`Proceedings of DETC ‘04:
ASME 2004 International Design Engineering Technical Conferences
And Computers and Information in Engineering Conference
September 28-October 2, 2004, Salt Lake City, Utah

DETC2004-57221

DRAFT

PROTOTYPE PARTITIONING BASED ON REQUIREMENT FLEXIBILITY

R. E. Moe
Department of Mechanical Engineering
The University of Texas at Austin
Austin, TX 78712
smerk1234@hotmail.com

Daniel D. Jensen
U.S. Air Force Academy
Dept. of Engineering Mechanics
USAF Academy, CO 80840-6240
Dan.jensen@usafa.af.mil

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

ABSTRACT

Prototype partitioning is an often-overlooked step in the product development process that has great potential for improving project success. This paper discusses the importance of applying a systematic prototype partitioning strategy to a product development project. Quite often, prototypes are chosen based on historical reasons, with the premise that requirements are rigid and inflexible. Alternatively, a method is proposed here for prescribing a partitioning strategy that is tailored to the specific characteristics of a project and is based upon the three components of *requirement flexibility*: cost, schedule, and performance. By considering the realistic flexibility in these requirements, strategic prototyping decisions may be made to promote the success of a development project. Three product development applications illustrate the proposed method: the Black & Decker SnakeLight ™, a senior-level design project at the United State Air Force Academy, and the product development of a new umbrella concept based on compliant components.

1.0 INTRODUCTION

A broad definition of a prototype is a simplification of a product concept meant to resolve issues in product development (Otto, et al. 2001). These issues may include customer needs, manufacturability, technical performance, market analysis, product liability, aesthetics, ergonomics, and product theme among others. Inasmuch as effective prototyping hinges upon planning, a key to improving prototyping effectiveness, and thereby product development success, is to improve the prototype planning process. This planning process, including the allocation and division of prototyping tasks, is what we refer to as *prototype partitioning*.

There is a significant investment at stake in product development. Depending on the industry, companies will spend 2-8% of sales on R&D. In spite of this, there are estimates that 40-46% of product development resources are spent on products that are cancelled or do not yield adequate returns (Cooper 1993). Product development project cancellation rates are reported to be between 30-50% (Ulrich, et al. 2000; Linberg 1999; Galal, et al. 1999; Patel 1999). Although there are many reasons for the failure of product development projects, two types of risks *can* be mitigated. First, excessive resources might be spent to determine that a project is not worth pursuing. These are wasted resources that could be applied to other projects. Second, resources might be directed away from a project that is worthwhile. This case represents a lost opportunity. Both of these issues may be mitigated using the correct application of the information from the product requirements.

In addition to these resource management issues, many products that enter the market have not been developed to their full potential. This reduced potential may be associated with many aspects of the product development process. Decisions regarding prototyping are one such aspect. Effective prototype decisions, such as how many concepts to prototype simultaneously, how the information is used to sequence the prototypes, and whether replanning of the prototype strategy is needed based on testing results, are critical aspects for creating an efficient product development process. The greater the efficiency, in this sense, the greater realization of product potential (performance) may be achieved.

1.1 HOW TO IMPROVE PROTOTYPE PLANNING

Prototyping is included in many models of the product development process (Otto, et al. 2001; Juran 1992; Hupka, et al. 1996; Badri, et al. 1997; Krishnan, et al. 1995). In general, these models identify a planning phase, an execution phase, and an evaluation phase. The planning phase includes determining project requirements, determining data needed to satisfy project requirements, and partitioning the prototypes. The execution phase includes conceiving, selecting, designing and fabricating prototypes. The evaluation phase includes the prototype assessment, in which prototypes are tested against their own objectives, and the project review, in which the data produced by the prototyping effort, in its entirety, is compared against the project requirements. Figure 1 shows a simplified representation of these steps. The crux of the different product development models is how, which, and when information is passed along, or feedback, through these steps. A product development project can be considered successful if it passes the project review. At this point, the design team will have shown, to a satisfactory level, that it has met the project requirements. After the tooling design and pre-production stage, the project will then usually be transferred into production. One alternative outcome of the prototyping effort is for the product development team to show clearly that the project cannot meet its requirements. In this instance, the project should be recommended for termination. While the project is unsuccessful, the prototyping accomplished its objective. The goal is for this to be done quickly and efficiently so that resources can be used elsewhere.

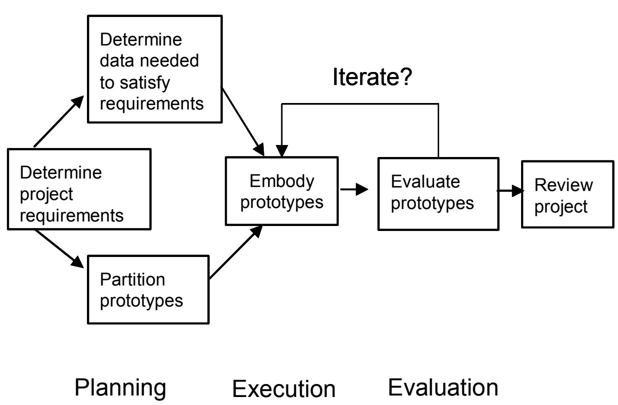

Figure1. General Prototyping Process

Most designers have been sensitized to the importance of determining both the project requirements and the data needed to satisfy them. Quality Function Deployment (QFD) is a widely accepted tool that gives guidance to prioritizing customer needs and converting project requirements into measurable engineering outputs (Hauser 1988; Otto & Wood 2001). Along with QFD, other widely accepted design philosophies such as Hazard Analysis (e.g., Failure Modes and Effects – FMEA) and Design for X (manufacturing, service, reliability, recycling, human factors, etc.) give guidance towards the data needed to satisfy a variety of requirements.

1.2 CONTEXT: PROTOTYPE PARTIONING

While methods exist to elicit and systematically address requirements, strategies for allocating prototypes (number, serial versus simultaneous, etc.) are not well formalized. Such strategies tend to be more art than science, more pragmatic than grounded with a theory. Companies tend to develop their in-house strategy and apply it repeatedly, regardless of the individual requirements of the product being developed.

Thus, prototype partitioning bears need of careful consideration, in addition to a clear definition. The word "partitioning" means to divide into parts. So, a prototype partitioning strategy is a plan to divide a prototyping effort into multiple parts. For example, if a design team chooses to execute two prototypes, the team may pursue the two ideas in parallel or in series. They may adjust their strategy to account for lessons learned as they go, or they may hold firm to their original strategy. They may choose to iterate as part of the prototyping plan or not iterate based on performance requirements. The options produce different information, at different times, with different implications on cost, performance and schedule. In good planning, these differences will be leveraged to increase the likelihood of project success and the potential to realize the most out of the product.

