimization process in Section 3.2, the fundamentals of valve operation and valve design are discussed.

3.1. Valve Fundamentals

Valves are common components in nuclear plant piping systems, and many of them are custom built to respond to specific design and accident scenarios. The example considers a targeted market segment consisting of automatically actuated gate valves such as those shown in Figure 2. Gate valves are used to isolate flow, and they can accomplish this better than most other valve types because (1) they are reliable due to their simple design, (2) they require less actuator force while closing against flow, (3) they introduce minimal flow resistance while open, and (4) differential pressure across the gate aids in sealing off flow.

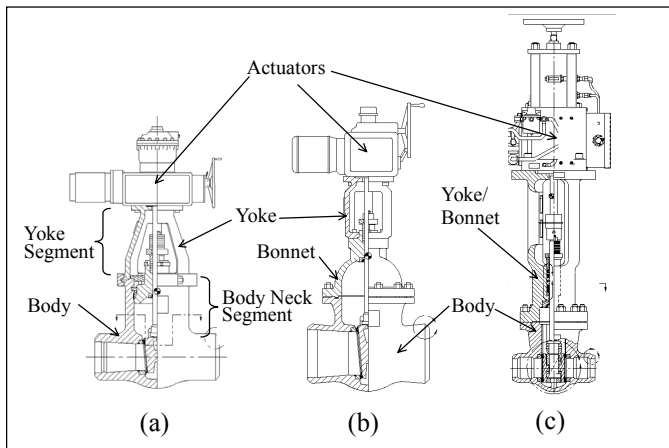


Figure 2. Typical Gate Valves (Courtesy of Flowserve Corporation): (a) Size 6, Class 900 Flex Wedge, (b) Size 8, Class 150 Flex Wedge, (c) Size 4, Class 150 Double Disc

As an example, Figure 3 shows a flex wedge gate in the closed position. Flow isolation and sealing are achieved through bearing contact between the gate and the seat ring that is welded into the valve body. The actuator provides thrust to the gate and must provide enough force to overcome frictional and flow induced drag and to wedge the gate into the seat. Once closed, differential pressure can develop, which forces the gate against the seat on the downstream side of the valve. Then,

both differential pressure and wedging forces are available to affect a seal between the gate and seat. Often, differential pressure alone provides adequate seat bearing stress to seal. Due to design symmetry, a flex wedge gate valve is bi-directional in that it can isolate flow moving in either direction.

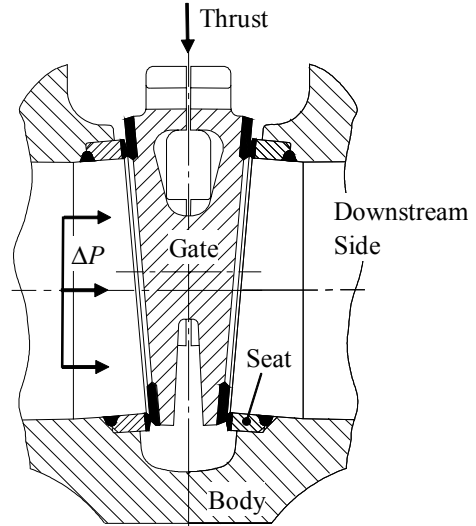


Figure 3. Flex Wedge Gate Valve Sealing

The actuator can be manually, electrically, pneumatically, or hydraulically energized, and its function on a gate valve is to open and close the valve by raising and lowering the stem. The yoke is one of the valve components that often requires modification to respond to specific customer requirements such as loading associated with an anticipated seismic event, sensor or control installation, and mounting and support for the specific actuator size and type. As shown in Figure 4, the yoke consists of top and bottom mounting flanges joined by two legs. This example has a transition neck between the legs and the actuator mounting flange.

