JOURNAL OF AIRCRAFT
Vol. 48, No. 3, May–June 2011

Value-Driven Design

Paul D. Collopy*
Value-Driven Design Institute, Urbana, Illinois 61803
and
Peter M. Hollingsworth†
University of Manchester, Manchester, England M13 9PL, United Kingdom

DOI: 10.2514/1.C000311

Value-driven design is a movement that is using economic theory to transform systems engineering to better use optimization to improve the design of large systems, particularly in aerospace and defense. This paper describes the ideas and methods that are current in value-driven design. An overview of the history of value-driven design is provided, leading up to a survey of the approaches and techniques that are currently being studied and implemented in university research and experimental government programs. The paper closes with a discussion of the potential benefits that may be realized by employing the value-driven design framework.

I. Introduction

OUR ability to develop large complex artifacts such as aircraft, space launch systems, submarines, and even automobiles is in disarray. Symptoms are ubiquitous development cost overruns and schedule delays. As of 2009, NASA's entire agenda has been hijacked by overruns on the International Space Station and the Constellation program [1]. U.S. automobile manufacturers are nearly out of business, due largely to their inability to develop competitive car designs on time. The new Chevy Volt, carrying the hopes of the reorganized General Motors, has apparently doubled in manufacturing cost during design [2]. Boeing is two years late on developing its next commercial airliner, the 787, and has just incurred a $2.5 billion overrun [3]. But that is hardly a surprise, since the last major new airliner, the Airbus A380, was two years late and overspent by over €2 billion.‡

The U.S. Department of Defense executes a large-enough set of complex development programs to yield some meaningful statistics. The set of 96 major weapon system development programs currently underway have overrun by a total of $296 billion, with an average development cost growth of 42%, and an average delay of 22 months [4]. Extrapolating to completion shows that the total loss to delay, overruns, and reductions in material (generally caused by overruns) is $55 billion per year, or $150 million each day.

This is not a new phenomenon, but it appears to be worsening. Current defense system developments, projected to completion, are overrunning 78% in cost and 63% in schedule. Augustine's [5] study of similar programs in the 1970s and 1980s showed overruns of 50% in cost and 33% in schedule.§ The NASA Ares I launch system is overrunning its $28 billion original estimate by $12 billion (43%).¶ Historically, large NASA programs have overrun by 50% [6].

Examination of these programs shows that our systems engineering processes simply do not work. Requirements are set, but the artifacts do not meet the requirements. Risk-management systems are put in place and carefully executed, but fail to prevent major delays and shortfalls on almost all programs. A theory has been developed that these overruns and delays are not an unfortunate hazard, but are instead a natural outcome of complex product development, performed in accordance with systems engineering standards [7].

Decades of cutting-edge methods, processes, and tools have been injected into the systems engineering process to no avail. This includes a great deal of research and tool development in multidisciplinary optimization. Despite major innovations enabled by desktop computing and networks, bottom-line performance in the large class of engineering development programs is no better, and perhaps worse, than in the late 1970s and early 1980s.

We propose value-driven design (VDD) as a solution to these problems. VDD is not a new method, process, or tool for design. Rather, in the spirit of Hazelrigg [8], it is a framework against which methods, processes, and tools can be assessed.

Value-driven design changes the way designers deal with extensive attributes. *Extensive attributes* are attributes of the system or product being designed, or attributes of its components, where the system attribute is a function of component attributes. That is, extensive system attributes are functions of extensive component attributes. Examples of extensive attributes are weight, all performance attributes, reliability, maintainability, safety, and similar supportability attributes, plus all aspects of cost. In a programmatic sense, schedule and technical risk are also extensive attributes. Although these do not encompass all the important attributes of a system, they do include the elements most associated with program delays and cost overruns.

In the value-driven design framework, there are no requirements applied to extensive attributes, either at the system level or at the component level. That is, there is no directive to the design engineers stating that the aircraft must weigh less than 31,300 kg, nor is there a threshold weight or objective weight or anything of the sort. Instead,

Presented as Paper 2009-7099 at the 9th AIAA Aviation Technology, Integration, and Operations Conference, Hilton Head, SC, 21–23 September 2009; received 11 February 2010; revision received 14 August 2010; accepted for publication 7 December 2010.

*Executive Director, P.O. Box 247; paul.collopy@vddi.org. Associate Fellow AIAA.

†Lecturer, School of Mechanical, Aerospace and Civil Engineering, Sackville Street; peter.hollingsworth@manchester.ac.uk. Member AIAA.

‡Data available online at http://en.wikipedia.org/wiki/Airbus_A380 [retrieved 2 September 2009].

§Cost overruns and schedule delays on current projects reported by the Government Accountability Office (GAO) [4] are comparable with Augustine's [5] reported overruns on completed projects. However, current overruns understate the magnitude of overrun at completion, because most of the projects surveyed by the GAO are in the middle of development, and cost overruns will tend to increase more as they complete. Thus, we use projected completion statistics, which show today's programs much worse than Augustine's 1970s to 1980s programs. On the other hand, Augustine's data eliminate programs that were cancelled due to excessive cost growth, so the performance deterioration from the 1980s to current programs is somewhat overstated when we say that 50% cost growth has increased to 78% cost growth, and likewise for schedule.

¶Data available online at http://en.wikipedia.org/wiki/Ares_I [retrieved 2 September 2009].

every engineering team (system, component, subcomponent, etc.) has an objective function, which is a scalar function that converts the team's full set of attributes into a score. The design team's task is to create a design that yields the highest score (while meeting all the requirements on the nonextensive attributes). For a more thorough discussion of value, utility, and objective functions, see Collopy [9].

Under value-driven design, the function of systems engineering is to flow objective functions down to each component, to monitor the status of component attributes and collective status of system attributes, and to take appropriate actions to maintain balances in the system. Collopy [7] shows that VDD, by flowing down objective functions, enhances the probability of achieving top-level requirements in comparison with traditional systems that flow down and allocate requirements. This result may be counterintuitive. The gist of the argument is recapitulated in Sec. IV.C.

This paper is a survey of the current status of value-driven design and some of the relevant research now underway. We will begin with a review of the trends that led up to present state of value-driven design.

II. History

Rather than a chain of brilliant insights by individuals, the development of what is now called value-driven design has been more like a river of inexorable progress. Many ideas have occurred simultaneously to a large number of people, and in most cases, several of the inspired have pressed for research, written seminal papers, or simply pushed the ideas into practice in engineering design. Because of the breadth of this movement, our coverage is necessarily spotty. We offer our apologies to the many contributors whose work deserves acknowledgment, but is neglected here.

It is almost a tautology that the best choice you can make is the one that leads to the best outcome. By the 20th century, economists such as DeBreu [10] were stating in formal terms that given a set of alternative actions, a rational person will choose the action that is expected to yield the outcome that is most preferred. However, the interesting case, and the only case generally relevant to design, is when the outcomes of alternatives are uncertain. In a book that was really about something else altogether, von Neumann and Morgenstern [11] launch the normative theory of decision-making by showing that the rational choice under uncertainty is the action for which the probability distribution of outcomes has the highest probabilistic expectation of utility. Collopy [9] shows that for usual design situations, utility and monetary value are equivalent, so that the preferred design choice is the one with the greatest value in monetary units.

Since 1950, von Neumann and Morgenstern's [11] notion of rationality has penetrated many fields of scholarship and human endeavor, but, until now, has not changed the way we design large systems. This is not to say that leading thinkers have not tried to introduce best value as a fundamental design rule. Simon [12] recommends that designers find the optimum value, but noted that this may be too difficult, so that most designs would settle for the highest value design that they could easily locate.

In the 1970s Sage [13] examined large-scale systems engineering and systems analysis as decision processes, using the decision analytic perspective of Keeney and Raiffa [14]. Sage [13] recommends value system design as an early step in the systems design process. Sage's description of value system design, "the transformation of the properties of a thing into a format amenable to instrumental or extrinsic valuation," accurately characterizes system value modeling today. Sage found that value system design is necessary to enable rational design decisions. He developed a process for worth assessment that acknowledged the hierarchical structure of values, and in this case described the lowest level inputs to the assessment as *attributes*, the word we use today, rather than *properties*.

Schmit [15] promoted the idea of structural design optimization in the 1960s. Sobieszczanski-Sobieski et al. [16] and others developed techniques of integrating engineering modeling codes into decomposed optimization structures in the 1980s, launching the field of multidisciplinary optimization (MDO). MDO uses optimization techniques to find the best designs, although some methods drive the search by constraints more than by an objective function, and most examples of MDO in the extensive literature take little care in defining what "best" might mean. Often, the objective function is simply to minimize weight or perhaps minimize drag [17]. The central problem in MDO is to coordinate, during conceptual and preliminary design, design work in different disciplines. For example, an aircraft wing design may simultaneously present structural and aerodynamic design challenges, and these problems are intimately interrelated.

