

## ECONOMIC-BASED DISTRIBUTED OPTIMAL DESIGN

Paul D. Collopy, DFM Consulting, Urbana, Illinois \*

### Abstract

Spacecraft and launch systems are examples of complex products which require a careful balance between competing concerns, such as performance, weight, and reliability, to serve their mission. Complexity requires design by large engineering organizations, so this balance must be achieved across many teams of people working on various components. This paper uses optimization theory to derive a method for distributed optimal design. Each component design team is provided with a separate optimization problem such that, as each team finds the best design solution to their problem, the teams together design the best system. To date, distributed optimal design has been difficult because complex system design spaces have extremely high dimensions over which design objectives are poorly correlated. Instead, the paper proposes that design objectives be expressed as functions in *property spaces*, which have few dimensions and are much smoother than design spaces. Property spaces are generated from design spaces by traditional engineering analysis processes. Economic analysis of all parties to the spacecraft launch and operation is used to construct a top-level value function on the system property space. This function is linearly decomposed into value functions for component property spaces. This provides the needed objective functions for distributed optimal design.

### Introduction

In the 1950's, the engineering community developed an amazing skill: the ability to systematically design systems so complex that they cannot be understood by

any single individual mind. Usually referred to as "systems engineering", the method of hierarchical decomposition of requirements, developed on the intercontinental ballistic missile program, enabled the creation of large software systems, commercial jet aircraft, nuclear submarines, and modern space launch systems and the spacecraft that carried mankind to the moon and back, explored much of the solar system, provide worldwide communication and navigation, and today provides a permanent international presence in space.

Systems engineering, in breaking through the limitations of the human mind<sup>†</sup>, opens the possibility for artifacts of essentially unlimited complexity. Yet it is a most human process. Implicit in the hierarchical decomposition of requirements is a parallel hierarchical organization of the design team (Figure 1). Each specification in the requirements tree is assigned to a corresponding group in the organization structure. (In practice this is not always one-to-one.) Thus, systems engineering is a strategy for organizing a team of people, sometimes numbering in the thousands, into groups of five to ten to work together on a task that is only partially visible to any single group. The product is created by the organization, not by a single person. The product is understood in all its detail by the organization, but not by any individual. Like language, economics and culture, these very complex artifacts are born and exist in society, outside the realm of the individual human mind.

---

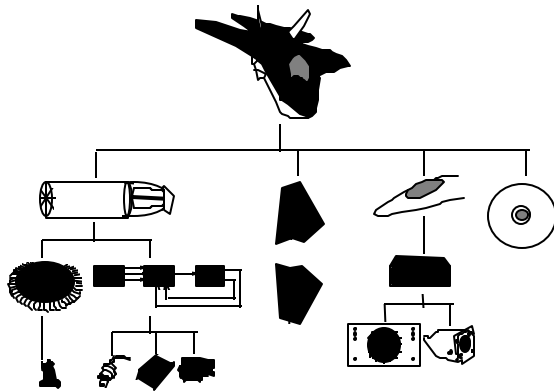
\* Member, AIAA

Copyright © 2001 by DFM Consulting. Published by the American Institute of Aeronautics and Astronautics, Inc. with permission

---

<sup>†</sup> E. Dijkstra expressed this most clearly: "The technique of mastering complexity has been known since ancient times: *Divide et impera* (divide and conquer). ... Some people might think the dissection technique ... (is) a rather indirect and tortuous way of reaching one's goals. My own feelings are perhaps best described by saying that I am perfectly aware that there is no Royal Road to Mathematics, in other words, that I only have a very small head and must learn to live with it." [Dijkstra, 1965]

## Design Requirements Decomposition



## Design Organization Hierarchy

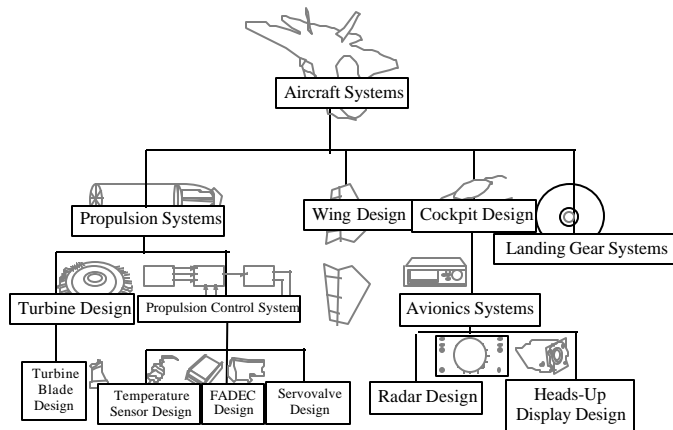


Figure 1: Organization hierarchy parallels design decomposition

Thus system engineering, by transforming design from an individual to an essentially social activity, and by exploding the creative capacity of society, ranks with the most important developments of the century just passed. Yet it is not without its flaws, and one of these is the subject of this paper. Figure 2 illustrates the following story, which is invented, but all too typical of the workings of the systems engineering process:

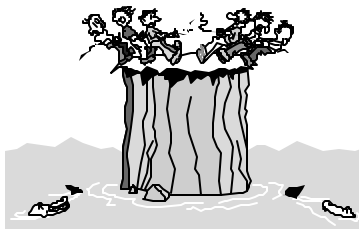
*Tillie's Foundry has won the contract for the turbopump housing on the new SpaceTruck launcher program. Manufacturing the first article indicates that the titanium housing will be more expensive than anticipated. In fact, Tillie's will lose money at the projected production cost. However, the housing is also smaller and lighter than anticipated. The design engineers show that by changing the*

*titanium housing to aluminum alloy, the cost can be reduced \$10,000 and the weight will only increase 40 lbs., which will still be in spec.*

*Meanwhile, Acme Ringrolling is struggling with a weight problem on the payload union ring. The titanium ring design has proved to be 10 lbs. over the weight specification. A carbon-carbon ring design is used, which reduces the weight 15 lbs. although it raises the cost \$80,000. Fortunately, Acme negotiated a favorable price for its share of the SpaceTruck and can absorb the extra cost.*

Each of the actions taken by the contractors was reasonable to them, given their specifications. Globally, however, the two actions are inconsistent. Together, they result in a launch system that weighs 25 lbs. more and costs \$70,000 more—it is worse by any measure.

Such contradictions are inherent in the specification system, and their cost to complex products is equivalent to doubling the manufacturing cost, as will be reiterated toward the end of the paper. First, though, the concept of optimal design will be explored.



<b>Housing</b>	+ 40 lbs.	- \$10,000
<b>Ring</b>	- 15 lbs.	+ \$80,000
<b>Combined</b>	+ 25 lbs.	+ \$70,000

Figure 2: Inconsistent design occasioned by specifications

## Optimization

Optimization means choosing the best. Optimal design suggests a method that produces the best design. Optimal design methods compare favorably to good design methods, for who would prefer a good design to the best design?

Perfect optimization would consist of three steps:

1. Generate all possible designs
2. Evaluate each design
3. Choose the design with the highest value.<sup>‡</sup>

In most cases of interest, practical optimization is less than perfect. There are very many possible designs, usually an infinite number, so only some of the designs are evaluated. Optimization theory focuses on methods that locate likely good results by evaluating a finite number of alternatives. Examples of the extensive literature in this field are Luenberger [1989] and Rich and Knight [1991, chapters 2 and 3].

### Optimal Design Process

Optimal design leads to a formal concept of the design process, such as the cycle pictured in Figure 3. Starting on the right hand side, the designer describes the proposed design by a set of parameters, such as the length and thickness of the crank in the drawing, the spacing of the forks, and the crank angle. She elaborates these parameters into a detailed definition of the part, perhaps a mechanical drawing and material specification. Next, on the lower left arc, the definition is analyzed to determine its properties, such as weight, life and manufacturing cost. On the upper left arc, the properties are evaluated. This evaluation process will be discussed extensively below. Depending on the result of the evaluation, another design cycle may be attempted.

For an important design, one trip around the cycle is not enough. An attempt will be made to select a set of parameters that improves the design (the upper right

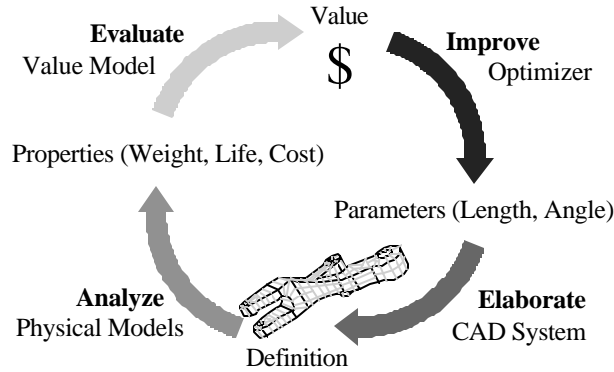


Figure 3: Formal Design Cycle

arc). An optimal design algorithm will pair each set of parameters with the value generated by elaborate-analyze-evaluate applied to that set, and use the previous parameter-value pairs to guess each new set of parameters. Without such algorithms, designers accomplish more or less the same thing, relying on intuition.