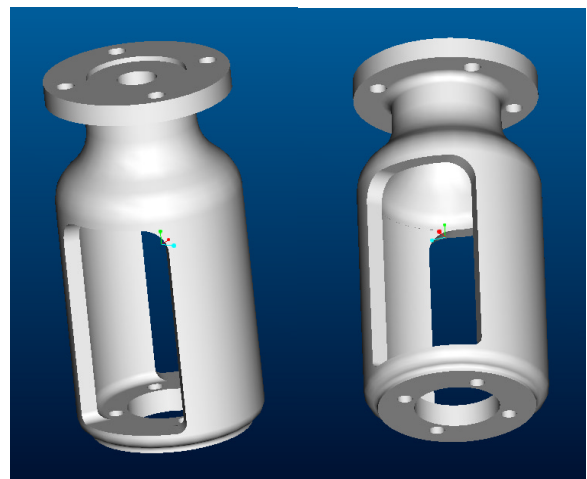


Figure 4. Solid Model Views of a Typical Yoke

In our previous work [24], we develop a product platform portfolio for the yokes that mount the automatic actuators on the nuclear grade gate valves using the *Product Platform Concept Exploration Method* (PPCEM) [25]. The step-by-step application of the PPCEM is demonstrated through the creation of product platforms consisting of modular, scalable valve yoke cross-sections for use on the gate valves. First, the market segmentation grid is constructed based on past sales data that sets a target range of valve designs for the yoke redesign. A set of design parameters is established that reasonably meet the past requirements of the targeted segment to benchmark against the redesigned yoke cross-section platforms. Parameters include actuator weight and center of gravity, yoke construction material, yoke height, actuation load, seismic load, allowed natural frequency of vibration, allowable stress criteria, and operating temperature. In addition, a yoke casting pattern modular architecture has been proposed as shown in Figure 5.

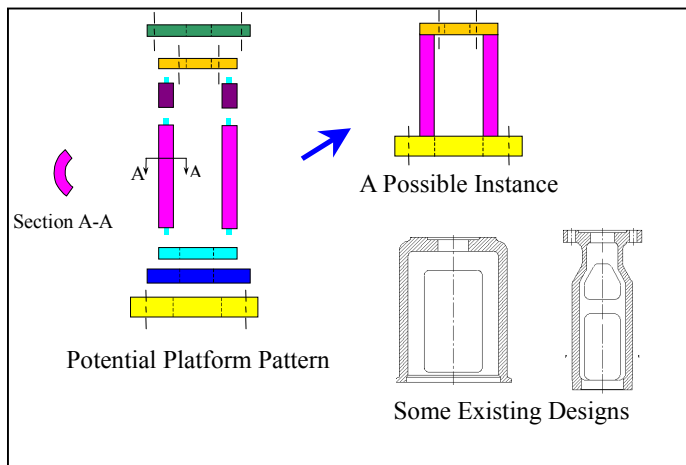


Figure 5. Modular Platform Pattern Concept for Yoke Cross-Sections [24]

In subsequent work [26], we introduced a prototypical web-based specification system for the yokes. The platform leveraging strategy was refined, more platforms were created, and a body component was included to demonstrate how an entire valve external structure can be addressed. Figure 6 shows the resulting leveraging strategy, which employs horizontal leveraging on two characteristics (size and class). In addition, horizontal leveraging is employed across two types of gate valves: the flex wedge, and the double disc. Note that this leveraging strategy was chosen simply by judgment, without consideration of optimal cost effectiveness.

This previous work forms the basis for the market segmentation grid and the platform design strategy employed in the example in Section 3.2. Here, we add one size (size 12) and one class (class 1500) to the grid, and apply the proposed method in an attempt to achieve optimal leveraging. In addition, the underlying platform optimization problem is refined such that platform solutions are obtained that better satisfy design criteria optimally. It is important to realize that

(1) a bottom-up platform approach [24] is employed in that an existing valve line is redesigned to improve commonality, and (2) the subject valve line involves low volume demand yet can involve very diverse design requirements.

Size 3 & 4 Class 600 & 900		Size 6 & 8 Class 600 & 900		No Platform Developed	900
Size 3 & 4 Class 150 & 300		Size 6 & 8 Class 150 & 300			600
3	4	6	8	10	300
					150

Connecting Pipe Size (inches)

Pressure Class

Figure 6. Previous Market Segmentation Grid Leveraging [26]

Then, the artifact addressed in the example is the valve, and the common, but uniquely designed, component is the yoke leg for each valve. The existing artifacts contain yoke legs with variously shaped cross-sections, and it is desired to design the legs based on the common shape shown in Figure 7. The figure shows the design parameters that effect critical performance, including valve fundamental natural frequency of vibration and yoke leg stress. The valve market segmentation grid is defined by type, consisting of double disc gates (DD) and flex wedge gates (FW), pressure class, which relates to the allowable working pressure of the process fluid, and size, which relates to the connecting piping nominal size.