A notable advance in MDO has been the bilevel integrated system synthesis (BLISS). The premise behind BLISS is to create "a method for the optimization of engineering systems by decomposition" [18]. BLISS does this by linking a series of decomposed black-box analysis tools in an integrated computational environment. The key component to allowing for a manageable optimization problem is the decomposition of the overall objective function to a series of subfunctions and associated constraints. Sobieszczanski-Sobieski et al. [18,19] and Kodiyalam and Sobieszczanski-Sobieski [20] provide a rigorous framework for creating the subobjective functions and constraints and linking a series of either computational analysis programs or surrogate models in a wholly computational environment.

Whereas MDO addresses the problem of coordinating the work of engineering disciplines during preliminary design, VDD is chiefly concerned with defining a process that will continue to provide value into detailed design, when components are designed in relative isolation (often at separate firms, linked primarily by contracts). BLISS decomposes constraints into separate disciplines, where VDD eliminates constraints wherever possible and decomposes the system objective function to flow down to components. Where BLISS can iterate on cross-linkages between disciplines during preliminary design, VDD must confront fairly static interfaces between components in the detailed design phase.

Sobieszczanski-Sobieski [21] and others have suggested that it is possible to use a multiobjective approach at the top or S-optimizer level, especially in conjunction with using BLISS for optimizing a system-of-systems. This contrasts with the original BLISS paper in which Sobieszczanski-Sobieski et al. [18] argued that BLISS essentially converts a multiobjective problem, where each sublevel optimization would be its own objective function and hence extensive attribute, to a single objective optimization problem where each of the contributing attributes would be weighted into the S-optimizer's objective function. VDD pays a great deal of attention to constructing the system objective function (often called a system value model) and ordinarily does not formulate separate objective and combine them by weighting.

In the 1990s Hazelrigg [22] developed the a decision-theoretic systems engineering process that meets our definition of value-driven design. Hazelrigg emphasized engineering design as a process of developing actionable information about a prospective product and employing the information to make rational design decisions. He emphasizes the formulation of a system of values to direct design decisions.

In the late 1990s, Collopy developed a value-based communication and control process [23] and a distributed optimization process [24] built around flowing component design objectives down from a system value model. Each component objective function is a linear function of component attributes, such as weight and cost. The output of a component objective function varies one for one with system value. The (nonlinear) system value model is a scalar function of system attributes, constructed using the laws of economics [9]. The output of the system value model, system value, need only be a measure of goodness, such that one design is better than another design if and only if its system value is greater. Inevitably, this value is measured in monetary units such as dollars. Collopy [25] has demonstrated that system value models can be very simple, even for large complex systems.

At the same time, Barry Boehm and others formed a community of interest in economics-driven software engineering research (EDSER). This evolved into a research program in value-based

software engineering (VBSE) [26]. VBSE incorporates all the elements of the current value-driven design framework, including a focus on rational decision-making during design, but is particularly focused on software. An important area that VBSE began to explore is the deep linkage between theories of decision-making under uncertainty and systems engineering risk-management processes.

Also during the 1990s and into the 2000s, the Aerospace Systems Design Laboratory at the Georgia Institute of Technology produced a large amount of new research analyzing complex systems and systems-of-systems against a range of metrics, including top-level extensive attributes and value. Significant advancements in the use of surrogate models to analyze and select technologies, both at the subsystem level and at the system level were performed by Kirby [27] and Baker [28]. This work was extended by Hollingsworth [29] to include the ability to concurrently consider the effect of attributes on system availability. Biltgen [30], Ender [31], and others also contributed to the foundations of what is now value-driven design. Development is continuing with the research of Pfaender [32], Fernandez Martin [33], and Briceno [34], focusing on the dynamic aspects of development and the external market.

During the early 2000s, a great deal of research on economically directed systems engineering was produced by the Engineering Systems Division of the Massachusetts Institute of Technology under the direction of Daniel Hastings [35,36]. Within this group, de Weck et al. [37], developed new methods for exploring a design tradespace using generalized information network analysis and many efforts evaluated designs using real options analysis [38].

Of particular interest is the work of Rhodes and Ross, who have developed value-driven design processes for the conceptual and preliminary phases of design [39–41]. These studies have particularly investigated flexibility in the face of requirements changes over time.

Both Ross et al. [39–41] and de Weck et al. [37] employ a graphic form of optimization, tradespace exploration, in which designs are plotted against two axes that measure components of value. The position of a design within the plot indicates its value. Tradespace exploration can visually group the best designs and suggest the reasons why they are best.

Joseph Saleh, now at the Georgia Institute of Technology, has continued the investigations into value-based design, which he began under Hastings. Saleh and his students have developed several spacecraft value models, exploring alternative ways of thinking about satellite design and how designers create value. In particular, they have investigated the value of reliability [42], maintenance [43], and responsiveness [44] and argued against cost metrics that do not account for value [45].

Brown and Eremenko [46,47] developed another incarnation of value-driven design at the U.S. Defense Advanced Research Projects Agency. Their process is called value-centric design [46–48]. The original concept owes some debt to conversations between Brown, Eremenko, and Saleh while the F6 fractionated spacecraft program was still in the planning stage. During the preliminary design of F6, four performer teams led by Boeing, Lockheed Martin, Northrop Grumman, and Orbital Sciences, respectively, each developed a value-centric systems engineering methodology and constructed a system value model for a satellite cluster [49–51]. The emphasis in F6 value-centric design has been on quantifying the value of flexibility and robustness, which are key benefits of fractionated architectures.

In 2005, Paul Collopy, James Sturges, and Fred Striz founded a committee to promote the application of value-driven design (a term coined by Sturges). At the time, all three chaired technical committees (TCs) within AIAA. Collopy chaired the Economics TC, Sturges chaired the Systems Engineering TC, and Stritz chaired the Multidisciplinary Optimization TC. The AIAA VDD Program Committee, officially launched in 2006, was envisioned as a way to address an important concern to the mutual benefit of all three technical committees.

The VDD committee has conducted four workshops to demonstrate the application of value-driven design thinking. A documented example is the 2006 workshop that applied VDD to design of a notional Global Positioning System [52]. The workshop illustrated the difference between the value models used in VDD and other value- and optimization-based approaches to systems design. VDD flows down objective functions to components to formulate self-contained component design problems that can be assigned to a team or subcontracted to a separate organization. This is different from the multidisciplinary optimization paradigm, where parallel design tasks continually exchange information and modify each other's objectives and constraints. To create component objective functions, the system objective function (or value model) must be differentiated in its value output with respect to each of the important system attributes (a gradient). This cannot be done if the objective function does not have a single scalar output. For instance, multi-attribute tradespace exploration [53] (MATE) combines all attributes into two summary measures, value and cost. But a usable gradient cannot be taken on the vector [value, cost], so VDD cannot use MATE as a value model. Likewise, a method that only addresses weight, drag, and durability cannot work as a VDD value model, because it does not incorporate vital attributes such as reliability and cost. Many of the more sophisticated value models employ simulation techniques such as Monte Carlo. These cannot be used as VDD value models, because they are not differentiable. More detail is provided in Collopy [9].

A. Attribute- and Capability-Based Design

The ability to identify, measure, and model quantifiable system attributes is a challenge for many complex systems. Current efforts tend to focus on quickly visualizing and evaluating solutions in a rapid manner. To identify good solutions, multicriteria decision-making techniques are used to create one or two measures of goodness [54].

Although these techniques are not directly value-based, they help create the framework necessary to create and use a value model for design of future systems. They provide a means of evaluating and analyzing a significant number of design alternatives in a timely manner. Toward this end, they use an integrated modeling and simulation environment. There are a number of different means of achieving this integration; however, the most promising is the development of a surrogate- or metamodel-based analysis capability [55,56].

The use of surrogate models is useful in that it creates two characteristics that are beneficial for value-driven design. One, the model is more or less continuously differentiable. This enables gradient-based optimizers to operate on the value model and all of its submodels. Two, surrogate models significantly decrease the computational effort needed to evaluate changes in attributes of any of the subsystems. The surrogate model process achieves this by replacing potentially complex disciplinary analysis tools with locally appropriate statistical models of the resulting behavior. These models estimate the response of a system as either a linear, exponential, or transcendental function. The key to the reduction in complexity is the fact that sufficiently local behavior tends to behave smoothly and predictably. The use of surrogate representations is well understood and accepted in conceptual and preliminary design, and the principles are sufficiently straightforward that expansion to other uses, including detailed design, may be feasible.

One of the outcomes of attribute-based approaches is the ability to user cluster analysis to look at the availability of solution technologies depending upon the desired settings of attributes. Cluster analysis has proven useful for first-order selection of materials based upon a small number of extensive attributes [57]. This idea is extensible to other processes for selecting system solutions [29]. An example of this for general solution systems is shown in Fig. 1.