Notice that the formal design cycle, although inspired by optimal design, can also be fit to the traditional specification-based process. In this case, *evaluate* is a binary function: either the properties meet the specification or they do not. If they meet the spec, the cycle is usually terminated. If they do not, parameters are changed and another design is attempted.

### Evaluate

Of the four arcs in figure 3, the upper right arc, *evaluate*, is the least developed in terms of theory or tools to support the engineer. Optimization theory presumes a procedure for evaluating options, and although the development of such procedures is outside its scope, the theory does inform us of the properties an evaluator must possess. Luenberger [1989, p. 1] calls the evaluator *the objective*. The objective is a function with all possible designs as its domain. The range of the objective function is an ordinal value, so that designs may be compared.

### Nature of the Objective Function

The necessity for an ordinal objective function is more fundamental than the methodology of optimization. Any method<sup>§</sup> which unambiguously finds the best of a set of options must utilize an objective function that is at least ordinal. Ordinality means the objective function  $\pi(x)$  must satisfy two properties:

<sup>‡</sup> The precise requirement is to choose a design such that no other design has a higher value. If there are many such designs, they are all *best*, and any should be satisfactory. For easy conceptualization, the paper discusses only cases where a single design is best, but all results extend to the more general case.

<sup>§</sup> *Method* here means a process that one practitioner can teach to another person, such that the second person, practicing the process, achieves equivalent results

Let  $\pi$  be an ordinal function with domain  $X$ , and let  $x_1$ ,  $x_2$  and  $x_3$  be members of  $X$ .

**Order Property:** Given two alternatives  $x_1$  and  $x_2$ , exactly one of the following cases is true:

1.  $\pi(x_1) > \pi(x_2)$
2.  $\pi(x_2) > \pi(x_1)$
3.  $\pi(x_1) = \pi(x_2)$

In the first case,  $x_1$  is preferred to  $x_2$ . In the second,  $x_2$  is preferred to  $x_1$ . In the third case, we are indifferent toward  $x_1$  and  $x_2$ .

**Transitive Property of Inequality:** If  $\pi(x_1) \geq \pi(x_2)$  and  $\pi(x_2) \geq \pi(x_3)$ , then  $\pi(x_1) \geq \pi(x_3)$

**Corollary—The Transitive Property of Equality:** If  $\pi(x_1) = \pi(x_2)$  and  $\pi(x_2) = \pi(x_3)$ , then  $\pi(x_1) = \pi(x_3)$

*Proof:* By definition, if  $\pi(x_1) = \pi(x_2)$ , then  $\pi(x_1) \geq \pi(x_2)$ . Similarly,  $\pi(x_2) \geq \pi(x_1)$ . Similarly,  $\pi(x_2) = \pi(x_3)$  implies  $\pi(x_2) \geq \pi(x_3)$  and  $\pi(x_3) \geq \pi(x_2)$ . Therefore,  $\pi(x_1) = \pi(x_2) = \pi(x_3)$  implies  $\pi(x_1) \geq \pi(x_2)$  and  $\pi(x_2) \geq \pi(x_3)$ , so by the transitive property  $\pi(x_1) \geq \pi(x_3)$ . Also  $\pi(x_3) = \pi(x_2) = \pi(x_1)$  implies  $\pi(x_3) \geq \pi(x_2)$  and  $\pi(x_2) \geq \pi(x_1)$ , so  $\pi(x_3) \geq \pi(x_1)$ . The order property implies that if  $\pi(x_1) \geq \pi(x_3)$  then  $\pi(x_1)$  is not less than  $\pi(x_3)$ . Similarly, if  $\pi(x_3) \geq \pi(x_1)$  then  $\pi(x_3)$  is not less than  $\pi(x_1)$ . The order property also implies that if  $\pi(x_1)$  is not less than  $\pi(x_3)$  and  $\pi(x_3)$  is not less than  $\pi(x_1)$ , then  $\pi(x_1)$  must equal  $\pi(x_3)$ . •