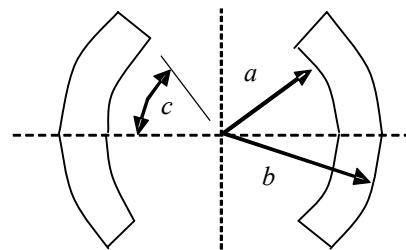


Figure 7. Generalized Yoke Legs Cross-Section

3.2. Example Product Platform Portfolio Optimization

The example uses Microsoft Excel™ to perform calculations and store platform design parameters and a Microsoft Access™ database to track optimization statistics. A single workbook contains a spreadsheet of parameters for each member of the market segment grid, and other workbooks contain macros used to access performance and to conduct the optimizations. Table 1 is a pivot table from the database that lists grid-defining parameters and shows the ordinals that define

the valve artifact members. As can be seen, the grid consists of a total of 60 different valves.

Table 1. Sample of Market Segment Grid Artifact Ordinals

Type	Class	Size					
		3	4	6	8	10	12
		Ordinal	Ordinal	Ordinal	Ordinal	Ordinal	Ordinal
DD	150	1	2	3	4	5	6
	300	7	8	9	10	11	12
	600	13	14	15	16	17	18
	900	19	20	21	22	23	24
	1500	25	26	27	28	29	30
	Total						
FW	150	31	32	33	34	35	36
	300	37	38	39	40	41	42
	600	43	44	45	46	47	48
	900	49	50	51	52	53	54
	1500	55	56	57	58	59	60

Step 1: Optimal Yoke Leg Cross-Sections

Each artifact's yoke leg cross-section size was optimized using Excel's Solver Add-in. We desire the resulting cross-section dimensions to be of a specified precision, i.e., lengths in eighth-inch increments and angles in five-degree increments, and the integer programming capabilities of the Excel Solver Add-in make this possible. The optimization problem is:

$$\text{Minimize } F = \{A + (f_1 + f_2) - (\sigma_1 + \sigma_2)\} \quad (2)$$

Subject to:

$$\begin{aligned} P_1 &= (1 - f_1 / f_{\text{MIN}}) \leq 0 \\ P_2 &= (1 - f_2 / f_{\text{MIN}}) \leq 0 \\ P_3 &= (\sigma_1 / S_A - 1) \leq 0 \\ P_4 &= (\sigma_2 / S_A - 1) \leq 0 \\ B_1 &= a_{\text{MIN}} - a \leq 0 \\ B_2 &= a - a_{\text{MAX}} \leq 0 \\ B_3 &= b - b_{\text{MAX}} \leq 0 \end{aligned}$$

The main objective in this step is to minimize the yoke leg's cross-sectional area. This objective closely matches the goal of an expert valve designer. The presence of the performance parameters in the objective function force the solution to satisfy the performance constraints as close as possible to their limits. The bounds constraints are based on proper fit of the yoke on the subject valve artifact. Fit is governed by the size and type of connection such as a flanged or clamped connection, and the yoke must conform to the artifact's existing connection design.

It is known that the resulting solution is not necessarily the most optimal. However, the solutions are judged acceptable because the objective is accomplished satisfactorily in that the results are at least as good as what an experienced designer can achieve manually through iteration. What is important is that the optimization process satisfactorily automates the design process in that manual iteration is replaced by optimization.

Step 2, Phase 1: Bounds Feasibility Test

During the optimization process in Step 1, optimal yoke leg parameters (a , b , and c) and bounds constraints (B_1 , B_2 , and B_3) for the 60 valves are stored in a database table for use in this step. For the first test phase, each member is tested in turn for bounds constraint feasibility with cross-section parameter input equal to the optimal parameters from all other members. Each member requires 59 tests (the member's own optimal parameters are feasible by definition), and so a total of 3540 tests is required ($= 60^2 - 60$). If the bounds constraints are satisfied, that cross-section represents a candidate platform, and a one "1" is assigned to a test result variable; otherwise, zero "0" is assigned. A star "*" is assigned to the test variable for a member's self-parameter test. For each member, test results are stored in a character string of length 60, and the sequential row upon row combined results for the 60 valve members yields a square matrix of dimension 60, with ones or zeros off diagonal, and stars along the diagonal. Although many tests are required (3540), this sub-step takes little time because reformulation of the optimization problem is not required.