The result is not a value model but the ability to visualize the quantitative relationship between solution options and the extensive attributes that can be used to define a value model. The concept is readily extended to higher numbers of attributes. It is relatively easy to understand the behavior in three dimensions, as shown in Fig. 2;

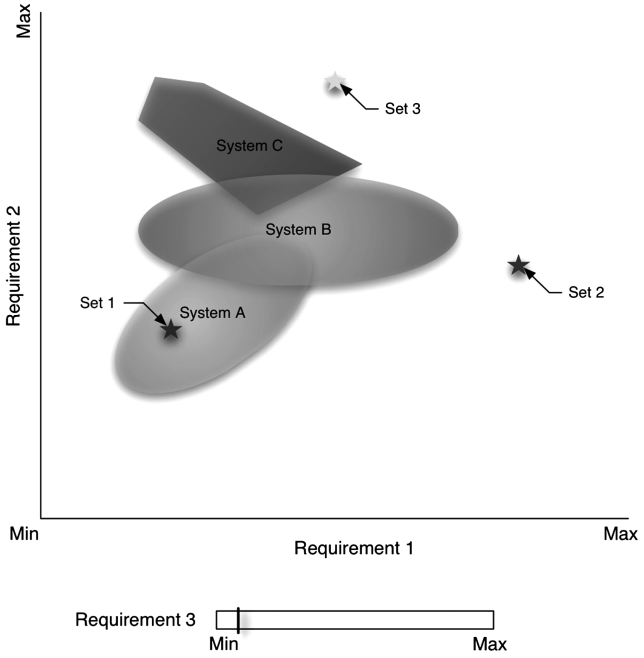

Fig. 1 Example cluster representation of three potential solution systems. Note that attributes are referred to as requirements.

however, as the dimensionality increases it becomes increasingly cumbersome to assimilate the information in a timely manner.

One other benefit of the cluster analysis approach is that it is possible to quickly visualize the relationship between the solution system's extensible attributes and the attributes of the lower-level systems. For example, one could investigate the behavior of a top-level capability to the propulsion system design variables. An example of this for a integrated system in an operational environment is shown in Fig. 3 [55]. In a crude sense, this enables the identification of what subsystem properties would give access to the desired range of system attributes. Biltgen et al. [55] refer to this as inverse design; however, because of the fact that there is no guarantee that the mapping between a specific setting of system level attributes and the underlying subsystem attributes is unique it is not simply a matter of determining what combination of attribute settings are desired to determine the design of a particular subsystem.

B. Dynamics, Options, and Competitive Analysis

Another recent development is the creation of methods to investigate the value of design, operational, or maintenance flexibility. This

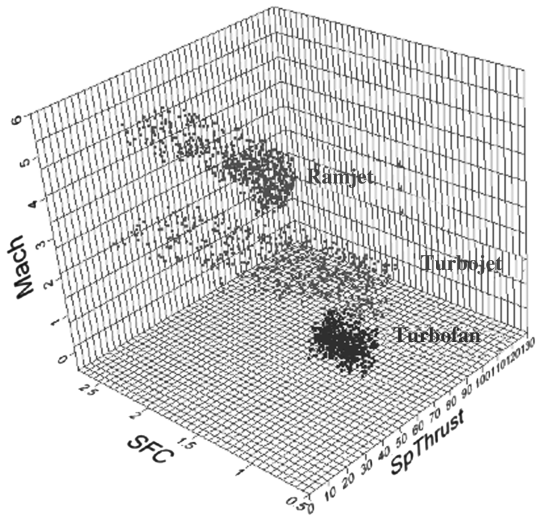

Fig. 2 Example three-dimensional cluster representation for three different possible propulsion system options.

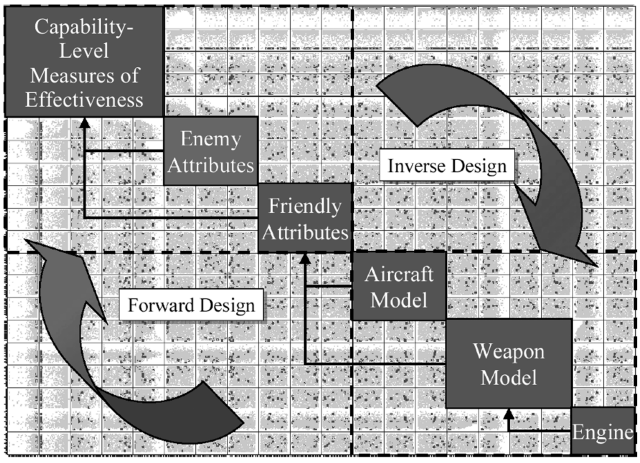

Fig. 3 Example linkage between subsystem design state and top-level capability.

has led to the use of options analysis to examine phenomena such as the effect of competition on design choices.

The development of options analysis methods, often called real options, requires that options be valued in a manner similar to financial options. The easiest way to do this is to use net present value (NPV) in the calculations of the options [29,39,58]. The beauty of using NPV as a measure of goodness is that it allows for a comparison of different systems across a range of attributes in commensurate units. In other words, it is one form of a value model. By using a single value model it is possible to easily make decisions that occur over multiple time periods and provide the capability to modify the solution to external and internal changes in the product's environment. The maintenance phase of a system's life presents challenges to the value modeler, because the user is concerned about different issues during different time periods. For example, during early operation, training and ease of use might be paramount. Late in life, reliability and maintenance cost are stronger concerns. Marais and Saleh [43] use NPV and Markov chain modeling to provide a technique that explicitly models a maintenance program as a series of options. Thus, choices made at different points in time can be weighed together. The decisions to take maintenance actions can be represented in a decision tree, and a decision policy or strategy can be developed that maximizes the expected net present value of the decision tree. VDD supports decision-tree analysis, because the value model can assign a value to the prospect at the terminus of each branch of the tree. (More information on decision-tree analysis is provided in Abbas and Howard [59].) This process is widely expandable to other aspects of the design process including technology selection and development [33]. Another approach to this issue is epoch-era analysis [60], which explicitly partitions the life cycle into separate epochs and addresses different value propositions during each.

Using the NPV-based approach, Briceno and Mavris [58,61] (and later Briceno [34]) have developed a means of comparing technology selection and design choices in the face of uncertain knowledge about the choices and decisions of competitors. Leveraging the principles behind game theory, Briceno [34] considers the end product (in this case, a propulsion system for a commercial airliner) in the face of a diverse nonuniform set of customers (airlines) with competition from dissimilar competing firms. This means that one firm may be better positioned to capture the business of a subset of customers, whereas another firm may be better able to capture the business of another subset of customers. This approach, summarized in Fig. 4, moves away from the concept of creating an average customer/user out of a disparate group and the use of market share or customer value as the measure of goodness of a product. Although market share and customer value are important surrogate metrics, the net present value of future profits more directly measures the firm's value [34].

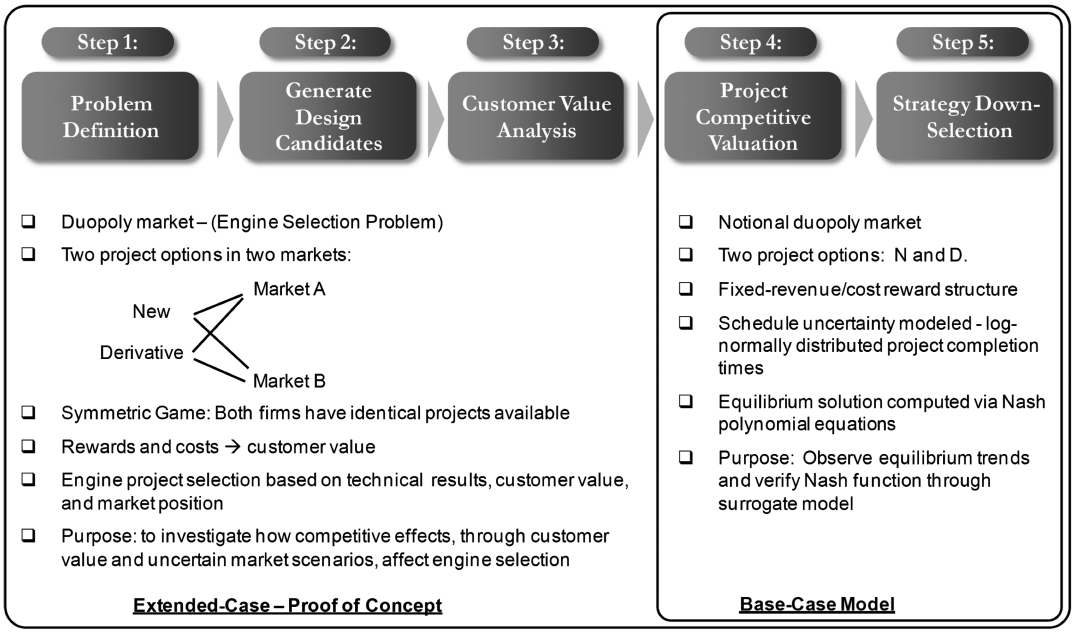

Fig. 4 Briceno's [34] proposed competitive analysis methodology.