The *Real Numbers* are the complete ordered field for which transitivity of inequality applies. It is generally convenient to use Real functions as objective functions because of the arithmetic operations defined over Real numbers. However, Real numbers specify more than is necessary for ordinal values. One significant consequence is that a Real objective function is unchanged by a monotonic transformation, as follows:

**Theorem—Invariance of order under monotonic transformation:** Let  $\pi_1$  be a real function on domain  $X$ , and let  $x_1$  and  $x_2$  be members of  $X$ . Let  $h$  be a monotonically increasing real function on the domain of real numbers, and let  $y_1$  and  $y_2$  be real numbers. *Monotonically increasing* means that if  $y_1 > y_2$  then  $h(y_1) > h(y_2)$ . Let  $\pi_2$  be the composite function of  $h$  and  $\pi_1$ , such that  $\pi_2(x) = h(\pi_1(x))$  for all  $x$  in  $X$ . Then preferences under  $\pi_1$  will also hold under  $\pi_2$ . That is,

1. If  $\pi_1(x_1) > \pi_1(x_2)$ , then  $\pi_2(x_1) > \pi_2(x_2)$ .

2. If  $\pi_1(x_1) < \pi_1(x_2)$ , then  $\pi_2(x_1) < \pi_2(x_2)$ .

3. If  $\pi_1(x_1) = \pi_1(x_2)$ , then  $\pi_2(x_1) = \pi_2(x_2)$ .

*Proof:*

Case 1: Given  $\pi_1(x_1) > \pi_1(x_2)$ . Because  $h$  is monotonically increasing,  $h(\pi_1(x_1)) > h(\pi_1(x_2))$ . By definition of  $\pi_2$ , then,  $\pi_2(x_1) > \pi_2(x_2)$ .

Case 2: Given  $\pi_1(x_2) > \pi_1(x_1)$ . Because  $h$  is monotonically increasing,  $h(\pi_1(x_2)) > h(\pi_1(x_1))$ . By definition of  $\pi_2$ , then,  $\pi_2(x_2) > \pi_2(x_1)$ .

Case 3: Given  $\pi_1(x_1) = \pi_1(x_2)$ . Because  $h$  is a function,  $h(\pi_1(x_1)) = h(\pi_1(x_2))$ . By definition of  $\pi_2$ , then,  $\pi_2(x_1) = \pi_2(x_2)$ . •

The following corollary will prove useful:

**Corollary—Invariance of order under constant shift:**

Let  $\pi_1$  be a real function on domain  $X$ , and let  $x_1$  and  $x_2$  be members of  $X$ . Let  $c$  be a real number. Let  $\pi_2$  also be a real function on domain  $X$  such that  $\pi_2(x) = \pi_1(x) + c$  for all  $x$  in  $X$ . Then preferences under  $\pi_1$  will also hold under  $\pi_2$ .

*Proof.* Let  $h$  be a real function on the domain of real numbers such that  $h(y) = y + c$ . Then  $\pi_2(x) = h(\pi_1(x)) = \pi_1(x) + c$ . The slope of  $h$  is the derivative of  $h$  with respect to  $y$ , which is one. Therefore,  $h$  is monotonic, so preferences under  $\pi_1$  will also hold under  $\pi_2$  by the Theorem above: *Invariance of order under monotonic transformation.* •

With this understanding of the nature of objective functions in general, specific objective functions for the design of launch systems can be explored.

### Traditional Financial Goals

Product performance impacts program profitability because customers will pay more for a launch system that is higher in performance. How much more? Commercial customers use capital investment analyses to determine how much they are willing to pay.

A capital investment analysis compares cash flow in the case that the investment is made to cash flow in the case that the investment is not made (differential cashflow). Luenberger [1998] provides an excellent theoretical basis and lucid examples of this kind of analysis. According to Luenberger, the superior

measure of the value of the investment is the net present value of the differential cashflow.

### Net Present Value

A *cashflow* is cash received or cash paid out at a particular time. Net present value measures a set of cashflows by assigning a value to each flow depending on the time at which it occurs and summing the values. The key idea is that a dollar today is worth more than a dollar tomorrow. How much more depends on the preferences of the person or organization assessing the value. The benchmark for measuring this preference for money over time is the differential value of a dollar received today to the value of a dollar received one year from today, ratioed to the value of the future dollar:

$$r = \frac{\text{dollar today} - \text{dollar in one year}}{\text{dollar in one year}} \quad (1)$$

$r$  is called the *discount factor*, and the reduction in value of future year cashflows is called *discounting*. The US government uses a discount factor of 7%, corporations typically range from 10% to 30%, and individuals run somewhat higher than corporations.