Step 2, Phase 2: Performance Feasibility Test

In Phase 2, each candidate (i.e., those corresponding to "1" from the matrix) is tested to determine if the performance constraints (P1 through P4 in Eq. 2) are satisfied. Because performance constraints are involved, reformulation of the optimization problem is required. Although, the reformulation can be time consuming, the test needs to be conducted over only the candidates. If the candidate meets the constraints, then the test matrix entry is marked with "2". Once complete, the final candidates are those marked with "2".

For this example, the resulting matrix is shown in Appendix A. Note that of the 3540 required Phase 1 feasibility tests, 1195 tests are required during phase 2. The Phase 2 testing yields 374 feasible candidates (those marked with either "2" or "*").

Step 3: Optimization Problem Formulation

The optimization requires building candidate platform arrays from the matrix in Appendix A. Using the first row as an example, three cross-sections satisfy all valve ordinal 1 constraints; therefore, 3 is the upper bound on Y_1 . The row 1 ordinals corresponding to the feasible cross-sections are contained in this candidate array: $C_1 = \{1, 16, 56\}$. Notice that the valve's own cross-section is included (1 in this case), which should always be a condition. As another example, the last matrix row has four feasible cross-sections ($Y_{60} = 4$), and the candidate array (C_{60}) is $\{29, 30, 59, 60\}$. No tradeoff metric (T) is employed, which is equivalent to setting α to zero in Eq. 1.

Step 4: Solving the Optimization Problem

The optimization is solved using a Simulated Annealing (SA) algorithm and a Genetic Algorithm (GA), both of which are adapted from Belegundu and Chandrupatla [27]. Both algorithms contain various tuning parameters, and after a little

tuning, the SA algorithm consistently converges to a viable solution, but the GA could not be trained to obtain good results.

The SA algorithm consistently yields an optimal objective (N^*) of 10 platforms; however, the corresponding platform ordinals can vary widely between solutions. Table 2 shows pivot tables that give two such platform portfolio solutions. A non-zero value in the ‘Qty’ columns corresponds to a chosen platform, and gives the number of valve artifacts that use this platform. The ‘Use’ columns give the platform to use for the corresponding valve artifact. For both solutions for instance, platform 48 is used 9 times, and the valves that use it include {5, 8, 14, 15, 21, 33, 40, 45, and 48} for the first solution, and {3, 5, 7, 8, 14, 41, 45, 48, and 55} for the second solution, and these have {5, 8, 14, 45, and 48} in common. As discussed previously, Table 1 provides a cross-reference between valve ordinals and valve description: for instance, ordinal 48 corresponds to the size 12, class 600 flex wedge gate valve.

Table 2. Platform Portfolio Solutions

		Size ▾											
		3		4		6		8		10		12	
Type ▾	Class ▾	Qty ▾	Use ▾	Qty ▾	Use ▾	Qty ▾	Use ▾	Qty ▾	Use ▾	Qty ▾	Use ▾	Qty ▾	Use ▾
DD	150		16		19		51		59		48		47
	300		35		48		35		59		16		16
	600		37		48		48	6	16		30		30
	900	9	19		19		48		47		16		30
	1500		19		59		16		30		30	9	30
	Total												
FW	150		37		19		48		51	4	35		59
	300	3	37		19		35		48		51		47
	600		19		19		48		59	4	47	9	48
	900		19		57	6	51		59		30		30
	1500		57		51	3	57		51	7	59		30

		Size ▾											
		3		4		6		8		10		12	
Type ▾	Class ▾	Qty ▾	Use ▾	Qty ▾	Use ▾	Qty ▾	Use ▾	Qty ▾	Use ▾	Qty ▾	Use ▾	Qty ▾	Use ▾
DD	150		16		25		48		59		48		16
	300		48		48		59		59		16		16
	600	3	13		48		51	5	16		30		30
	900		37		25		59		47		30		30
	1500	7	25		59		30		30		30	11	30
	Total												
FW	150		37		37		25		51		59		59
	300	4	37		25		51		51		48		47
	600		25		13		48		59	3	47	9	48
	900		13		25	7	51		59		30		30
	1500		48		51	1	57		51	10	59		30