An additional set of techniques that have been applied to the design of complex systems is the use of system or business dynamics [62,63] to describe the adoption and penetration of new technologies and designs into the market. Hollingsworth et al. [64] shows and Pfaender [32] further expands on the possibility that small and incremental changes to the product design, or that of its competitors, can lead to significant changes in the adoption and success of a product, especially when commonality effects can drive customer perception. As a consequence, value can change rapidly because of these differences, and metrics that do not use a universal representation of goodness might produce products that are not only unsuccessful, but have missed the opportunity for success by only having seemingly insignificant differences from those products that are successful.

Although not all of these developments fit strictly within the definition of VDD, they each relate in some way to the incorporation of value as a focal point in design. VDD is an evolving process, and we anticipate that syncretic practitioners will draw from all these areas to extend and refine the VDD process. The next section will describe where the process stands today.

III. Current State of Value-Driven Design

The defining characteristic of value-driven design is that engineers, when making design choices, select the best design rather than selecting any design that meets requirements, or the design that is most likely to meet requirements [8].

Figure 5 illustrates this contrast in a cyclical view of the design process. The cycle implies that a design results from multiple passes at the design problem. Starting, arbitrarily, at design variables on the right side, the design team picks a point in the design space at which to attempt a design.** The design variables that parameterize the design constitute a rough outline of the design. In the Elaborate arc, designers elaborate this rough outline into a detailed representation (configuration, also called a product definition or part definition) of the object to be designed. In the Analyze arc, engineers estimate the attributes of the object, often using physics-based predictive modeling tools such as finite element stress–strain models or computational fluid dynamics. Analysis produces a second description of the design instance, a vector of attributes of the design. Simon [12] describes the lower two arcs as a mapping of the design from internal substance (which is described in engineering terms) to external function (which is the perspective relevant to the user or customer). Thus, the design variables are defined to make sense to the design engineers, but the attributes are defined to connect to the customer.

The upper left arc is the process that differentiates value-driven design from traditional systems engineering. In the latter method, the Evaluate arc is a determination of whether the attributes meet requirements. If they do, the cycle is complete. Otherwise, another round is attempted, or the team capitulates. Under VDD, the attributes are assessed with an objective function or value model, which gives a scalar score to any set of attributes. If the current configuration has a better score than any previous attempt, it is the preferred configuration to date. At this point, the design team can accept the configuration as their product or try to produce an even better design by going around the cycle again.

The upper right arc is the domain of optimization, or, less formally, improvement. Optimization algorithms use vectors of design variables (points in the design space) and the resulting score (value) from the evaluation arc in order to guess where in the design space to next look for the best design. VDD is not an optimization process in this sense; instead, VDD is a framework that enables the use of optimization. However, VDD does not compel designers to use a formal optimization methodology. They are free to guess or use their best judgement to select the new design variables on each iteration through the design cycle.

The cycle in Fig. 5 applies to detailed design of a system component, or to conceptual design of the overall system. That is, the design cycle is scalable, from components to subsystems to systems or even systems-of-systems.

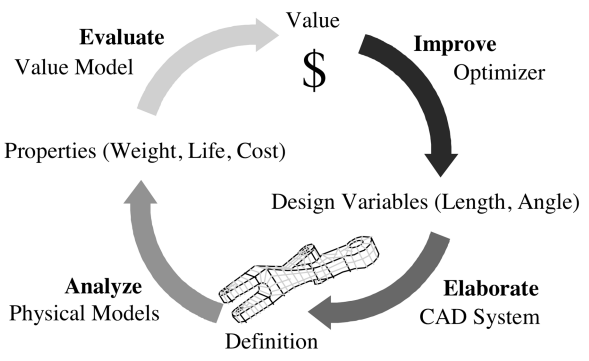

Fig. 5 Design process.

**The starting point in the design space could be new, but is more often an existing design.

A challenge in implementing a process within the VDD framework is generating the objective function (the function that inputs attributes and outputs value). The benefits cited in Sec. IV below can only be achieved if every component has a consistent objective function. To do this, the system must have an overall system objective function, which Keeney [65] calls a *value model*, or, in the engineering domain, a *system value model*. The current state of system value modeling is discussed in Collopy [9] and is beyond the scope of this paper. The system value model inputs system attributes and outputs a system score.

A. Conceptual Design

The purpose of the conceptual design phase is no different in VDD than it is in the current existing design processes. If anything, the VDD approach makes the goals of the conceptual design phase stand out with greater clarity. The primary purpose of conceptual design is to identify and select the best baseline concept for the needs of the customer market, with respect to the producer's bottom line. It is not, strictly, to create the product that provides the most customer value. The extreme example of this is a winner-takes-all government competition where the choices considered are, effectively, the highest customer value design or no design. An opposite example would be the consumer electronics market, where some of the most profitable products are not those that capture the most market share, but those with the highest profit margin, such as Apple laptop computers. The modern aerospace industry lies somewhere in between. The majority of large-scale government projects are single-customer winner-takes-all arrangements. On the commercial side, the market for aircraft is an oligopoly, whereas the market for engines and avionics, which is actually a dual market, has features of both the oligopoly and of the monopsony. Finally, the market for air transportation services is highly competitive.

The consequence of this is that for many applications it has never been possible to meet Lave and March's [66] three evaluation criteria of *truth*, *beauty*, and *justice*, using just extensive performance attributes. Collopy [9] reviews a range of methods that are commonly used to make choices in conceptual design and finds them to be lacking. Decision support is even more difficult to achieve when dynamics and competition are included. Instead, by using a value model, a change in market conditions or an increase in the diversity of customers or suppliers requires only an update to the model rather than a reconstruction of the entire extensive attribute space.

Unlike the late preliminary and detailed design phases, the cost of making decisions and including options in the conceptual design phase is relatively minor, both computationally and monetarily. As such, it is reasonable to envision that a number of different potential solution systems and technologies will be compared. Furthermore, we expect many of these will be modeled using different tools and use different intensive attributes. For example, a jet propelled aircraft and a solid rocket propelled missile will have different propulsions system design parameters. However, in operation a series of common extensive attributes and ultimately a value model can be developed. This is possible because regions of the value model can be mapped from multiple regions of the extensive attribute space, which in turn, can be mapped from different intensive attribute spaces. Figures 1 and 2 above illustrate the mapping of multiple, disparate systems to a single set of extensible attributes. Hollingsworth and Mavris [67] demonstrate that substantially different systems with similar design variables could be mapped to the same set of extensive attributes, and Hollingsworth [29] later demonstrates that it was possible to map relatively similar systems with diverse design variables to the same extensive space. This is no different than what has been done experimentally with materials and presented by Ashby [57]. An example is shown in Fig. 6.

Note that neither Hollingsworth [29] nor Hollingsworth and Mavris [67] considered the goodness of any of the systems investigated. Value-driven design would add a value model to any of these of multisystem mapping to enable a search for the best design. The methods proposed by Hollingsworth [29], Biltgen [30], and Ender [31], each resulting from a slightly different approach to

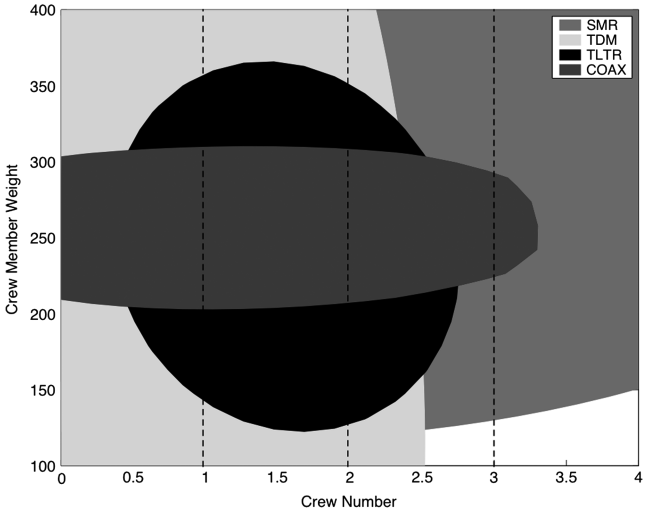

Fig. 6 Example of systems with different intensive attributes mapped to a common extensive space.

tackling the multisystem problem, can be employed within the value-driven design conceptual design process. Each of the methods discussed here makes use of surrogate models to simplify complex interactions. The worst-case scenario is that the conceptual design engineer must keep track of a handful of basic candidate solutions that are compared using the same value model, but each one has a different set of design variables.