Figure 2 shows an example of prototype partitioning, in this case, successfully applied at a product development firm. Notice in the figure that prototyping efforts for this printer product are partitioned to correspond to the development process, from industrial design models to pre-production prototypes with baseline tooling.

This type of process, as typified by Figure 2, presents more than a few questions. What are the indicators that more than one prototype should be considered? When are two half-scale prototypes superior to a single full-scale prototype? Which is better, two simultaneous prototypes that divide the budget in half, or two iterations that divide *both* the budget and timeline in half? When should more resources be requested to complete additional prototypes? If a second effort can be justified, why not a third, and a fourth? Indeed, how many? Should *these efforts* be simultaneous or iterative?

It is not enough to just acknowledge that partitioning, in the abstract, may improve product development. A method for prescribing the most appropriate strategy based upon specific project information must be provided and validated. This strategy should be expected to increase the probability that information will be available at the time it is needed in order to make decisions about product improvement or project continuance (budget reviews, design reviews, phase reviews, etc.). If information is made available, then efficient decisions may be made for rapidly improving product concepts, and projects that cannot be executed within their constraints will be terminated efficiently. This later case will free up resources for other projects, and projects that can be executed will be continued towards their fruitful ends. In this way a systematic method for creating a prototype partitioning strategy directly addresses the risks of product development. The result will be an increase in the number of projects that reach a successful conclusion, i.e., satisfy the project requirements, for a set amount of resources. In addition, there should also be an increase in the efficiency of the way that the prototyping process aids in the fulfillment of the product requirements. Finally, the termination of products that will not meet performance targets can be done efficiently.

2.0 REVIEW OF LITERATURE

Prototype partitioning has gone mostly unrecognized in the literature. Many authors do not identify or include a partitioning strategy as one of the elements of the prototyping processes or as a factor important to the success of a project (Belassi, et al. 1996; Cooper 1993; Glegg 1981; Hupka, et al. 1996; Mullins, et al. 1998; Otto, et al. 2001; Reik 2001; Souder, et al. 1998; Ulrich, et al. 2000; Williams 1996). The reader is left with the impression that, despite all the encouragement given for planning in other aspects of the product development process, prototype partitioning need not be considered.

Partitioning, when mentioned, is sometimes presented as the passive result of other decisions. It is a consequence, not a strategy at all. For example, the number of prototypes may be given by the size of the budget divided by the expected prototyping cost (Ulrich, et al 2000, Krishnan 1995). Likewise, the number of iterations may be given by the timeline divided by the expected duration of the prototyping cycle. Three criticisms of this line of thinking are evident. First, prototype cost and duration can only be predicted after conceptualizing ideas and investigating the technology limits of their fabrication. These steps occur during the embodiment of a prototype. But by this time, the partitioning, as a planning step, should already have been performed. Second, partitioning should be set up *a priori* to improve the quality, relevance, or timeliness of information. To define partitioning based upon technology limits is to blindly accept the information that technology can provide, whether or not it answers the questions posed by the requirements. Third, it fails to leverage the innovative nature of design teams. Consider, for instance, a project with a budget of $100,000, an

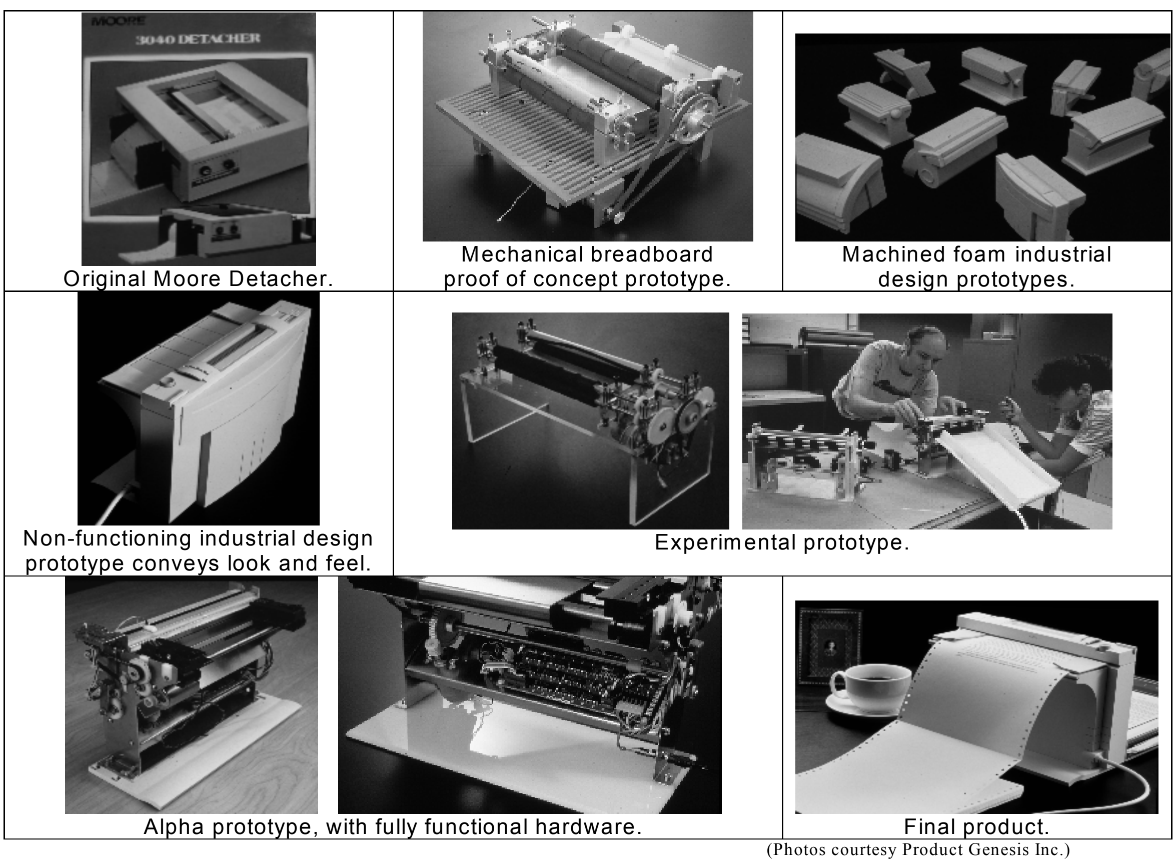

Figure 2. Example of prototype partitioning of a printer product.

expected prototype cost of $20,000, and a prescribed strategy of six prototypes. The prototyping strategy obviously exceeds its budget. However, inventive solutions are one outcome of such gaps in the state-of-the-art technology. Partitioning should drive the technical embodiment, not be a slave to it.