Two assumptions are normally made to simplify present value analysis:

1. The ratio in (1) is unaffected by the amount of money is used. That is, the discount factor for \$1,000 or \$1,000,000 is the same as the factor calculated for \$1.
2. The ratio in (1) is constant over any one year interval in the past or future. Thus, the differential value of a dollar three years from today to a dollar four years from today, ratioed to a dollar four years from today, also equals  $r$ , since the interval between the cashflows is still one year.

### Spacecraft Operation

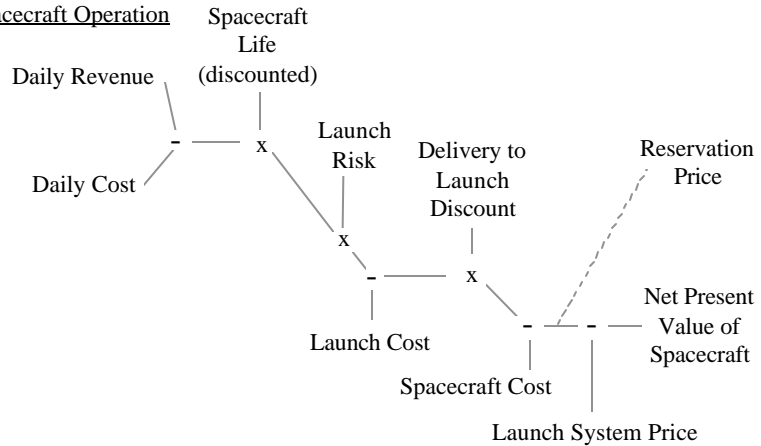


Figure 4: Net present value of spacecraft

If we start with the basis that the value of \$1 today is \$1 (the *present value* basis), the following formula will give the present value for any single cashflow:

$$\text{Present Value} = \text{cashflow} \cdot \left( \frac{1}{1+r} \right)^t \quad (2)$$

where  $t$  is the number of years into the future when the cashflow occurs. For a series of cashflows  $c_i$  that occur at different times  $t_i$ ,

$$\text{Net Present Value} = \sum c_i \cdot \left( \frac{1}{1+r} \right)^{t_i} \quad (3)$$

where cashflows in are valued positively and cashflows out are valued negatively.

### Reservation Price

A corporation planning to operate a spacecraft will perform an analysis like that illustrated in Figure 4 to determine whether the venture is worthwhile. (The concept of operations in figure 4 and continued throughout the paper is that the launch system is a physical product sold to the satellite operator, who separately procures the satellite and launch services. This concept is chosen for its simplicity, but any other concept can be modeled as effectively.)

*Reservation price* is the highest price that a customer is willing to pay for a product or service in the absence of competition. The reservation price of a launch system is the highest price that the satellite operator will pay for the launch. If there were no competition, the operator would proceed with the launch so long as the net

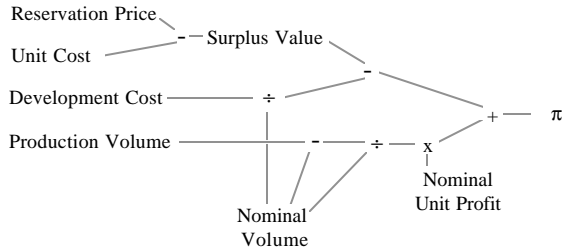
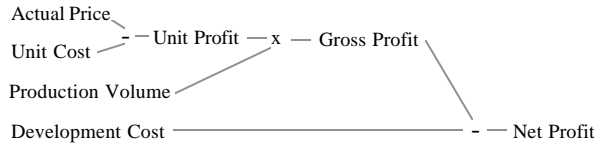


Figure 5: Derivation of Surplus Value

present value calculated in Figure 4 were positive. Thus the reservation price is the price that makes the net present value zero. This is the value coming into the last subtraction in Figure 4, which is marked as reservation price, because if the launch system price is equal to this reservation price, the last subtraction subtracts reservation price from reservation price yielding a net present value of zero.