The same process for comparing different systems can also be used for comparing different technologies. For those technologies that share the same intensive attributes and can be modeled using the same computational tools it is often possible to perform a simple linear analysis. Even in the cases where the behavior of the modeling and simulation tools limits the amount of direct gradients that can be achieved, it is often possible to create surrogate models that span the extensive attribute and technology space. One means of achieving this is the unified tradeoff environment (UTE) [28,68]. The UTE specifically uses quadratic linear surrogate models to map similar intensive attributes of similar technologies through a higher-level system to a single set of extensive attributes. A notional representation of the UTE is shown in Fig. 7. Again, note that the authors did not apply value models, but by mapping to the same extensible space it is reasonable to see where a value model could be applied.

If the technologies' intensive attribute spaces are sufficiently different, it is still possible to use the method shown by Hollingsworth [29] to map technologies, effectively bifurcating each candidate system into separate systems based upon each class of technologies.

Finally, once the best candidate solution system(s) and associated high-level technologies have been identified, it is possible to account for uncertainty by combining the mapping of intensive to extensive attributes and system value model, and thereby begin to manage risk. Because of the low expense of modeling multiple solution systems, it is fairly easy to develop an option chain where decisions can be mapped out in time based upon knowledge gained from technology development, market changes, and competitors' behavior. This is, in effect, leveraging the work performed by Marais and Saleh [43], Saleh et al. [69], Fernandez Martin [33], Briceno [34], and Briceno and Mavris [61]. There is still a lot of research needed to fully develop this aspect of VDD; however, the work above already uses what is effectively a value model approach and it is reasonable to assume that further developments along these lines will prove to be both reasonable and fruitful.

The down selection of solution systems and technology sets that results from conceptual design inherently reduces the combinatorial complexity of the design problem. This allows an increase in the amount of detail undertaken in modeling each particular system and its component subsystems, and a corresponding increase in the model and data complexity. Thus, the value model and evaluation

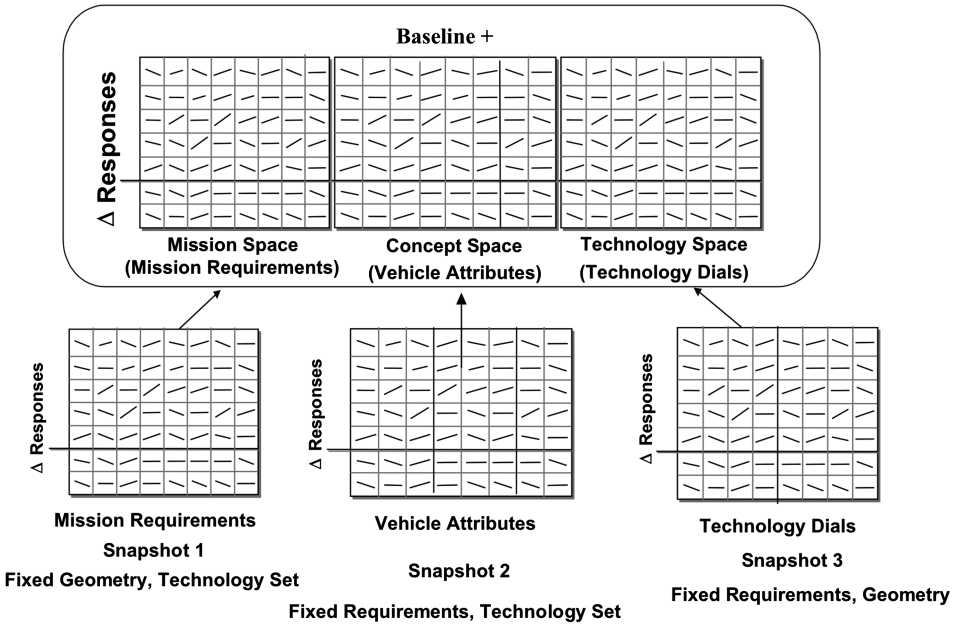

Fig. 7 Notional unified tradeoff environment.

structure can be used directly in the later phases of design. New analysis has only to be mapped into either the extensive attribute space of the solutions system or the intensive attribute space of the relevant subsystem.

B. Preliminary Design

Once the candidate design concept is fixed, the preliminary design phase begins. The design as a whole is elaborated as far as is practical. Next, the design is partitioned into components. For a large system, these components are subdivided into more components, and subdivided again several times, until the total number of components may number in the hundreds or more. At each level, partitioning results in definition of the scope of each component and definition of interface requirements between components.

Traditionally, extensive requirements, such as weight and cost, are allocated to components. In value-driven design, objective functions are allocated instead. A distributed optimization method has been established [20] for uniquely deriving the component objective functions from the system value model. In brief, the process involves the composition function, which is a function that inputs component extensive attributes and outputs system extensive attributes. The composition function exists by the definition of extensive attributes. The composition function is concatenated with the system value model. The resulting function inputs component attributes and outputs a scalar system value. This concatenated function is approximated with a first-order Taylor series. Constants in the series are dispensed with, and the linear terms are separated by component. These collections of terms become the component objective functions that enable value-driven design.

C. Detailed Design

The linearized component objective functions can be displayed on component scorecards (Fig. 8) to guide component design teams during the detailed design phase. With the scorecards it is possible to for each component design team member to see the impact of a change of any of their design variables on system value.

Scorecards can feed the design status to project management in real time, permitting instantaneous roll-ups of weight, cost, reliability, and other system attributes. If management observes that the system is moving toward a hard limit, component objective functions can be modified to adjust the design. For example, if a satellite is growing in weight and threatening to exceed launch system limits, the value of weight can be increased in all the scoreboards. This will have some clearly deleterious effects, particularly due to inconsistencies between design choices made before the change with choices made later, but it can move the program away from a point of failure.

Management can also plot trends of the design value of different components. This should be a useful indicator of which components are getting into trouble.

D. Risk Management

Value-driven design will have a major positive impact on the management of risk during system development. Under VDD, unfortunate eventualities are not simply bad, they are quantifiable. For example, if the new lightweight shaft material does not pass its coupon test, we will make the shaft out of a heavier nickel alloy, which will increase the shaft weight by 42 lb, increasing system weight by 63 lb, and reducing system value by $22,000 per production unit. If there is a 30% chance of coupon test failure, the expectation of loss in $6,600 per unit. This calculation is illustrated in Fig. 9. We would deduct the $6600 from the system's value until the test resolves the outcome one way or the other [70].

	Status		Gradient		Value
Efficiency	90%		150,000		135,000
Weight	700		-130		-91,000
Reliability	1500		2.3		3,450
Maintainability	7.8		-340		2,652
Maintenance Cost	500	X	-0.5	=	-250
Support Equipment	12		-15		-180
Radar Cross-Section	0.1		-1200		-120
InfraRed Signature	1.4		-50		-70
Manufacturing Cost	700		-1		-700
Design Value					**43,478**

Fig. 8 Component design scorecards. There is one row per component attribute. The left column, Status, records the current state of component attributes. The middle column shows the coefficients of the component objective function (which are also the partial derivatives of system value with respect to each attribute. The right column is the product of the first two columns (and is not especially meaningful). The sum of the right column is the current component design value. The component design team's job is to make this sum increase.

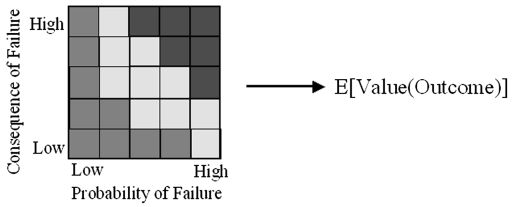

Fig. 9 Risk-assessment template is replaced with quantified expectation. Instead of using qualitative five-level templates, the probability of failure is simply multiplied by the consequence of failure to give the expectation of the consequence. Under VDD, calculation of the consequence of failure is much more tractable than under traditional systems engineering, because the value model dollarizes so much of the event space.

Current risk-management processes simply hope that the coupon will pass the test and the problem will go away. The test is recorded as a step down on the waterfall chart on the left side of Fig. 10. Properly, every test point should lead to a step down and a step up, because the risk level coming into the test (which we interpret as expectation of loss) is the probabilistic average of the possible outcomes of the test. The average of the outcomes can never be greater than all the outcomes.

These thoughts are just the beginning of a rigorous probabilistic reformulation of the systems engineering risk-management process. We believe there is a great deal of useful progress that can be made in this area.

E. Value-Based Acquisition

If government programs are to realize the benefits of value-driven design, they must integrate VDD into their acquisition processes. Carter and White [71] provide one approach, which they call value-based acquisition, or VBA:

> The key to VBA at the program level is the development of a value model that embodies key system design features, such as weight, manufacturing cost, reliability, and the like, as well as key acquisition concerns, such as cost and schedule. Once a quantitative value model has been defined, it can become the basis for contracting. A program officer can offer a contract in which price is a function of value. The contract would specify the price that the government would be willing to pay for different levels of performance..Under a value-based contract, a contractor maximizes profit by including only those features whose value to the government exceeds their cost.