There are anecdotes of partitioning decisions playing an important role in protecting project risk. Badri, et al. (1997) tells of a "promising alternative noted by the CEO" to fund two parallel design efforts. Reik (2001) relates a case study in which the decisions to pursue two alternatives proved fortuitous. In both references, the successful partitioning strategy was treated as a serendipitous decision. No rules are presented for identifying when this strategy should be employed.

Some references apply rules of thumb to all projects. Glegg (1981) suggests product development projects are best served by a progression of three design iterations: the base idea, the first embodiment and the contemporary embodiment. Koen (1985) suggests a time in every project, about three-quarters of the way to completion, when investigation of alternatives should cease and efforts be focused on perfecting the preferred embodiment, or risk exhausting resources before anything is embodied. It seems unlikely that universally applied rules will be as effective in improving product development success as would strategies tailored for the specifics of a project.

Some references assert the appropriateness of particular partitioning strategies, but do not indicate how to implement them. Reik (2001) suggests that a plan is needed for developing prototypes early if they are to enhance project goals. Also multiple paths should be selected "if risk and timing requirements dictate." However, no specific prototyping plan is presented, nor methods for assessing the value of prototypes or project risks. Koen (1985) recommends, "design decisions that carry a high penalty should be identified early, taken tentatively, and made so as to be reversible". The reader is left to determine how to identify these decisions and implement this guidance. Fricke (1999) reports that successful designers maintain a well-balanced number of design variants by *intuitively* alternating between generating solutions and assessing them. Less useful are references that describe the opposed benefits and risks of a strategy without resolving the dilemma. Rodrigues, et al. (1996) suggests that increasing schedule pressure may increase the project progress rate but cause quality to fall. Lastly, some strategies depend on information that is knowable only after the project has been completed. These cannot be prescriptive. Krishnan, et al. (1995) discuss this shortcoming in a strategy for scheduling overlapping design efforts. They conclude that the method is best applied to well understood, or re-engineered, design tasks.

The literature is also lean with regard to descriptions of experiments that could evaluate the effectiveness of prototype partitioning strategies. There are studies that perform post-mortem project analysis to find causal links between planning steps and project success (Cooper 1993, Souder, et al. 1998). These studies report, but do not control, the planning steps; they do not attempt to test the results of multiple teams executing the same design task under different strategies. Savage, et al. (1998) describes an experiment that shows promise. To test the results of cost and schedule constraints on design quality, the same simple design task was assigned to a number of independent subjects. The subjects were divided into experimental groups and different project constraints were imposed upon each group. Then subjects were allowed to solve the problem by their own design means. Unfortunately for the current purpose, the authors discarded all design efforts that were not successful, so their data cannot be used to evaluate success rates. Nevertheless, some of the logistics of their experimentation can be built upon, as described below.

Without external guidance, the design team is left to formulate a partitioning strategy in an unstructured and/or subconscious manner. Partitioning might be based upon experience or gut feel. Particularly talented, insightful, or fortunate individuals may be successful. Unfortunately, this approach is not repeatable, teachable, improvable, or testable. Because a review of the literature fails to present a systematic method for prescribing a prototype partitioning strategy, with testable effectiveness, tailored to project specifics from information that is available during project planning, a new method is proposed. A process for applying the method is presented, along with examples of its use and a protocol for evaluating its effectiveness.

3.0 PROPOSED METHOD FOR PRESCRIBING A PROTOTYPE PARTITIONING STRATEGY

In order to propose a method for prescribing a prototype partitioning strategy, it is necessary to identify the input to the method and the manner for converting the input into a strategy.

3.1 Input for the Method

Project requirements are an attractive candidate for input to the proposed method for several reasons. First, the ultimate purpose of partitioning is to improve project efficiency and success, which is tied directly to requirements through the evaluation step. Second, requirements are universal to all projects. Third, specific requirements are unique for each project. Fourth, requirements are one of the scant pieces of information that can be expected to be available in the project planning phase.

Of all the aspects of requirements, it is requirement flexibility that holds the key to project termination; and so, should be considered as input for the method. Requirement flexibility is defined, here, as the relationship between the likelihood of project termination and the degree by which the requirement is missed. Alternately, requirement flexibility can also be seen as the degree to which project requirements must be strictly adhered to in order to facilitate project success. Because there are three components of project requirements: cost, schedule, and performance (Juran 1992; Hupka, et al. 1996; Reik 2001; Otto, et al. 2001), we propose the use of these same three components of requirement flexibility: cost flexibility, schedule flexibility and performance flexibility as a basis for development of a prototype partitioning strategy.

Some may contend that there is no such thing, or should be no such thing, as requirement flexibility but that, by definition, requirements are rigid. This claim may be true in an ideal world, but it is inconsistent with how projects are reviewed. First of all, numerous authors state implicitly, or explicitly, that project requirements are negotiable, dynamic, indefinite, subject to creep, or changed based upon the results of prototyping (Alexander 1999; Al-Karaghouai, et al. 2000; Bailetti 1995; Berry 1998; Cooper 1993; Galal 1999; Juran 1992; Otto, et al. 2001; Patel 1999; Robertson 2000; Rodrigues 1996; Souder 1998; Ulrich, et al. 2000). Requirements are not static and the agreements that define them as such are contestable. For example, it is not in a business' best interest to terminate a project because it fails to meet a requirement that no longer needs to be met. Also, projects are seldom terminated at the time they actually violate requirements. If it appears that requirements will be violated, the design team is expected to propose an adjusted plan that sets the project back on track. Usually, one type of requirement is traded for another or inventive paths/solutions are developed to remove conflicts in requirements. Many times projects are only terminated when management cannot accept these trade-offs or the risks involved with inventive solutions. Some projects are terminated upon the perception that they will go over budget, run past schedule, or under-perform, when, technically, at the time of termination, they have not violated requirements. Finally, there are many projects that simply violate requirements by going over budget and past schedule without being terminated (Cooper 1993; Linberg 1999; Belassi 1996; Rodrigues 1996; Berry 1998). So, in practice, requirements are not rigid measures for evaluating project continuance. Instead it appears that there is some amount of elasticity (mix of rigidity and flexibility) that is associated with different requirements and different projects. A method for deriving prototype partitioning should take advantage of this flexibility information.

3.2 REPRESENTING REQUIREMENT FLEXIBILITY

Each component of requirement flexibility can be rated on a continuum scale between "0" for absolute flexibility; no implication to the project if the requirement is missed; and "1" for absolute rigidity; a project is terminated if the requirement is missed. Figure 3 shows this continuum. Each component of requirement flexibility can be associated with a particular prototype partitioning sub-strategy, or vector. Therefore, each of the three requirement flexibility ratings (cost, schedule and performance) can be converted into a specific prototype partitioning vector. These three vectors can then be used as input to an overall prototype partitioning strategy.

