2011 NDIA GROUND VEHICLE SYSTEMS ENGINEERING AND TECHNOLOGY SYMPOSIUM
SYSTEMS ENGINEERING AND INTEGRATION (SE) MINI-SYMPOSIUM
AUGUST 9-11 DEARBORN, MICHIGAN

IDENTIFYING THE COMPONENT AND ARCHITECTURAL DRIVERS OF COST IN MILITARY SYSTEMS

Robert Bordley
Lead Associate
Booz Allen Hamilton
Troy, MI 48084

Troy Peterson
Senior Associate
Booz Allen Hamilton
Troy, MI 48084

Frederick Samson
Associate
Booz Allen Hamilton
Troy, MI 48084

ABSTRACT

Department of Defense (DoD) systems are often highly complex, costly and have extraordinarily long life cycles. Due to these characteristics requirements that these systems will need to meet over their life cycle are highly uncertain. To meet future requirements more rapidly at a lower cost requires an understanding of how to manage uncertainty and architecture to make these complex systems more flexible, adaptable and affordable. This paper proposes an alternative approaches to traditional development through managing uncertainty and architecture in an iterative fashion with decision analysis methods. Several specific methods and tools are discussed to include: Influence Diagrams, Design of Experiments, Design Structure Matrix and Target-Oriented Utility. Collectively the approach identifies the component and architectural drivers of cost in military systems.

INTRODUCTION

Motivation

Future Department of Defense (DoD) requirements are highly uncertain due to the dynamics in the operational context, agile adversaries, decreased innovation cycles and alternative unknown futures. As a result, time-consuming and costly redesign is no longer effective making alternative approaches to traditional development and acquisition essential. This paper will propose an alternative approach which couples uncertainty management and architecture management methodologies to make informed decisions and reduce total life cycle costs. More specifically, it will outline approaches to identify and manage component and architectural drivers of cost due to future uncertainties.

Design requirements may change because of changes in customer needs or because the designer learns that certain requirements are not feasible. Since the designer does not know how design requirements may change, the designer must make decisions under uncertainty. This uncertainty creates design risk.

There are three ways of dealing with design risk:

(1) Reduce the uncertainty (uncertainty management)
(2) Reduce the design's sensitivity to uncertainty (architecture management)
(3) Accept the uncertainty and make the best design decisions given that uncertainty (decision analysis)

Uncertainty management reduces the uncertainties affecting the architecture while architecture management minimizes the impact of uncertainty through system flexibility. Due to their interdependence these two techniques must be used iteratively to explore the trade space for an optimal solution which balances cost and capability. Decision analysis then addresses whatever uncertainties are not easily mitigated with uncertainty management and architecture management.

Uncertainty Management includes:

(1) Collecting information on how customer needs are likely to change and which requirements are likely to change
(2) Eliminating the physical cause of the uncertainty

(3) Postponing the design until the uncertain variables are known

Architecture management includes

(1) Reducing the degree to which components are directly sensitive to the uncertainty
(2) Reducing the degree to which components are sensitive to changes in other components and hence are indirectly sensitive to the uncertainty.

The design structure matrix is a general approach for dealing with sensitivity to uncertainty

Decision analysis

(1) Accepts the uncertainty and the sensitivity to uncertainty and move forward with a decision in light of the uncertainty
(2) Explicitly accepts the risk of a decision leading to an infeasible design or a product the customer no longer wants
(3) Explicitly pre-specifies a tradeoff between performance and risk
(4) Makes the decision optimizing this tradeoff given explicit quantification of both the performance and the risk associated with each decision

The next three sections of this paper discuss the relevant tools for each of these general sets of methods.

APPROACH

UNCERTAINTY MANAGEMENT

Overview

Uncertainty reduction tools involve several steps

(1) Describe the potential uncertainties. This requires envisioning alternate futures for operational context, mission, potential technological advances etc. and assessing their probability of occurrence and how that probability changes over time. It requires understanding how such uncertainties are possibly driven by more fundamental uncertainties (political, economic, social etc.). Brainstorming is often a useful tool.

(2). Categorizes *these issues.* Issues may be
  a. criteria: factors that are of immediate value to the client
  b. chances: factors that affect achievement of the criteria but are not controlled by the client. This involves envisioning alternate futures, their probability and how that uncertainty changes over time.
  c. Choices: factors that affect achievement of the criteria but are controlled by the client
  d. Constituencies: Individuals whose preferences must be consulted in determining the problem criteria
    (i) For each criterion, repeatedly ask why that criterion is important until we reach a set of highest-level criteria from which all other criteria are derivative. (This technique is sometimes known as the five Whys.)
    (ii) Define a deterministic multi-objective measure of performance which describes performance on these highest-level criteria
      a. Identify first-level factors that immediately determine the design performance on the objective function
      b. Identify second-level factors that immediately determine the performance on the first-level factors
      c. Continue until all brainstormed uncertainties and decisions are identified or until the analyst is satisfied that the model is comprehensive enough. This creates an influence diagram.
      d. This process can be facilitated with influence diagram software (Morgan and Henrion, 1990). This software is widely used in many industries for identifying factors affecting the probability of system elements being changed. Influence diagram software couples the development of a visual model and analytical model, which allows the analytical model to be checked using the visual model. This visual cross-check avoids the many hidden errors, which as industry surveys show; arise in spreadsheet software like MS Excel.