Competition causes sales price to be less than reservation price. Nevertheless, reservation price is significant because every dollar change in reservation price will cause a dollar change in sales price. The reason is that two products can only compete in the same market segment if they provide customers equivalent value. That is, the analysis via Figure 4 of a launch system and its competitor must yield approximately the same net present value, or else the lower value system will go out of business. (This assumes the two systems are true competitors, serving the same types of spacecraft). Thus, the net present value output, given a particular launch system, is fixed by the competition. This output is the difference between the reservation price of the launch system and the actual price, as shown in Figure 4. Since this difference is fixed,

it implies that every change in reservation price must be reflected by a change of the same dollar amount in actual price (an idealization, but roughly conforming to actual market performance).

In a nutshell, given a constant competitor, if you are considering two launch system designs, A and B, and A is worth \$m more to your satellite customer than B, then you should be able to get your customer to pay \$m more for A than for B.

### Using Reservation Price

A product such as a launch system is a capital investment, so that launch system design decisions are capital investment decisions. The objective is to maximize the program net present value, shown in the upper part of Figure 5 as net profit. In design optimization at a for-profit corporation, therefore, the objective function is the net profit model shown.

For use by engineers, the model in the lower half of Figure 5 is preferable for two reasons: the objective is in dollars per unit, like manufacturing cost, which is more quickly grasped by design engineers; and the model is insensitive to actions of the competitor and less sensitive to program changes, like unit volume. Note that the objective  $\pi$  is equivalent to the objective net profit because, except that nominal unit profit is used in place of actual unit profit in the correction term,  $\pi$  is a monotonic function of net profit:

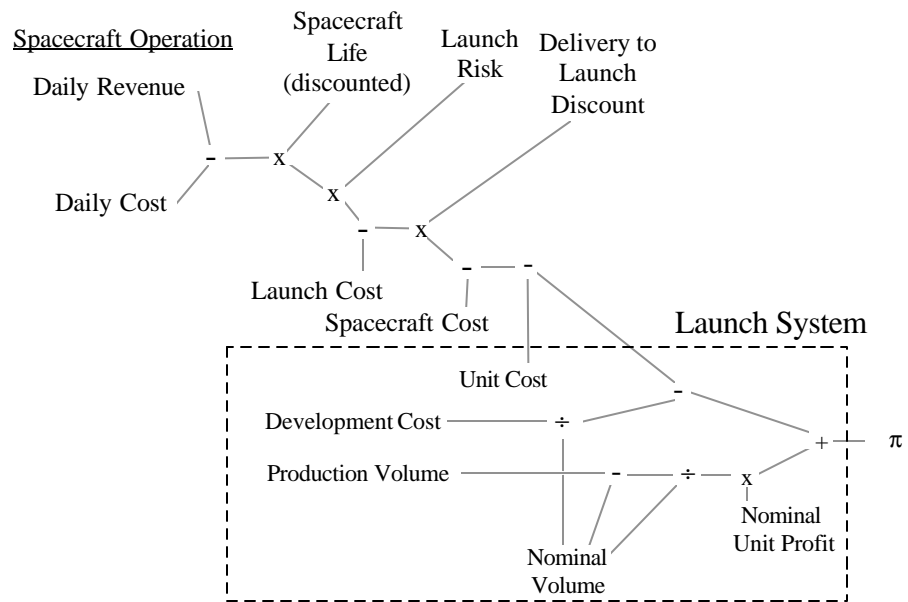


Figure 6: Launch System Value Model

$$\pi = \frac{\text{Net Profit}}{\text{Nominal Volume}} + \text{Reservation Price} - \text{Actual Price} \quad (4)$$

Note that Nominal Volume is a constant, and Reservation Price minus Actual Price is a constant (as discussed above), so  $\pi$  is a monotonic function of Net Profit.

Figure 6 combines the lower part of Figure 5 with Figure 4 to provide a complete value model for the launch system. Figure 6 suggests properties of the launch system which affect its value,  $\pi$ . Clearly, the manufacturing and development cost impact value. Performance of the launch system also affects the launch and operation characteristics in the upper left of Figure 6 as follows:

1. Revenue per day earned by the spacecraft in orbit. The orbital altitude achieved by the launch system may affect the field of view of spacecraft sensors. The orbital inclination may allow the spacecraft to observe high latitudes inaccessible to competing spacecraft. A highly elliptical orbit may allow a spacecraft to spend a large fraction of its time over a region with greater revenue potential, such as North America or Western Europe for a communication satellite. A low perigee may provide enhanced resolution for critical observations. Precision launching minimizes the propulsion necessary for the spacecraft to rendezvous with a space station or quickly transfer to lunar orbit. A higher payload capacity can permit a communication satellite to use more power.
2. Cost per day to operate the spacecraft. Better orbital inclination may reduce the required number of ground stations. Higher orbital altitude can also reduce ground station requirements. A launch system that can accommodate odd-shaped payloads may allow larger antennae or less complex deployment mechanisms.
3. Orbital Life. If the launch system reduces the heat and vibration imparted to the payload, spacecraft life can be enhanced. A manned launch system can reduce infant mortality by manually correcting some startup problems, such as faulty deployment of solar panels.
4. Launch cost. Launch cost includes payload packaging costs which are largely determined by launch system requirements. Also, payload propulsion and staging are influenced by the initial orbital capability of the launch system. Fuel cost is another element which is impacted by launch system fuel type and specific impulse.
5. Delivery interval. The interval between the time the spacecraft is ready for payload processing and the time it becomes operational in orbit reduces the net present value of the launch system by the discount corresponding to the length of the interval. There may be other cost elements tied to the delivery interval. Fast delivery allows more flexibility to configure spacecraft to the needs of the market. Delivery interval is not only impacted by the time required for payload processing and launch operations, it is also impacted by the reliability and frequency of the launch schedules. The delivery date for payload processing might be fixed a year or years in advance by the spacecraft manufacturing cycle. If, when the spacecraft is ready, the scheduled launch is delayed or canceled, and particularly if the spacecraft must then wait a year or more for another launch opportunity, the impact on spacecraft net present value is significant. Quantifying this necessitates assessing the probabilities of short and long delays and using the expectation (probability weighted average) of the delay in the launch system value model.
6. Spacecraft Cost. Many of the factors cited above also impact spacecraft cost. For example, a powerful launch system reduces the spacecraft orbital injection propulsion system required to achieve a final high-altitude or non-terrestrial orbit.

These examples show how the model in Figure 6 relates customer-relevant properties of the launch system and the costs in the dashed box to  $\pi$ . Figure 6 can therefore be translated into a mathematical function that calculates  $\pi$  from the launch system properties. This is the objective function that performs the evaluate arc in Figure 3—the guide for preliminary design of the launch system. The  $\pi$  function is a very valuable tool in itself,

but it is also the first prerequisite for distributed optimal design.

## Distributed Optimal Design

This section will show how to derive objective functions for components from the objective function for the entire system ( $\pi$ ), so that components can be separately designed according to the design cycle in Figure 3, and providing that the components are optimal, the entire system will prove to be optimal.

### Assumptions and Limitations

The following caveats apply to this method:

1. **Preliminary Design.** It is assumed that a preliminary design of the overall system has been done and the properties of the overall system have been estimated. Further, it is assumed that the final design will be close to the preliminary design. Operationally, this means that the properties estimated for the preliminary design will be within about 10% of the properties of the final design.
2. **Partitioned Design.** It is assumed that the system has been partitioned into components and the interfaces between components have been specified. The distributed design method below will produce an optimal design within the constraints of the fixed partitioning and fixed interface.
3. **Extensive Properties.** The optimization method applies to the extensive properties of the components. *Extensive properties* are those component properties which collectively impact the overall properties of the system. For example, component weights, costs and reliability collectively determine system weight cost and reliability. Other extensive properties are performance measures, design life, development time and cost, and refurbishment cost for reusable components.

### Distributed Optimization Defined

Given:

1. The overall system is defined by properties  $x_1, x_2, \dots, x_n$ . These properties are coordinates of a property space X, and the particular properties

of the system at any point in the design is a vector  $\bar{x}$ .

2. Each component  $i$  is defined by extensive properties  $y_1, y_2, \dots, y_{m_i}$ . These properties are coordinates of a property space  $Y_i$  and the particular properties of the system at any point in the design is a vector  $\bar{y}_i$ .
3. The objective is to maximize  $\pi(\bar{x})$ .

Distributed optimization, using the approach illustrated in figure 3 can be achieved if, for every component  $i$ , an objective  $\phi_i$  can be defined such that if the design for every component  $\bar{y}_i$  maximizes  $\phi_i(\bar{y}_i)$  then the resulting system  $\bar{x}$  will maximize  $\pi(\bar{x})$ . More formally, if for every component  $i$ , a design with properties  $\bar{y}_i^*$  is found such that  $\phi_i(\bar{y}_i^*) \geq \phi_i(\bar{y}_i)$  for all possible designs, the respective properties of which are  $\bar{y}_i$  (the feasible region of the component property space Y) then the resulting system property vector  $\bar{x}^*$  will satisfy the condition  $\pi(\bar{x}^*) \geq \pi(\bar{x})$  for all feasible  $\bar{x}$  in X.