0 ←——————————————→ 1
Flexible — Rigid
(No consequence) — (Termination)

Figure 3. Requirement Flexibility Continuum (Cost, Schedule or Performance)

3.3 PROTOTYPE PARTITIONING VECTORS

Three partitioning vectors are proposed based upon understanding of how a design team would likely undertake a project at the extremes of requirement flexibility. Table 1 summarizes these three vectors. As shown in Table 1, each requirement (cost, schedule, and performance) is associated with a premise and an implication for both flexibility and rigidity. To understand these premises and implications, five definitions of prototype partitioning terms are important.

- *Effort*: prototype(s) and test(s) of a single concept,
- *Iteration*: a sequence of tasks on an effort(s),
- *Plan*: a statement of the process and tasks needed to complete an effort(s),
- *Process*: a series of actions, changes, or functions bringing about a prototype result(s). A process answers the questions of "when" and "how" for a prototype plan,
- *Task*: a piece of work associated with a prototyping plan. A task answers the question of "what" for a prototype plan.

The premises in Table 1 provide insights into the meaning we associate with the concepts of flexibility and rigidity for the partitioning vectors. They also provide the philosophical underpinnings of our approach to partitioning. From a vectorial standpoint, the premises define the direction or bounds on each partitioning vector, and the implications provide a prototyping strategy per dimension of the partitioning space.

Table 1. Foundations of Requirement Flexibility.

	COST	PERFORMANCE	SCHEDULE
Flexibility premise	Multiple efforts produce a superior product	No iteration is necessary to produce the required performance	Reworking the prototype plan during development produces a superior product
Flexibility implications	Should use multiple efforts	No iteration should be performed	Should rework the plan
Rigidity premise	A single effort uses all available resources	An iteration of prototyping effort(s) produces progress toward required product performance	A single plan will use all the available time
Rigidity implications	Should use only a single effort	Should use iteration(s)	Should use only a single plan

3.3.1 Cost Flexibility - Proliferation Vector

The premise of cost flexibility is that multiple efforts (see definitions above) produce superior products if cost is completely flexible, whereas a single effort will use all available resources when the cost requirement is rigid. For all potential new products, there are a number of competing conceptual designs. If the cost requirement is absolutely flexible, it would benefit the team to assign resources to pursue any and all ideas. On the other hand, if cost is absolutely rigid, the team will want to execute one effort to efficiently use the available resources to maximize product success. Cost flexibility, therefore, is related to the "Proliferation Vector", the number of simultaneous prototyping efforts. This vector is not "trial and error," it is just "trial." Proliferation works on the premise that some designs perform better than others, or, in other words, multiple efforts produce a superior product. It can be expected that the performance of the highest performing prototype increases as the population of efforts increase. The drawback to proliferation is that resources must be assigned to each additional effort.

3.3.2 Performance Flexibility – Planned Iteration Vector

The premise of performance flexibility is that iteration (see definition above) of a prototyping effort produces progress toward required performance if performance is rigid, whereas no iteration is necessary to produce required performance if performance is completely flexible. For all projects, there is a choice to iterate or not. If the performance requirements are absolutely flexible, then any design that is proposed at a project review should be accepted. Any iteration is a waste. However, if performance is absolutely rigid the team should plan for iteration(s) until targets are met. Performance flexibility, therefore, is related to the "Planned Iteration Vector." The basic idea of iteration is that lessons will be learned from previous observations. The second attempt will be better than the first; the third better than the second. The drawback to iteration is that time and resources must be assigned to each additional task.

3.3.3 Schedule Flexibility - Replanning Vector

The premise of schedule flexibility is that multiple plans (see definitions above) produce superior products if schedule is completely flexible, whereas a single plan will use all available time when the schedule requirement is rigid. For all projects, there are a number of potential prototyping processes and associated tasks. If the schedule requirement is absolutely flexible, it would benefit the team to rework the prototyping plan (as many times as necessary) based on information gained during the product development process. On the other hand, if schedule is absolutely rigid, the team will want to execute one plan to efficiently use the available time to maximize product success. Schedule flexibility, therefore, is related to the "Replanning Vector." Replanning works on the premise that as information becomes available, predictions will improve. The drawback to replanning is that it involves backtracking and restarts, resulting in lengthening schedules.

3.4 Graphical Representation

Each prototype partitioning vector can be placed on its own axis with its associated requirement flexibility continuum, as shown in Figure 4. These axes can be combined into a representation as a unit cube. This Partitioning Strategy Cube, shown in Figure 5, contains all the possible combinations of the prototype partitioning vectors, and therefore all possible partitioning strategies.

Proliferation Vector

0 ◄————————————► 1

Cost Flexible — Cost Rigid

Multiple Efforts — Single Effort

Replanning Vector

0 ◄————————————► 1

Schedule Flexible — Schedule Rigid

Ongoing Replanning — Single Plan

Planned Iteration Vector

0 ◄————————————► 1

Performance Flexible — Performance Rigid

No Iteration — Iteration(s)

Figure 4. Prototype Partitioning Vectors

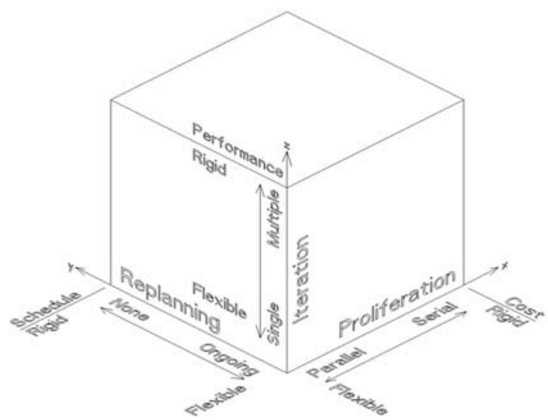

Figure 5. Partitioning Strategy Cube

3.5 The Prototype Partitioning Method

Based on our description of flexibility, a number of implications result. These implications lead to strategies for prototype partitioning. Table 1 presents the implications for each partitioning vector, and Figures 6(a) and 6(b) illustrate these implications graphically for the case of complete rigidity and complete flexibility (of cost, schedule and performance), respectively.