When a firm accepts a contract under which their profit is directly tied to the value of their current design, per the value model, they will naturally adopt the value model to guide the design, since this is the route to maximizing profits.

The firm will also want their subcontractors to adopt value-driven design, to enhance profitability, and to offload risk onto the subcontractors. The prime contractor will be driven to build incentives into its subcontracts that directly parallel the incentives in the government's prime contract.

However, these incentives may be intractable unless the value model is kept simple. We believe that a realistic application of value-based acquisition would limit the system value model to less than 20 attributes and less than 100 equations. Furthermore, every subcontractor will work toward a flowed-down objective function, which, by distributed optimization theory [24], is always a single linear equation.

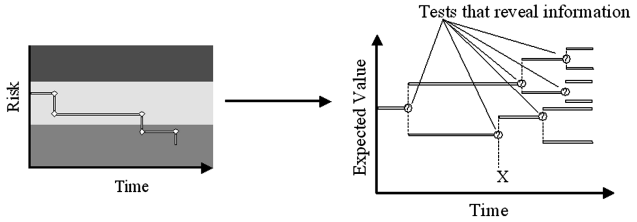

Fig. 10 Waterfall charts run downhill, but true uncertainties must have outcomes that are better than, and worse than, the current expectation. A VDD risk plan would identify test points that will reveal whether the situation is improving or deteriorating. Planning should determine what decisions will be taken depending on the test results.

IV. Benefits of Value-Driven Design

VDD provides three major benefits to the engineering design of complex systems:

1) VDD enables and encourages design optimization for the whole system during early design phases and for each component during detailed design.

2) VDD prevents design trade conflicts, and thereby prevents dead-loss trade combinations.

3) By eliminating requirements for extensive attributes at the component level, VDD avoids the cost growth and performance erosion caused by requirements.

Each of these three benefits will be explained in turn.

A. VDD Enables Optimization

VDD addresses the engineering of complex systems with a simple scalable process that enables design optimization. At its essence, optimization is a design rule that says, "Find the best design." In contrast, today's systems engineering process says, "Find a design that meets the requirements." The latter directive gives up a certain amount of value: namely, the difference between a selected design that meets the requirements and the best design. As an example, consider Fig. 11. Figure 11a illustrates the current process in a case with only two requirements: cost and weight. Any design inside the shaded box meets the requirement, so all are equally good. However, most are not feasible (such as an airplane that costs 0 and weighs 0). Figure 11b shows the same design problem from the perspective of optimization. Here, the region of feasible design is shaded. A scoring function (called an *objective function* in optimization theory) is used to search for the best design, where a higher score indicates that a design is better. The vector is the gradient of the objective function, and the small square shows where the best design is found. Because the small square is the best feasible design, it is better than any design that meets the requirements. Moreover, without using optimization, a design team has little chance and no motivation to find the best design within the requirements box.

For visualization, Fig. 11 has simplified the attribute space to only two dimensions, but typical system designs (and component designs) may have 10 to 20 important attributes. In such high dimensional spaces, the impact of optimization is much greater. For example, in a 10 dimensional space, requirements, on average, remove 99.9% of the relevant design space. In a 20-dimensional space, the corresponding figure in 99.9999%. It is not unreasonable to say that allocated requirements have a one-in-a-million chance of finding the best design. Optimization is not perfect, but it can easily beat requirements allocation by several orders of magnitude, when we look at it this way.

According to George Hazelrigg, "Values tell engineers what you want. Requirements only tell them what you do not want."

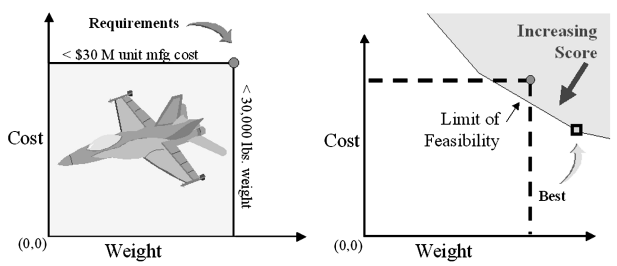

Fig. 11 Optimization chooses the best design.

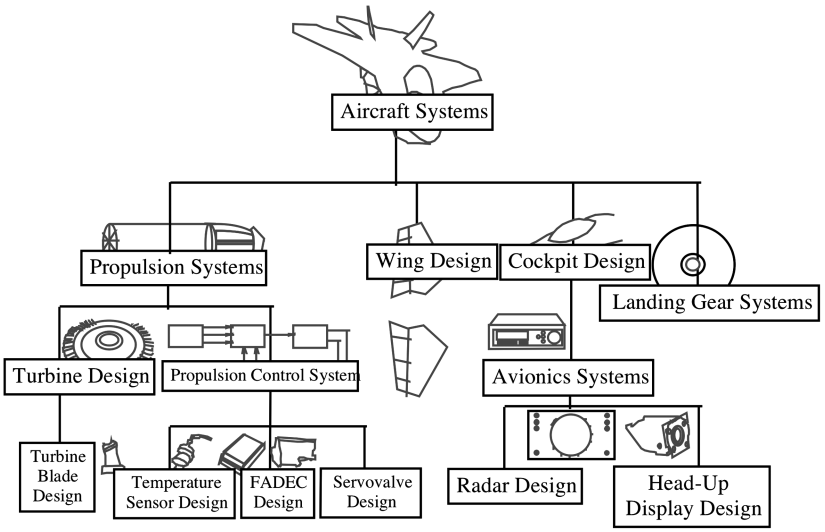

Fig. 12 Hierarchical organization of an aerospace system design (FADEC denotes the full authority digital engine control).

B. VDD Prevents Design Trade Conflicts

When design engineers make rational decisions to meet component requirements, the collective results are often clearly irrational. Although one component design team sacrifices great cost to achieve a small weight reduction, another team on the same program gives up far more weight to realize a small reduction in cost. Teams work at cross purposes so that they can each meet their allocated requirements for weight, cost, reliability, performance, and so on. The result is what economists refer to as a dead loss, a net increase in weight and cost and decrease in performance and reliability, which, on the whole, clearly degrades the total system [23]. Requirement-induced dead losses tend to reduce the value of large systems by tens of percent.

VDD prevents dead-loss trades by providing each component with an objective function that implicitly contains all the trade factors among all extensive attributes, and ensures that all these trade factors are consistent across all components. Under VDD, it is impossible for two separate trades which improve two separate components to combine into a dead loss.

C. VDD Avoids Cost Growth and Performance Erosion

What does a component design team do when faced with a set of requirements for performance, weight, reliability, and so on? In a deterministic world, or for a simple component, the task would be to pick a design that meets the requirement. However, we live in an uncertain world, and components of large aerospace systems are generally complex. In our world, design becomes a process of search and discovery, in which the end point is only revealed gradually, as the result of a series of design choices made under uncertainty. The task of the traditional design engineer is best described as maximizing the probability that the design will meet the requirements. This contrasts strongly with the task of optimal design, which is to design the best component, or more formally, search for the design which will maximize the value of the attributes (measured by the objective function) while not violating interface constraints. Research [7] shows that when a team maximizes the probability of meeting requirements for a complex and nondeterministic design task, the resulting attributes are skewed, with a tight grouping on the slightly better side of each attribute and a much longer tail on the worse side. For example, if a requirement says that a particular component should weigh less than 10 lb, much of the distribution of the resulting weight might be packed between 9.5 and 10, with most of the rest spread between 10 and 15. As a result, even though the median and the mode of the distribution may be less than 10 (on the good side of the requirement), the mean is usually greater than 10. As the components are aggregated into the whole system, it is the mean that is more predictive of system performance. Thus, even when most of the components meet their requirements, it is likely that the system will fall short of the aggregate system requirements. This skewing is an artifact of the requirements process, of the very effort to maximize the probability of meeting component requirements. VDD eliminates the skewing, greatly improving system attributes. The simulations performed in Collopy [7] indicate that this effect alone could account for 50% cost growth and a significant schedule delays on large programs. VDD would eliminate this source of cost growth and delay.

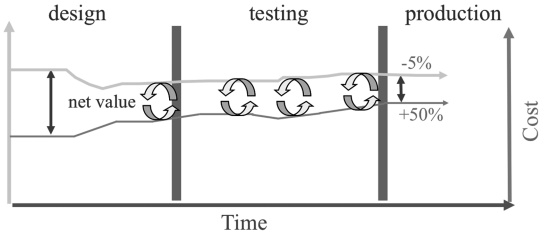

Fig. 13 Performance erosion, cost growth, and redesign.