(4) Perform a sensitivity analysis on the end-point uncertainties of the influence diagram to determine which uncertainties, when varied over their range, cause the greatest chance in value

  a. Use a tornado chart (a two-side vertical Pareto chart) to identify the most critical uncertainties
  b. Estimate the value of reducing these uncertainties (i.e. the critical uncertainties.)
  c. For the critical uncertainties,
    i. determine how to reduce these uncertainties either by collect information on eliminating the physical cause of these uncertainties
    ii. Determine which decisions could be deferred until after new information on the uncertainties appears. Determine the value of deferring such decisions.

A central tool in this procedure is the influence diagram.

Influence Diagrams

When people want to communicate with each other about complicated situations, they often draw diagrams with bubbles and arrows. One such diagram, the influence diagram, is especially intuitive and provides good notation for creating, understanding and explaining models for policy and risk analysis. Influence diagrams were developed by the decision-analysis community as a representation for working with knowledge, uncertainties, objectives and decisions.

Figure 1 shows an example influence diagram for the drivers of military costs. This diagram indicates that there are two relevant criteria, cost and loss of human life. These are indicated in red by hexagons. Suppose we focus on cost. The diagram indicates that cost is a function of RDTE (research, developing, train and evaluation) costs, operations maintenance Costs and procurement costs. Embedded in the red hexagon, if you were working with influence diagram and could click on it, is an equation specifying how cost is a function of these three variables.

Now suppose we focus on procurement costs. Procurement costs and written as a function of supplier variable costs and amortized supplier fixed costs. If we focus on amortized fixed costs, these are written as a function of supplier fixed costs and units purchased. (If we clicked on amortized fixed costs in the software, it would indicate that amortized fixed costs is suppler fixed costs divided by units purchased.)

Units purchased is a rectangle indicating that it is a decision which the military can make. Typically the decision node contains a vector of possible decisions the military might make. This will cause the software to compute the total costs as a vector where every element of the vector is associated with each possible decisions. Meanwhile supplier fixed costs are written as specialized tooling and plant overhead. They are both ovals meaning that the military does not control these costs and also does not have knowledge on what these costs are. We can similarly interpret every other part of this tree.

Since any one page diagram is typically an oversimplification, some nodes are written as modules. For example, we may create a complex sub-model specifying specialized tooling as a function of many other variables. To