Although Figure 6 illustrates the extreme cases, most product development projects will likely have a combination of rigidity and flexibility in the three partitioning vectors, or will have partial rigidity or flexibility. Consider for example the case of the Black & Decker SnakeLight™ product, shown in Figure 7. Originally introduced in 1994, the requirements of this product, during its development, included cost flexibility, schedule flexibility, and performance rigidity. Cost and schedule were flexible because of the potential success of this new technology. A light of this type did not exist and was expected to "re-invent the existing [handheld personal lighting market]" (Giesecke 2004). Flexibility was provided to enable project success, especially due to the potential payoff of the SnakeLight™. The partitioning vector for performance, however, was rated to be rigid. This rating originated from the technological advancements needed to accomplish the bendable function of the light's body.

As shown in Figure 6(b), the cost and schedule flexibility lead to a prototyping plan that includes multiple efforts and reworking of the plan. "Multiple efforts" implies the simultaneous design and prototyping of more than one concept. Replanning (reworking of the plan) implies the adoption of new choices of prototyping processes and tasks as new information becomes available. Likewise, from Figure 6(a), the performance rigidity leads to a prototyping plan that includes iteration. Iteration(s) is needed to progressively improve the product concept(s) toward the goal of meeting the performance requirements.

According to the literature, Black & Decker followed this prototype partitioning strategy. It appears this strategy was followed intuitively. In particular, the SnakeLight™ development included a number of bendable concepts, replanning resulted in "several failed starts" until a appropriate management team was assembled, and a number of iterations were pursued before the bendable body ("a ball and socket type device") was successful in meeting performance requirements. This product has been extremely successful commercially.

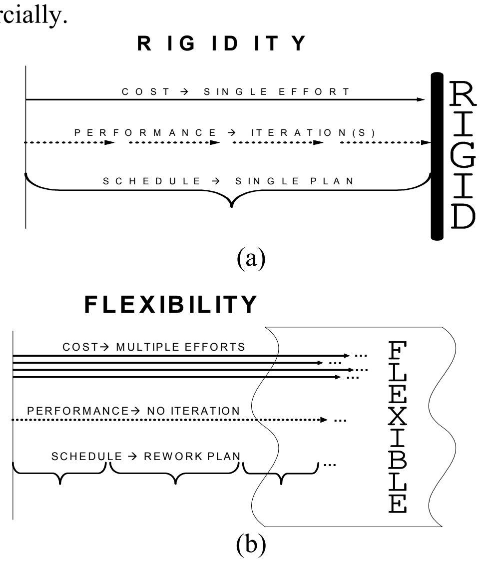

Figure 6. Strategies for Complete Rigidity and Flexibility.

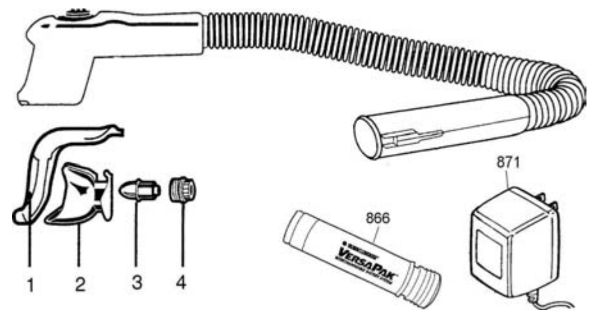

Figure 7. Black & Decker SnakeLight™ Product

Building from this example, and combining the implications of Figure 6 and Table 1 with the Partitioning Strategy Cube, Figure 5, a two-step process for prescribing prototype-partitioning strategies can be developed as shown in Figure 8. In the first step, project information is converted into a requirement flexibility rating. In the second step, Table 2 is consulted to convert the rating to a partitioning strategy, based on the combined effects of all three partitioning vectors.

Table 2 presents a subset of the alternatives represented in the Partitioning Strategy Cube. The requirement flexibility continua are reduced to their extrema, resulting in eight requirement flexibility combinations. These correspond to the corner points (vertices) of the cube. The individual cost, schedule, and performance flexibility ratings are converted to the proliferation, replanning, and planned iteration vectors, respectively. The three vectors are combined into coherent strategies. Notice the schedule vector in these strategies implies either investigatory effort(s) or intuitive effort(s). Investigatory means that replanning is possible and encouraged based on information generated from the prototyping effort(s). Intuitive means the opposite. Replanning is not possible given the schedule rigidity. Thus, the plan is not researched to the same level of depth as would be expected for investigatory studies.

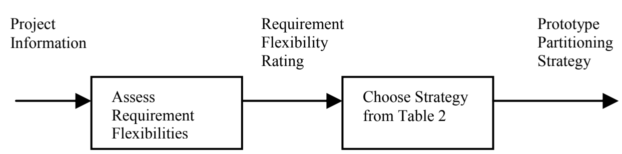

Figure 8. Process for Applying Method

Table 2. Partitioning Strategy Look-Up Table

Given these conditions:				Do the following:
	Cost Flexibility	*Schedule Flexibility*	*Performance Flexibility*	*Prototype Partitioning Strategy*
Cost and schedule flexible	0	0	0	No iteration of multiple investigated efforts
	0	0	1	Iteration(s) of multiple investigated efforts
Cost rigid	1	0	0	No iteration of a single investigated effort
	1	0	1	Iteration(s) of a single investigated effort
Schedule Rigid	0	1	0	No iteration of multiple intuitive efforts
	0	1	1	Iteration(s) of multiple intuitive efforts
Cost and Schedule rigid	1	1	0	No iteration of a single intuitive effort
	1	1	1	Iteration(s) of a single intuitive effort

4.0 APPLYING THE METHOD

Based on these partitioning strategies (Table 2), two applications of prototype partitioning for product development are developed in the following sections. These applications illustrate the method and provide initial validation of its underlying philosophical basis.

4.1 USAFA PROJECT APPLICATION

As part of the senior capstone design course at the U.S. Air Force Academy (USAFA), students design and build a baja car (Figure 9) to compete in an intercollegiate competition sponsored by the Society of Automotive Engineers (SAE).

Figure 9. USAFA Baja Car

Because USAFA is a government institution, corporate sponsors are not allowed for this project and therefore the development costs are paid directly with department funds. The cost of the project is therefore rigid. This project culminates in an intercollegiate competition which is held at a fixed time during the year. Therefore, the schedule is also rigid. Performance, in this case, is more difficult to rate. Although there are a number of target specs that will not be strictly met in the final design, there are also numerous specs that must be rigidly adhered to due either to the SAE rules or to the fact that, unless they are met, the vehicle will not achieve a minimum level of functionality. The performance rating is approximated to have value of 0.6. Therefore, the closest vertex in the cube of Figure 5, is the (1,1,1) vertex. Note that this is the "completely rigid" vertex and has its prototyping partition illustrated by Figure 6(a). The prototyping partition strategy from Table 2 is "Iterations of a single intuitive effort."