All three of these effects are exacerbated by the complexity of large systems such as aerospace systems. Figure 12 illustrates the hierarchical organization of an aircraft design, but falls far short of indicating the hierarchical complexity. A typical military fighter aircraft or commercial airliner would have seven to 10 levels in this tree with hundreds of components. All these levels obscure the system design intent from the component design teams, most of whom work for a different firm than the lead systems integrator. With more layers and increased complexity, the gap between the results of optimization and the results of striving to meet requirements builds up from a few percentage points to a tens of percent. With the first roll up of component attributes into system attributes, which usually occurs late in detailed design, the project inevitably finds that the system is too heavy or too inefficient to be useful. The common solution is a redesign, or a series of redesigns, to correct the excess weight or performance shortfall. In these redesigns, the offending attributes are traded for cost. The result is a cost growth of about 50% on average and a schedule delay, or series of schedule delays, to accommodate the redesign. As far as performance, weight, and other attributes, the inevitable result is that these attributes barely meet requirements or fall slightly short (see Fig. 13). It is this shortfall and destructive cycle of redesign, more than anything else, that VDD strives to correct.

V. Conclusions

Today is an exciting time to be participating in the value-driven design movement. New ideas, new energy, and new solutions are emerging from all quarters. VDD is not a particular method or

process. Instead, it is a clarifying way of thinking about systems engineering that will eventually lead to new methods and processes. Many people are adopting the VDD way of thought, beyond only the AIAA VDD Program Committee that the authors have the privilege of leading.

However, adoption of VDD is not without challenges. People seem to naturally grasp requirements as directives more easily than following objective functions. The whole notion of budgeting allocations of attributes like cost, weight, and reliability has the strong forces of tradition and familiarity behind them, whether or not they deliver successful results. Nevertheless, VDD is coming out of the woodwork of the systems community, simply because the time has come for a better approach.

References

[1] Achenbach, J., "NASA's Trajectory Unrealistic, Panel Says," *Washington Post*, 14 Aug. 2009, http://www.washingtonpost.com/wp-dyn/content/article/2009/08/13/AR2009081302244.html [retrieved 2 Sept. 2009].

[2] Valdes-Dapena, P., "Chevy Volt to Get 230 MPG Rating," *CNNMoney.com*, 11 Aug. 2009, http://money.cnn.com/2009/08/11/autos/volt_mpg/ [retrieved 2 Sept. 2009].

[3] Schofield, A., "End of Year is New Goal for 787 First Flight," *Aviation Week and Space Technology*, 27 Aug. 2009.

[4] Schwenn, R. E., Brink, H., Mebane, C. T., Seales, S. C., and Wintfeld, J. R., "Defense Acquisitions: Assessments of Selected Weapon Programs," U.S. Government Accountability Office, Rept. GAO-09-326SP, March 2009, http://www.gao.gov/new.items/d09326sp.pdf [retrieved 2 Sept. 2009].

[5] Augustine, N. R., *Augustine's Laws*, 6th ed., AIAA, Reston, VA, Sept. 2009, pp. 105, 153.

[6] Eveker, K., "The Budgetary Implications of NASA's Current Plans for Space Exploration," Congressional Budget Office, Washington D.C., April 2009, p. 7, http://www.cbo.gov/ftpdocs/100xx/doc10051/04-15-NASA.pdf [retrieved 23 Jan. 2011].

[7] Collopy, P. D., "Adverse Impact of Extensive Attribute Requirements on the Design of Complex Systems," AIAA Paper 2007-7820, 2007.

[8] Hazelrigg, G. A., "A Framework for Decision-Based Engineering Design," the *Journal of Mechanical Design*, Vol. 120, No. 4, Dec., 1998, pp. 653–658. doi:10.1115/1.2829328

[9] Collopy, P. D., "Aerospace System Value Models: A Survey and Observations," AIAA Paper 2009-6560, 2009.

[10] DeBreu, G., *Theory of Value: An Axiomatic Analysis of Economic Equilibrium*, Wiley, NY, 1959, pp. 50–102.

[11] von Neumann, J., and Morgenstern, O., *Theory of Games and Economic Behavior*, Princeton Univ. Press, Princeton, NJ, 1947, pp. 16–31.

[12] Simon, H. A., "The Science of Design: Creating the Artificial," *Sciences of the Artificial*, 2nd ed., MIT Press, Cambridge MA, 1981, pp. 128–159.

[13] Sage, A. P., *Methodology for Large Scale Systems*, McGraw–Hill, New York, 1977, pp. 69, 353.

[14] Keeney, R. L., and Raiffa, H., *Decisions with Multiple Objectives: Preferences and Value Tradeoffs*, Wiley, New York, 1976, p. 50.

[15] Schmit, L. A., "Structural Design by Systematic Synthesis," *Proceedings, 2nd Conference on Electronic Computation*, American Society of Civil Engineers, New York, 1960, pp. 105–122.

[16] Sobieszczanski-Sobieski, J., James, B., and Riley, M., "Structural Optimization by Generalized, Multilevel Decomposition," *AIAA Journal*, Vol. 25, No. 1, Jan. 1987, pp. 139–145. doi:10.2514/3.9593

[17] Cramer, E. J., Dennis, J. E., Frank, P. D., Lewis, R. M., and Shubin, G. R., "Problem Formulation for Multidisciplinary Optimization," *SIAM Journal on Optimization*, Vol. 4, No. 4, Nov. 1994, pp. 754–776. doi:10.1137/0804044

[18] Sobieszczanski-Sobieski, J., Agte, J. S., and Sandusky, R. R., Jr., "Bilevel Integrated Systems Synthesis," *AIAA Journal*, Vol. 38, No. 1, Jan. 2000, pp. 164–172. doi:10.2514/2.937

[19] Sobieszczanski-Sobieski, J., Altus, T. D., Phillips, M., and Sandusky, R., "Bilevel Integrated System Synthesis for Concurrent and Distributed Processing," *AIAA Journal*, Vol. 41, No. 10, Oct. 2003, pp. 1996–2003. doi:10.2514/2.1889

[20] Kodiyalam, S., and Sobieszczanski-Sobieski, J., "Bilevel Integrated System Synthesis with Response Surfaces," *AIAA Journal*, Vol. 38, No. 8, Aug. 2000, pp. 1479–1485. doi:10.2514/2.1126

[21] Sobieszczanski-Sobieski, J., "Integrated System-of-Systems Synthesis," *AIAA Journal*, Vol. 46, No. 5, May 2008, pp. 1072–1080. doi:10.2514/1.27953

[22] Hazelrigg, G. A., *Systems Engineering: An Approach to Information-Based Design*, Prentice–Hall, Upper Saddle River, NJ, 1996, pp. 158–161, 291–311.

[23] Collopy, P. D., "A System for Values, Leadership and Communications in Product Design," *International Powered Lift Conference Proceedings*, P-306, SAE Publications, Warrendale, PA, 1997, pp. 95–98.

[24] Collopy, P. D., "Economic-Based Distributed Optimal Design," AIAA Paper 2001-4675, 2001.

[25] Collopy, P. D., "Value of the Probability of Success," AIAA Paper 2008-7876, 2008.

[26] Biffl, S., Aurum, A., Boehm, B., Erdogmus, H., and Grünbacher, P. (eds.), *Value-Based Software Engineering*, Springer–Verlag, Berlin, 2006, pp. 3–33.

[27] Kirby, M. R., "A Methodology for Technology Identification, Evaluation, and Selection in Conceptual and Preliminary Aircraft Design," Ph.D. Thesis, Georgia Inst. of Technology, Atlanta, 2001.

[28] Baker, A. P., "The Role of Mission Requirements, Vehicle Attributed, Technologies and Uncertainty in Rotorcraft System Design," Ph.D. Thesis, Georgia Inst. of Technology, Atlanta, March 2002.

[29] Hollingsworth, P. M., "Requirements Controlled Design: A Method for Discovery of Discontinuous System Boundaries in the Requirements Hyperspace," Ph.D. Thesis, Georgia Inst. of Technology, Atlanta, March 2004.

[30] Biltgen, P. T., "A Methodology for Capability-Based Technology Evaluation for Systems-of-Systems," Ph.D. Thesis, Georgia Inst. of Technology, Atlanta, March, 2007.

[31] Ender, T. R., "A Top-Down, Hierarchical, System-of-Systems Approach to the Design of an Air Defense Weapon," Ph.D. Thesis, Georgia Inst. of Technology, Atlanta, 2006.

[32] Pfaender, J. H., "Competitive Assessment of Aerospace Systems Using System Dynamics," Ph.D. Thesis, Georgia Inst. of Technology, Atlanta, 2006.

[33] Fernandez Martin. I., "Valuation of Design Adaptability in Aerospace Systems," Ph.D. Thesis, Georgia Inst. of Technology, Atlanta, 2008.