Although some of the portfolio results vary widely, all of the results appear viable. A good reason for the varied results is this: given that a certain platform is suitable for a given valve, that valve’s cross-section may be suitable for the platform’s valve and also for many of the valves that use the platform. In other words, the platforms can have a ‘reflexive’ property in that it may be possible to switch a used platform for the platform of one of its users. At first glance, some of the leveraging apparent in the solutions does not seem viable. For instance, use of platform 48 (from the size 12, class 600 flex wedge gate valve) on valve 8 (the size 4, class 300, double disc gate valve) seems discrepant; however, this is a verified

possibility. A reason for the apparent discrepancy is that the underlying valve artifacts are currently lacking in commonality, as no smooth transition in yoke mounting parameters exists among valve sizes, pressure classes, or types.

Use of the SA algorithm is computationally expensive, but the expense is offset by the simplicity of the example portfolio objective function. This example takes about one minute to run on a 1.8 GHz PC running Windows XP Professional™ and requires about 400,000 function calls. Perhaps the number of function calls could be reduced by using less conservative tuning parameters. Parallel computing techniques could also reduce the overall time to achieve a solution.

4. CLOSING REMARKS

The proposed product platform portfolio optimization method shows promise for determining a component product platform mix that maximizes commonality. The given example involving redesign of a single component that is used by all members of an existing product line is the first attempt at employing the proposed method. The example involves an existing market segmentation grid that is fully populated, and the entire grid is addressed; however, the method can be applied to a partially full grid as well. For instance, the procedure could be used to redesign a component that is common to only a portion of the artifacts in a market segmentation grid, and these artifacts could have random placement within the grid. Although the presented method may be limited to problems similar to the example, variants may emerge for the general design or redesign of multiple components in an emerging or existing product platform portfolio.

Although no tradeoff metric was considered in the example, its proper use could improve the cost effectiveness of the resulting component portfolio, as the minimum number of platforms that satisfy the constraints may not be the most economical to implement. To be effective, the tradeoff metric should capture the cost of using the component platform portfolio relative to the cost of using the existing unique components. The metric should consider costs due to such things as new tooling, raw material, machining, and setup, or the benefit due to such things as reduced overhead, and savings in reduced time to market. In addition, the metric should include sales volume in order to assess payback time, as in general, a component platform portfolio will not be profitable until a sufficient sales volume is realized.

Based on this, future work suggests implementing a general complementary method for constructing a tradeoff metric that considers cost. Proper modeling of costs may help improve consistency of results, or further research efforts may lead to other techniques for improving consistency. Other potential work is to develop an algorithm that explores changing chosen platform parameters such that fewer platforms are required yet portfolio performance is not compromised.

ACKNOWLEDGMENTS

This work was supported in part by the National Science Foundation under Grant No. DMI-0133923. Any opinions, findings, and conclusions or recommendations presented in this paper are those of the authors and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