Because the cost is rigid, only one prototyping effort is completed. The rigidity in schedule necessitates that we will not rework the prototyping plan during the project. The rigidity in performance causes a need to iterate on the single prototyping effort in order to make progress toward the performance goals. In this case the effort includes various forms of prototypes of the same concept as shown in Figures 10(a) and 10(b). Iteration of the concepts included some detailed, full scale prototyping of things like the steering upright shown in Figure 11. This iteration did not produce a change in the *concept* of the design, but did reveal deficiencies in performance characteristics that resulted in changes in some details of the concept. This prototyping partition strategy has been proven to provide a path for successful completion of the Baja project. Although failures have occurred (resulting in the cancellation of that year's project), these failures have been a result of students failing to adhere to the schedule and/or performance rigidity.

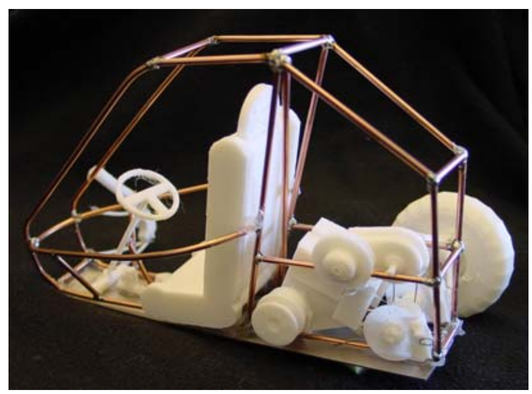

Figure 10 (a). Baja Car Initial Concept Prototype

Figure 10 (b). Baja Car Full Scale Prototype

Figure 11. Detailed Prototype of Steering Upright

4.2 Product Development of a New Umbrella Concept

Figure 12. Process for Applying Method

A design team is commissioned to develop a new umbrella product (Greer, et al. 2002; Greer, et al. 2004). As part of this commission, a primary intent is to simplify the working mechanism of previous product generations. Such a simplification is intended to improve the reliability of umbrellas, enhance manufacturability, and provide for a number of industrial design themes. Figure 12 shows a typical umbrella available on the market.

Construction of the umbrella, as illustrated in Figure 12, requires the manufacture and assembly of over 120 separate components, the majority of which are used in constructing the eight deformable beams that support the canopy. Figure 13 pictorially illustrates this complexity.

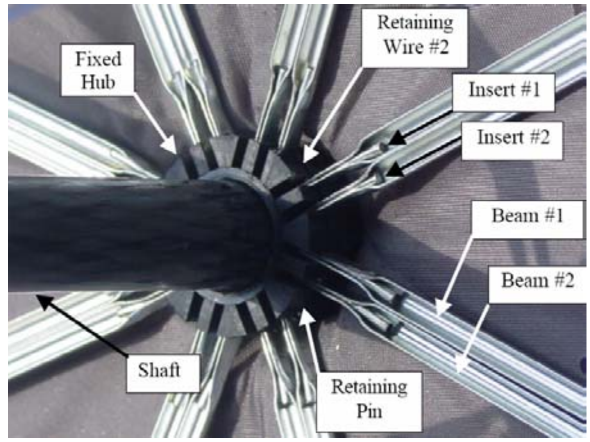

Figure 13. Umbrella Fixed Hub Structure

In order to enhance the success of the umbrella development project, a prototype partitioning strategy is tailored to the specifics of this project using the method described in Section 3.5. After generating a number of working concepts, three new designs are pursued for feasibility. Figure 14 shows early sketches of two compliant designs, and Figure 15 shows solid model illustrations.

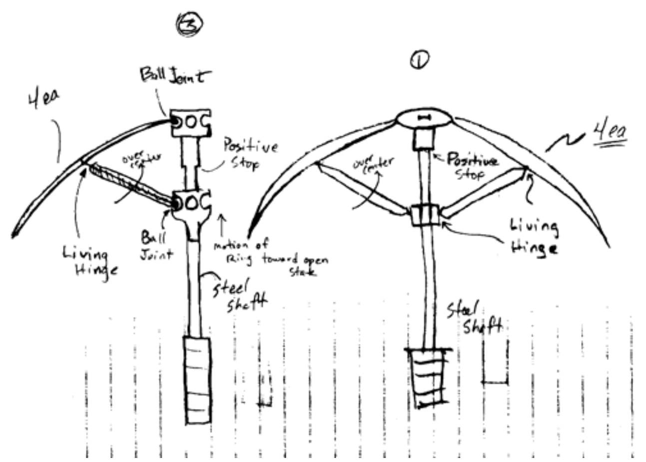

Figure 14. Sketches of compliant umbrella concepts

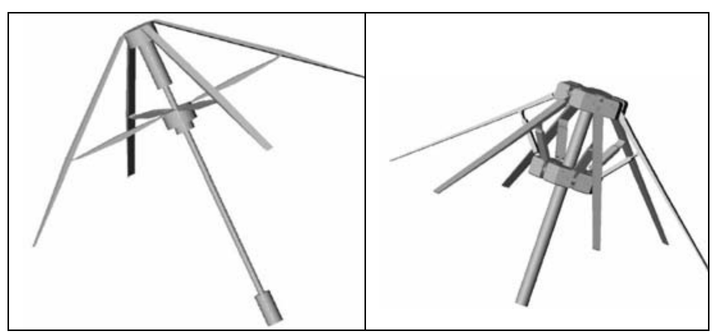

Figure 15. Solid models of compliant umbrella concepts: living hinge, and compliant-arm, quick-connect designs.

With these concepts in hand, the requirements for the project dictated the following conclusions:

- Cost is flexible based on the scale of the product and availability of Solid Freeform Fabrication (rapid prototyping) capabilities to produce prototypes.
- Schedule is flexible because the product represents new technology and because the client assumes that there will be false starts.
- Performance is inflexible and demanding due to the competing market with successful and well-established umbrella manufacturers.

Based on these conclusions, the requirement flexibility for the project (cost, schedule, and performance vectors) is (0,0,1), and the corresponding partitioning strategy is "iteration(s) of multiple investigated efforts." Interpreted for the umbrella development, this strategy means that cost flexibility provides for multiple concepts to be pursued and prototyped simultaneously. The three primary concepts from the development team will thus be pursued. Schedule flexibility focuses on investigated concepts that will be pursued based on known and synthesized compliant device technology that will be analyzed and tested to provide for replanning. Alternatively, the performance inflexibility implies that prototype iterations be planned (within the allotted schedule) to compensate for manufacturing problems that are common with compliant mechanisms.

The resulting partitioning strategy manifests as the plan(s) shown in Figure 16. Notice that iteration is shown. The first set of efforts (A, B &C) focus on the three initial concepts with prototype testing and evaluation. The iteration (A', B' & C') seeks adaptations to the three concepts. Next, based on a replan, three new concepts (D, E & F) are produced that compensate for performance issues indicated by the initial tests.