[34] Briceno, S. I., "A Game-Based Decision Support Methodology for Competitive Systems Design," Ph.D. Thesis, Georgia Inst. of Technology, Atlanta, 2008.

[35] Joppin, C., and Hastings, D., "Evaluation of the Value of the Flexibility Offered by On-Orbit Servicing: Case of Satellite Upgrade," AIAA Paper 2003-6366, 2003.

[36] Long, A. M., Richards, M. G., and Hastings, D. E., "On-Orbit Servicing: A New Value Proposition for Satellite Design and Operation," *Journal of Spacecraft and Rockets*, Vol. 44, No. 4, July–Aug. 2007, pp. 964–976. doi:10.2514/1.27117

[37] de Weck, O. L., de Neufville, R., and Chaize, M., "Enhancing the Economics of Communications Satellites via Orbital Reconfigurations and Staged Deployment," AIAA Paper 2003-6317, 2003.

[38] Saleh, J. H., Lamassoure E. S., Hastings, D. E., and Newman, D. J., "Flexibility and the Value of On-Orbit Servicing: New Customer-Centric Perspective," *Journal of Spacecraft and Rockets*, Vol. 40, No. 2, March–April 2003, pp. 279–291. doi:10.2514/2.3944

[39] Ross, A. M., and Hastings, D. E., "Assessing Changeability in Aerospace Systems Architecting and Design Using Dynamic Multi-Attribute Tradespace Exploration," AIAA Paper 2006-7255, 2006.

[40] Ross, A. M., McManus, H. L., Long, A., Richards, M. G., Rhodes, D. H., and Hastings, D. E., "Responsive Systems Comparison Method: Case Study in Assessing Future Designs in the Presence of Change," AIAA Paper 2008-7732, 2008.

[41] Ross, A. M., and Rhodes, D. H., "Using Natural Value-Centric Time Scales for Conceptualizing System Timelines Through Epoch-Era Analysis," *18th Annual International Symposium of INCOSE*, 15–19 June 2008.

[42] Saleh, J. H., and Marais, K., "Reliability: How Much Is It Worth? Beyond Its Estimation or Prediction, the (Net) Present Value of Reliability," *Reliability Engineering and System Safety*, Vol. 91, No. 6, June 2006, pp. 665–673. doi:10.1016/j.ress.2005.05.007

[43] Marais, K. B., and Saleh, J. H., "Beyond Its Cost, the Value of Maintenance: An Analytical Framework for Capturing its Net Present Value," *Reliability Engineering and System Safety*, Vol. 94, No. 2,

Feb. 2006, pp. 644–657.
doi:10.1016/j.ress.2008.07.004

[44] Dubos, G. F., Saleh, J. H., "Identifying and Quantifying the Value of Responsiveness in the Presence of Uncertainty," AIAA Paper 2009-6803, 2009.

[45] Saleh, J. H., "Flawed Metrics: Satellite Cost per Transponder and Cost per Day," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 44, No. 1, Jan. 2008, pp. 147–156.
doi:10.1109/TAES.2008.4516995

[46] Brown, O., and Eremenko, P., "The Value Proposition for Fractionated Space Architectures," AIAA Paper 2006-7506, 2006.

[47] Brown, O., and Eremenko, P., "Application of Value-Centric Design to Space Architectures: The Case of Fractionated Spacecraft," AIAA Paper 2008-7869, 2008.

[48] Brown, O. C., Eremenko, P., and Collopy, P. D., "Value-Centric Design Methodologies for Fractionated Spacecraft: Progress Summary from Phase 1 of the DARPA System F6 Program," AIAA Paper 2009-6540, 2009.

[49] McCormick, D., Barrett, B., and Burnside-Clapp, M., "Analyzing Fractionated Satellite Architectures Using RAFTIMATE—A Boeing Tool for Value-Centric Design," AIAA Paper 2009-6767, 2009.

[50] Maciuca, D., Chow J.K., Siddiqi A., de Weck O. L., Alban S., Dewell L. D., et al., "A Modular, High-Fidelity Tool to Model the Utility of Fractionated Space Systems," AIAA Paper 2009-6765, 2009.

[51] Eichenberg-Bicknell, E., Wisniewski, M. J., Choi, S. W., and Westley, D. M., "Using a Value-Centric Tool Framework to Optimize Lifecycle Cost, Value, and Risk of Spacecraft Architectures," AIAA Paper 2009-6766, 2009.

[52] Collopy, P. D., "Value-Driven Design and the Global Positioning System," AIAA Paper 2006-7213, 2006.

[53] Ross, A. M., "Multi-Attribute Tradespace Exploration with Concurrent Design as a Value-Centric Framework for Space System Architecture and Design," M.S. Thesis, Aerospace Engineering Department, Massachusetts Inst. of Technology, Cambridge, MA, 2003.

[54] Li, Y., Hollingsworth, P., and Mavris, D., "A Concept Selection Methods Developed from a Probabilistic Multi-Criteria Decision-Making Technique Using Utility Theory," *S.A.E. Transactions: Journal of Aerospace*, Vol. 114, 2006, pp. 1486–1495.

[55] Biltgen, P. T., Ender, T., and Mavris D. N., "Development of a Collaborative Capability-Based Tradeoff Environment for Complex Systems Architectures," 44th AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV, AIAA Paper 2006-728, Jan. 2006.

[56] Marsaw, A., German, B., Hollingsworth, P., and Mavris, D., "An Interactive Visualization Environment for Decision Making in Aircraft Engine Preliminary Design," 46th AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV, AIAA Paper 2007-1333, Jan. 2007.

[57] Ashby, M. F., *Materials Selection in Mechanical Design*, 3rd ed., Butterworth Heinemann, Oxford, 2005, pp. 46–77.

[58] Briceno, S., and Mavris, D., "Strategic Decision-Making: Applications of Game Theory in a Systems Approach to Commercial Engine Selection," AIAA 5th ATIO and 16th Lighter-than-Air Systems Technology and Balloon Systems Conferences, Arlington, VA, AIAA Paper 2005-7428, Sept. 2005.

[59] Abbas, A. E., and Howard, R. A., *Foundations of Decision Analysis*, Prentice–Hall, Englewood Cliffs, NJ, 2009, pp. 1–73.

[60] Ross, A. M., and Rhodes, D. H., "Using Natural Value-Centric Time Scales for Conceptualizing System Timelines Through Epoch-Era Analysis," *INCOSE International Symposium 2008*, Utrecht, the Netherlands, June 2008.

[61] Briceno, S. I., and Mavris, D. N., "Applications of Game Theory in a Systems Design Approach to Strategic Engine Selection," 25th International Congress of the Aeronautical Sciences, Paper ICAS 2006-5.4.1, Hamburg, Germany, Sept. 2006.

[62] Forrester, J. W., *Industrial Dynamics*, MIT Press, Cambridge, MA, 1961, pp. 73–76, 81–90, 102–108.

[63] Sterman, J. D., *Business Dynamics, Systems Thinking and Modeling for a Complex World*, McGraw–Hill, Boston, 2000, pp. 513–534.

[64] Hollingsworth, P., Pfaender, H., and Mavris, D., "Program and Design Decisions in an Uncertain and Dynamic Market: Making Engineering Choices Matter," *S.A.E. Transactions: Journal of Aerospace*, Vol. 114, No. 1, 2006, pp. 1486–1495.

[65] Keeney, R. L., *Value-Focused Thinking*, Harvard Univ. Press, Cambridge, MA, 1992, pp. 53–152.

[66] Lave, C. A., and March, J. G., "The Evaluation of Speculations," *An Introduction to Models in the Social Sciences*, Harper & Row, New York, 1975, Chap. 3.

[67] Hollingsworth, P. and Mavris, D., "Identification of the Requirements Space Topology for a Rapid Response Strike System," *S.A.E. Transactions: Journal of Aerospace*, Vol. 110, No. 1, 2002, pp. 663–673.

[68] Mavris, D. N., Baker, A. P., and Schrage, D. P., "Simultaneous Assessment of Requirements and Technologies in Rotorcraft Design," *Annual Forum of the American Helicopter Society*, AHS International, Alexandria, VA, May 2000, pp. 1241–1250.

[69] Saleh, J. H., Hastings, D. E., and Newman, D. J., "Flexibility in System Design and Implications for Aerospace Systems," *Acta Astronautica*, Vol. 53, No. 12, Dec. 2003, pp. 927–944.
doi:10.1016/S0094-5765(02)00241-2

[70] Collopy, P. D., "Balancing Risk and Value in System Development," AIAA Paper 2003-6376, 2003.

[71] Carter, A. B., and White, J. P., *Keeping the Edge: Managing Defense for the Future*, MIT Press, Cambridge, MA, 2001, pp. 194–202.