- [1] Sanderson, S. W. and Uzumeri, M., 1997, *Managing Product Families*, Irwin, Chicago, IL.
- [2] Meyer, M. H., 1997, "Revitalize Your Product Lines Through Continuous Platform Renewal," *Research Technology Management*, **40**(2), pp. 17-28.
- [3] Pessina, M. W. and Renner, J. R., 1998, "Mass Customization at Lutron Electronics - A Total Company Process," *Agility & Global Competition*, **2**(2), pp. 50-57.
- [4] Aboulafia, R., 2000, "Airbus Pulls Closer to Boeing," *Aerospace America*, **38**(4), pp. 16-18.
- [5] Dahmus, J. B., Gonzalez-Zugasti, J. P. and Otto, K. N., 2000, "Modular Product Architecture," *ASME Design Engineering Technical Conferences - Design Theory and Methodology Conference*, Baltimore, MD, ASME, Paper No. DETC2000/DTM-14565.
- [6] Gonzalez-Zugasti, J. P. and Otto, K. N., 2000, "Modular Platform-Based Product Family Design," *ASME 2000 Design Engineering Technical Conferences - Design Automation Conference*, Baltimore, MD, ASME, DETC-2000/DAC-14238.
- [7] Martin, M. V. and Ishii, K., 2002, "Design for Variety: Developing Standardized and Modularized Product Platform Architectures," *Research in Engineering Design*, **13**(4), pp. 213-235.
- [8] Ulrich, K., 1995, "The Role of Product Architecture in the Manufacturing Firm," *Research Policy*, **24**(3), pp. 419-440.
- [9] Zamirowski, E. J. and Otto, K. N., 1999, "Identifying Product Portfolio Architecture Modularity Using Function and Variety Heuristics," *ASME Design Engineering Technical Conferences - Design Theory and Methodology Conference*, Las Vegas, NV, ASME, Paper No. DETC99/DTM-8760.
- [10] Messac, A., Martinez, M. P. and Simpson, T. W., 2002 "Effective Product Family Design Using Physical Programming," *Engineering Optimization*, **34**(3), pp. 245-261.
- [11] Nayak, R. U., Chen, W. and Simpson, T. W., 2002, "A Variation-Based Method for Product Family Design," *Engineering Optimization*, **34**(1), pp. 65-81.
- [12] Simpson, T. W., Seepersad, C. C. and Mistree, F., 2001, "Balancing Commonality and Performance within the Concurrent Design of Multiple Products in a Product Family," *Concurrent Engineering: Research and Applications*, **9**(3), pp. pp. 177-190.
- [13] Seepersad, C. C., Mistree, F. and Allen, J. K., 2002, "A Quantitative Approach for Designing Multiple Product Platforms for an Evolving Portfolio of Products," *ASME Design Engineering Technical Conferences - Design Automation Conference*, Montreal, Canada, ASME, Paper No. DETC02/DAC-34096.
- [14] D'Souza, B. and Simpson, T. W., 2003, "A Genetic Algorithm Based Method for Product Family Design Optimization," *Engineering Optimization*, **35**(1), pp. 1-18.
- [15] Simpson, T. W., 1997, "Designing Ranged Sets of Top-Level Design Specifications for a Family of Aircraft: An Application of Design Capability Indices," *SAE World Aviation Congress and Exposition*, Anaheim, CA, Paper No. AIAA 97-5513.
- [16] de Weck, O. L., Suh, E. S. and Chang, D., 2003, "Product Family and Platform Portfolio Optimization," *ASME Design Engineering Technical Conferences - Design Automation Conference*, Chicago, IL, ASME, DETC2003/DAC-48721.
- [17] Fujita, K. and Yoshida, H., 2001, "Product Variety Optimization: Simultaneous Optimization of Module Combination and Module Attributes," *ASME Design Engineering Technical Conferences - Design Automation Conference*, Pittsburgh, PA, ASME, Paper No. DETC2001/DAC-21058.
- [18] Fujita, K., 2002, "Product Variety Optimization Under Modular Architecture," *Computer-Aided Design*, **34**(12), pp. 953-965.
- [19] Fellini, M. K., Papalambros, P. Y. and Perez-Duarte, A., 2002, "Platform Selection Under Performance Loss Constraints In Optimal Design of Product Families," *ASME Journal of Mechanical Design*, **127**(4), pp. 524-535.
- [20] Georgiopoulos, P., Fellini, M. K., Sasena, M. and Papalambros, P. Y., 2002, "Optimal Design Decisions In Product Portfolio Valuation," *ASME Design Engineering Technical Conferences - Design Automation Conference*, Montreal, Canada, ASME, Paper No. DETC2002/DAC-34097.
- [21] Hernandez, G., Allen, J. K. and Mistree, F., 2002, "Design Of Hierarchic Platforms For Customizable Products," *ASME Design Engineering Technical Conferences - Design Automation Conference*, Montreal, Canada, ASME, Paper No. DETC2002/DAC-34095.
- [22] Mather, H., 1995, "Product Variety -- Friend or Foe?," *38th American Production & Inventory Control Society International Conference and Exhibition*, Orlando, FL, APICS, pp. pp. 378-381.
- [23] Martin, M. V. and Ishii, K., 1997, "Design For Variety: Development Of Complexity Indices and Design Charts," *ASME Design Engineering Technical Conferences - Design For Manufacturing Conference*, Sacramento, CA, ASME, Paper No. DETC97/DFM-4359.
- [24] Farrell, R. S. and Simpson, T. W., 2001, "Improving Commonality in Custom Products Using Product Families," *ASME Design Engineering Technical Conferences - Design Automation Conference*, Pittsburgh, PA, ASME, Paper No. DETC01/DAC-21125.
- [25] Simpson, T. W., Maier, J. R. A. and Mistree, F., 2001, "Product Platform Design: Method and Application," *Research in Engineering Design*, **13**(1), pp. 2-22.
- [26] Farrell, R. and Simpson, T. W., 2003, "Product Platform Design to Improve Commonality in Custom Products," *Journal of Intelligent Manufacturing*, **14**(6), pp. 541-556.
- [27] Belegundu, A. D. and Chandrupatla, T. R., 1999, *Optimization Concepts and Applications in Engineering*, Prentice Hall, Upper Saddle River, NJ.