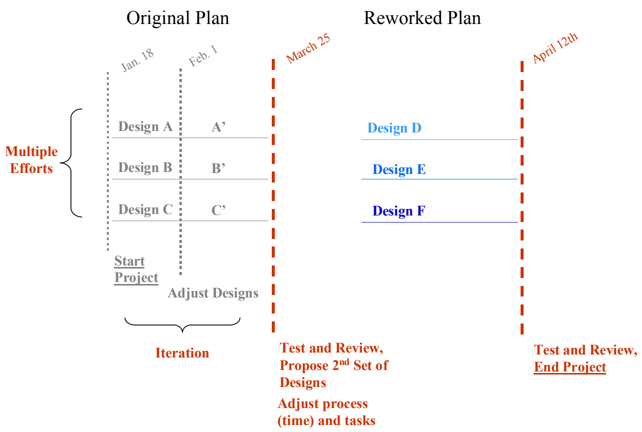

Figure 16. Project schedule based on partitioning strategy.

Figure 17 shows a scaled prototype produced from the reworked plan. This prototype hit performance targets for the project and compensated for compliant joint delamination and fatigue failure from the original iteration.

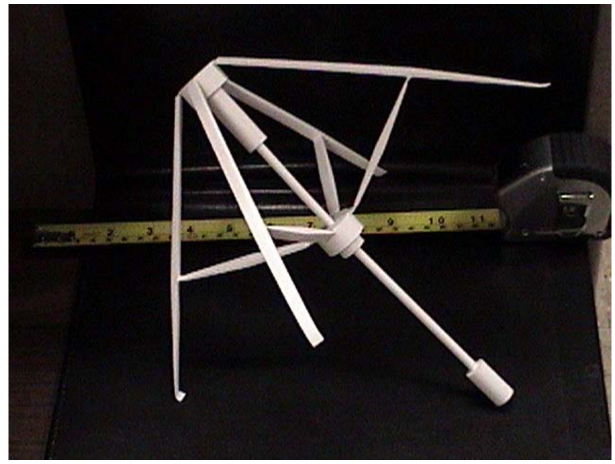

Figure 17. Scaled SFF compliant umbrella prototype

5.0 EVALUATING THE METHOD

The effectiveness of our method for prescribing prototype-partitioning strategies has been tested using individual projects with industry. It has also been tested in university design courses and in special workshops with industrial project managers. However, further evaluation and validation is needed to explore the full prototype partitioning space. This section describes the nature of the experimentation that could be used to further evaluate the effectiveness of the method. The hypothesis to be tested is that for a given design task and requirement flexibility rating, executing the prescribed partitioning strategy (Table 2), is more effective in producing successful projects than executing no strategy, or only an intuitive strategy.

5.1 Proposed Experimental Protocol

An experimenter presents a single specific design task with a given requirement flexibility rating, budget, completion date, and performance target (all at the experimenter's discretion) to a statistical population of independent design teams. The teams are divided into a minimum of two groups. The teams in the first group are controls. They perform the design task without any knowledge or predisposition towards partitioning strategies. Therefore they may use either no strategy or possibly an intuitive strategy will be adopted by default. The teams in the second group are directed to perform the design task while executing the partitioning strategy prescribed for the given requirement flexibility rating.

The design task should have one, or a very few, clearly definable specific measurable outcomes that are continuous and straight-forward to measure. Examples include building a vehicle that can reach a specific speed, or a structure that can support a specific load. Each team is given the same operating budget and deadline for collecting performance data. The teams are allowed to overspend their budgets and collect data late. Each team is informed that three pieces of data will be collected: the absolute prototype performance (speed, height, etc.), total expenditure, and completion date. In addition, each team is informed that a "project score" and a "project outcome" will be determined.

The project score is the sum of three scores, one each for results against cost, schedule and performance targets. One hundred points are to be awarded for each component if the requirement target is met, for a maximum of 300 points. Points are subtracted per a sliding scale for over-running the budget, completing late, or missing the performance target. For instance, five points may be subtracted from the cost score for each 1% of budget overrun; from the schedule score for each day that test data is late; and from the performance score for each 1% of performance underachievement.

The project outcome is a binary "success"/"failure" designation. It is determined from the project score by a simple algorithm shown in Table 3.

Table 3. Project Outcome Designation

Given Requirement Flexibility Rating	Definition of project Success
(1,1,1)	300 pts
(0,0,1)	100 performance points
(1,0,0)	100 budget points
(0,1,0)	100 schedule points
(1,1,0)	100 budget and 100 schedule points
(1,0,1)	100 budget and 100 performance points
(0,1,1)	100 schedule and100 performance points
(0,0,0)	All projects "successful"

Each team is responsible for a final report that contains performance results, total expenditures, and testing completion date. For the control group, this is the only documentation.

The experimental teams are assigned certain other documentation to confirm that they adhere to their assigned partitioning strategies. At the start of each iteration, and prior to executing any prototypes, teams are responsible for a "Proposed Design Report" that includes proposed bills of materials, feature sets, and manufacturing process diagrams. At the end of each iteration, they are also responsible for reports that include test results and design lessons (what worked, what did not, what should be done next, etc.). Teams assigned strategies with replanning are allowed to change their proposed designs, fabrication processes, and iteration timelines based upon information they collect. At the midpoint of each iteration, these teams are responsible for a "Replanning Report," which contains descriptions, and a justification of, the proposed changes. Teams assigned strategies with no replanning are discouraged from changes to the initial proposed designs or timelines. But, it is recognized that there may be changes, so they will be documented. Very little latitude should be granted for adjustments to designs and timelines. Teams assigned strategies with multiple concurrent prototypes are responsible for documenting each effort.

5.2 Analyzing Data

The most relevant data for comparison is the percentage of teams within a group that achieve a project outcome designated "successful". Significance between groups can be determined by evaluating the hypothesis that each group has the same success probability rate.

Project score and absolute performance, are only indirectly related to project success, but are continuous in nature, and therefore, have greater resolution for the same sample size. A Student's t-test can be performed to determine if there is a significant difference in project score and absolute performance across strategies. In either case, the experimenter must be prepared to add additional teams if increased resolution of data is needed.

6.0 DISCUSSION

This paper presents an approach to improving product development effectiveness. A method for prescribing a prototype partitioning strategy is proposed in which cost flexibility is converted into a proliferation vector, schedule flexibility is converted into a replanning vector, performance flexibility is converted into a planned iterations vector, and the vectors are combined into a single coherent strategy. A basic step-by-step process is presented for applying the method. Applications of the process are demonstrated, and a protocol is described for evaluating the effectiveness of the method.