### Derivation

Concatenate all component property vectors  $\bar{y}_i$  into a single very long vector  $\bar{z}$ , the collective properties of all components. By the definition of extensive properties, there is a function  $h$  (the composition function) such that, for a particular system with system properties  $\bar{x}$  and component properties  $\bar{z}$ ,  $\bar{x} = h(\bar{z})$ .  $h$  is a vector function, so it is an ordered set of scalar functions, such as a weight function, a cost function, a performance function, and so on.

The objective function over the domain of component properties is then  $\pi(h(\bar{z}))$ . If this can be linearized, it can be decomposed into independent linear objective functions for the individual components.

Because property spaces tend to be smooth,  $\pi(h(\bar{z}))$  can be linearized by a Taylor's series expansion in the vicinity of the preliminary design  $\bar{x}'$ , corresponding to component design reference properties  $\bar{z}'$ , as follows:

$$\begin{aligned} \pi(h(\bar{z})) &= \\ \pi(h(\bar{z}')) + \nabla\pi(\bar{x})|_{\bar{x}'} \cdot J_h|_{\bar{z}'} \cdot (\bar{z} - \bar{z}') + O^2 \end{aligned} \quad (5)$$

where  $J_h$  is the Jacobian matrix of  $h$  (the matrix of all the partial derivatives of the components of  $\bar{x}$  versus the components of  $\bar{z}$ ), and  $O^2$  represents the higher order terms of Taylor's series. We can ignore  $O^2$  and incur little error since we are in the vicinity of  $\bar{x}'$  in a smooth space.

Expanding the vector multiplications,

$$\begin{aligned} \pi(h(\bar{z})) &\cong \\ \pi(h(\bar{z}')) + \sum_j \left( \sum_{k=1}^n \frac{\partial\pi}{\partial x_k} \cdot \frac{\partial x_k}{\partial z_j} \right)_{\bar{z}'} \cdot (z_j - z'_j) \end{aligned} \quad (6)$$

Recall it was proven that adding a constant to an objective function leaves preferences under it unchanged. Therefore, we can drop the constant terms in (6) to simplify it:

$$\pi(h(\bar{z})) \approx \sum_j \left( \sum_{k=1}^n \frac{\partial\pi}{\partial x_k} \Big|_{x'_k} \cdot \frac{\partial x_k}{\partial z_j} \Big|_{z'_j} \right) \cdot z_j \quad (7)$$

This is a linear function on  $\bar{z}$ . The maximum of a linear function is achieved when each term is independently maximized. Therefore, the terms of (7) corresponding to each component  $i$  are the desired component objective function  $\phi_i$ :

$$\phi_i(\bar{y}_i) = \sum_{j=1}^{m_i} \left( \sum_{k=1}^n \frac{\partial\pi}{\partial x_k} \Big|_{x'_k} \cdot \frac{\partial x_k}{\partial y_j} \Big|_{y'_j} \right) \cdot y_j \quad (8)$$

Finding the component designs which independently maximize these component objectives will yield an optimal system design.

## Significance

The process just described provides a method to create evaluation functions for system components so that the components can be designed independently, according to the cycle illustrated in Figure 3, and yield an optimal design. Investigations not yet published suggest that this approach has the potential to reduce manufacturing

costs 50% versus the traditional specification-based design method.

## Conclusions

This paper provides a method to perform distributed optimal design of complex systems. The method consists of

1. Using a formal design cycle that includes evaluating the projected properties of the design.
2. Using capital investment analysis to create a system-level objective in the form of a function that provides an ordinal valuation of a set of system properties.
3. Deriving component level objectives from the system level objective, so that individual components can be optimally designed independently.

It is recommended that component level objectives largely replace traditional specification in providing hierarchical guidance to large design teams.

## References

- Dijkstra, Edsger. "Programming Considered as a Human Activity." Pages 213-217 in *Proceedings of the 1965 IFIP Congress*, North-Holland, Amsterdam, **1965**.
- Luenberger, David G. *Linear and Nonlinear Programming*. Second Edition, Addison-Wesley, Reading MA, **1989**.
- Luenberger, David G. *Investment Science*. Oxford University Press, Cambridge MA, **1998**.
- Rich, Elaine, and Knight, Kevin. *Artificial Intelligence*. Second Edition, McGraw-Hill, New York, **1991**.