APPENDIX A

FEASIBILITY TEST MATRIX FOR YOKE LEG EXAMPLE

```
*0010000001000112001010000010011000011000000000100001110121011
0*0000000000200000220000200000112000210000120001120000001000
00*010000000011000100000100000001110001110010002012000112200
001*10111200011000101000020001000211001110001100001100121022
0020*000000011000001100000100000001100000020010002110000112000
10010*000111011210101220012012000011000002000120001112011011
001010*00000112000102000000000001120002220010002011000111220
0010101*00000120001010000000000001120002220010002011000111220
00101011*000012000101000000000001222002120011102011000111120
001210111*00011000101000020001000211001110001100001100121022
1001000001*20112000101200010011000011000002000100001110121011
1001000001*011220101200011011000011000002000100001110011011
010000000000*10000110000100000001000210000110001110000211000
0100000000002*0000110000100000111000110000110002220000202000
00201011100001*000102000000000001212001120012202012000111220
100001000011000*20000110011012000010000001000120000122001012
1001000001110011*0100100011012000011000001000100001110011011
0000000000000001*0001100001200000000000000000000000011000011
0100000000000200000*1000010000011100021000011000010000000000
0200000000002000002*000020000011000021000010000010000000000
00101010000001100010*000000000001110001110010002011000111220
100001000012000221000*20011211000010000002000120000112001011
1001010001110012101002*0011012000010000002000120000112001011
00000000000000001000001*00001200000000000000000000000001000011
010000000000220000210000*0000000100001000012000122000212000
0011101111000110001010000*0001000111001110001100001100111022
10000100001200022100022000*212000000000002000120000112001011
100001000011000111000110001*12000000000001000010000111001011
0001101111000110001010000100*2000111001110000100001100011011
00000100000000011100001100111*000000000000000000000000011000011
020000000000100000220000100000*20000220000200000100000000000
0200000000001000002200001000002*0000220000200000100000000000
0100000000002200002100002000000*000010000120002220000212000
000100001100011100102000020021000*12000000000200002220112022
0020101110000120001020000000010002*2001120002200012200121222
00101011100001100010100000000100021*001110001100001200111122
01000000000200000110000100000110000*100001000001000000000000
0200000000002000002200002000001120002*0000220000100000000000
00201011100001200010200000000100012200*120012200012200111221
00201011000001200010200000000001120002*20012202012000111220
0020201110000120001020000000000011120011*0012202012000112220
1000010000110002200001100110110000100000*000120000112001011
020000000000200000220000200000112000220000*20000200000000000
0100000000002000002100001000001110001100001*0001220000002000
0010101100000120001010000000000111000112001*202011000111110
00101011000001100010100000000001110001110001*01001000111120
1001010001110011101001100110110000110000010001*0000112011011
00101011100001100010100000000100111100111000120*002000112111
001020000000220000200000100000001100000010120001*20000211100
010000000000220000110000200000010000100001200012*0000212000
00101000000001100010000010000000111000111001000201*000111200
001010111000011000101000000001000111001110001100001*00111122
0001000011000011001010000100120001110000000001000011*0011011
10010100011100011000011001101200001100000100001000011*001011
010000000000110000110000100000111000110000010002110000*02000
1001000001220121001021000200210000120000010002000022201*2021
0010100000000100001100001000000110000001001000101000011*000
0010101100000110001010000000000111000111000002012000111*10
0011101111000010001010000100020001110011100001000011000110*1
00010000110000010000000001002200011100000000010000010000102*
```