There have been several assumptions made in the creation of the method described in this paper that do not have prior validation. This paper is, by no means, the first to suggest that multiple prototypes have value; that there is value to parallel designs, replanning, and iterations; or that product development is improved with good planning. Nevertheless, it is asserted in this paper that prototype partitioning is a crucial and underutilized aspect of product development; that requirement flexibility is an appropriate basis for prescribing a partitioning strategy; that proliferation, replanning, and planned iterations are the most appropriate prototyping strategies to be combined with the components of requirement flexibility; and that all this will increase the number of successful projects.

No clear direction is given for how requirement flexibility can be discerned or rated. The presentation of this method assumes that the design team will be able to determine and rate requirement flexibilities by its own means.

In most cases, the described method can only provide guidance for the partitioning strategy and then only at the extreme values of these continua is there association with a clear strategy. Questions such as scale factor and extent of functionality are not addressed. One thrust of future work might address these more detailed aspects of the prototype partition. Also, because the assignment of values (0 or 1) to the partitioning vectors is not an exact science, it may become evident during the product development process that the original values assigned to the vectors was incorrect. Alternately, the flexibility of certain vectors may simply change during the development process if, for example, additional funding becomes available. A strategy could be formulated that would deal with the "mid-course corrections" to the partitioning vectors. Another aspect of future work is the execution of the experiments (see Section 5) needed to prove the effectiveness and validate the assumptions of this method.

7.0 SUMMARY

Product development can be improved by applying a systematic approach to prototype partitioning. Projects that cannot be executed within resource targets would be terminated more efficiently, freeing up resources for other efforts, and projects that can be successful will show their capabilities and be continued. Quality of products going to production will be enhanced. A method for creating a prototyping partitioning strategy is proposed that combines a proliferation vector derived from cost flexibility, a replanning vector derived from schedule flexibility, and a planned iteration vector derived from performance flexibility. Improvement to the product development process would be measurable in the number of projects that satisfy project requirements for a set amount of resources. A basic step-by-step process for applying the method is presented. Finally, successful applications of partitioning strategies are reported.

REFERENCES

Alexander, I., 1999, "Is There Such a Thing as a User Requirement," *Requirements Engineering*, 4:221-223.

Al-Karaghouai, W., AlShawi, S., Fitzgerald, G., 2000, "Negotiating and Understanding Information Systems Requirements: the Use of Set Diagrams," *Requirements Engineering*, 5:93-102.

Badri, M. A., Mortagy, A., Davis, D., Davis, D., 1997, "Effective Analysis and Planning of R&D Stages: a Simulation Approach," *International Journal of Project Management*, 15:351-358.

Bailetti, A. J., Litva, P. F., 1995, "Integrating Customer Requirements into Product Designs," *J Prod Innov Manag*, 12:3-15.

Belassi, W., Tukel, O. I., 1996, "A New Framework for Determining Critical Success/Failure Factors in Projects," *Int. J. of Proj. Manag*, 14:141-151.

Berry, D. M., 1998, "Software and House Requirements Engineering: Lessons Learned in Combating Requirements Creep," *Requirements Engineering*, 3:242-244.

Cooper, R. G., 1993, *Winning at New Products: Accelerating the Process from Idea to Launch*, Cambridge, MA: Perseus.

Fricke, G., 1999, "Successful Approaches in Dealing with Differently Precise Design Problems," *Design Studies*, 20:417-429.

Galal, G. H., Paul, R. J., 1999, "A Qualitative Scenario Approach to Managing Evolving Requirements," *Requirements Engineering*, 4:92-102.

Giesecke, F., SnakeLight™, Case Study., http://cwx.prenhall.com/bookbind/pubbooks/giesecke/chapter0/deluxe.html, January, 2004.

Glegg, G. L., 1981, *The Development of Design*, London: Cambridge University Press.

Greer, J., *Effort Flow Analysis: A Methodology for Directed Product Evolution Using Rigid Body and Compliant Mechanisms*, PhD Dissertation, The University of Texas at Austin, Summer, 2002.

Greer, J., Jensen, D. and Wood, K., "Effort Flow Analysis: A Methodology for Directed Product Evolution," Journal of *Design Studies*, 2004.

Hauser, J. R., Clausing, D., 1988, "The House of Quality," *Harvard Business Review*, 66:63-73.

Hupka, V., Eder, W. E., 1996, *Design Science: Introduction to the Needs, Scope and Organization of Engineering Design Knowledge*, London: Springer.

Juran, J. M., 1992*, Juran on Quality by Design: the New Steps for Planning Quality into Goods and Services*, NY: Free Press.

Koen, B. V., 1985, *Definition of the Engineering Method*, Washington, D.C.: Am. Soc. Eng. Ed.

Krishnan, V., Eppinger, S.D., Whitney, D. E., 1995, "Accelerating Product Development by the Exchange of Preliminary Product Design Information," *Journal of Mechanical Design*, 117:491-498.

Linberg, K. R., 1999, "Software Developer Perceptions about Software Project Failure: a Case Study," *The Journal of Systems and Software*, 49:177-192.

Mullins, J. W., and Sutherland, D. J., 1998, "New Product Development in Rapidly Changing Markets: An Exploratory Study," *Journal of Product Innovation Management*, 15:224-236.

Otto, K., and Wood, K., 2001, *Product Design: Techniques in Reverse Engineering and New Product Development*, Upper Saddle River, NJ: Prentice Hall.

Patel, N. V., 1999, "The Spiral of Change Model for Coping with Changing and Ongoing Requirements," *Requirements Engineering*, 4:77-84.

Riek, R. F., 2001, "From Experience: Capturing Hard-Won NPD Lessons in Checklists," *Journal of Product Innovation Management*, 18:301-313.

Robertson, J., Robertson, S., 2000, "Requirements Management: a Cinderella Story," *Requirements Engineering*, 5:134-136.

Rodrigues, A., Bowers, J., 1996, "The Role of System Dynamics in Project Management," *I. Journal of Project Management*, 14:213-220.

Savage J.C.D., Miles, C., Moore, C. J., Miles, J. C., 1998, "The interaction of time and cost constraints on the design process", *Design Studies*, 19:217-233.

Souder, W. E., Sherman, J. D., Davies-Cooper, R., 1998, "Environmental Uncertainty, Organizational Integration, and New Product Development Effectiveness: a Test of Contingency Theory," *Journal of Product Innovation Management*, 15:520-533.

Ulrich, K. T., Eppinger S. D., 2000, *Product Design and Development*, NY: Mc Graw-Hill.

Williams, P., 1996, "Project Failures and How to Avoid Them," *Computer Audit Update*, 29-31.