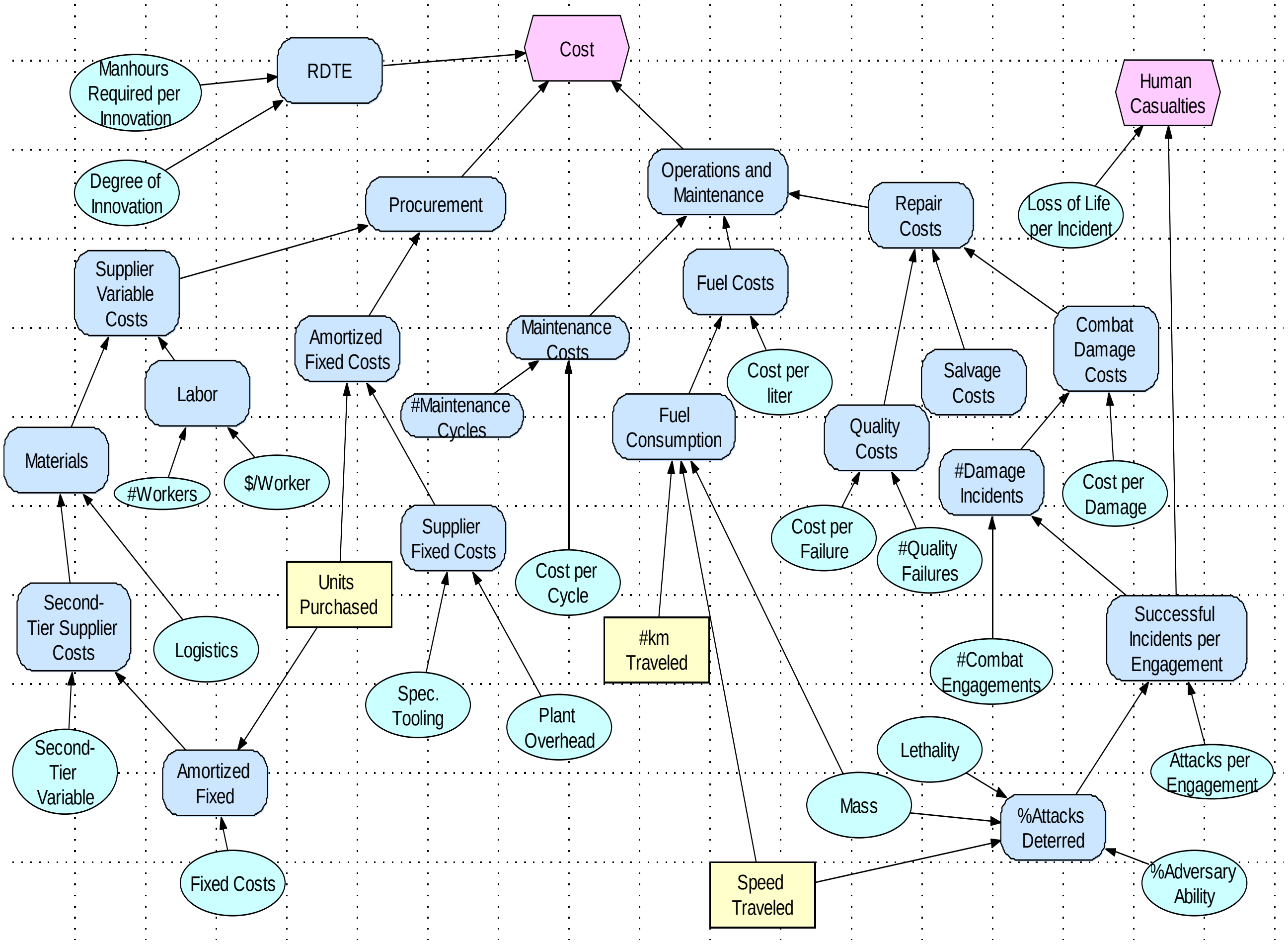


Figure 1: Notional Influence Diagram for the drivers of military costs

avoid too much clutter, we create this complex sub-model as another influence diagram whose output is specialized tool. But only specialized tooling will appear in our diagram. While not used in this diagram, modules are represented as rounded circles with very dark edges. The first page of a model will typically have several modules. Each one of these modules will typically have other modules, and so on. A key criteria in defining a module is that it truly be separable from other aspects of the model. (Obviously other software tools might be useful in highlighting how the influence diagram can be decomposed.)

An influence diagram is not complete until numbers are assigned to all of its terminal nodes. Sometimes this can be done by simply getting information on the terminal nodes. But sometimes we have information on non-terminal nodes as well. For example, we might have some good information about what the military's costs might be based on benchmarking. Or we might have good information about RDTE costs. In this case, we want to choose the values assigned to the terminal nodes so as to:

(1) Minimize the deviation between their values and the values suggested by direct information on that terminal node
(2) Minimize the deviation between the values they implied for upstream nodes (like cost) and the values suggested for cost by direct information on cost.

Thus it is not good procedure to simply focus on gathering information on the terminal nodes. If there is existing information on upstream nodes, then that information should be gathered to yield a model which is as consistent as possible with all available information. This is especially true when the information about terminal nodes is very sketchy.

Design of Experiments

Once the influence diagram is constructed, sensitivity analysis is used to identify the impact of different uncertainties. This sensitivity analysis is best conducted using the design of experiments (DOE). This involves first identifying the decisions which optimize our objective function. Given this decision, we then

(1) Identify the range over which each uncertain node might vary
(2) Estimate the direct effect of each uncertain variable by
    a. fixing all decision variables at the optimal value
    b. setting the value of each variable to its upper bound, computing the cost arising from randomly varying all other nodes between their lower and upper bounds, and then computing the average value of the variable at its upper bound.
    c. setting the value of each variable to its lower bound, computing the cost arising from randomly varying all other nodes between their lower and upper bounds, and then computing the average value of the variable at its lower bound.

The design of experiments also allows for the estimation of interaction effects.

We now identify other decisions that might be superior to the optimal solution for some setting of chance variables. If there are such decisions, then our current optimal decision is not robust to some uncertainties. After identifying these critical uncertainties, we must both

(1) Gather more information on those uncertainties and
(2) Investigate other decisions which might make our original decision more robust to these uncertainties

After we gather this information and identify other decisions, we then redo the design of experiments in order to identify whether the optimal decision is now robust to these key uncertainties.

ARCHITECTURE MANAGEMENT

Overview

Architecture management is especially important in military systems due to their very long life cycles. To accommodate change over these extended periods one must create flexible architectures to reduce the component and architectural drivers of cost. In particular, architecture must balance the benefits of commonality, modularity and flexibility against the benefits of an integral design with less flexibility. While flexibility and modularity can enable quick and easy adaptation to changing requirements, integral designs can provide operational efficiencies and reductions in size, weight, power and initial cost (SWAP-C) burdens due to shared functionality and fewer interfaces. Often the decisions around modular or integral design is not an “either/or” decision as much as it is understanding where within an architecture to implement these tenets.

Fundamentally, Systems Architecture is the mapping of function to form via concept. Alternately, it is “The fundamental organization of a system embodied in its components, their relationships to each other and to the environment, and the principles guiding its design and evolution.” Another definition is, “The arrangement of functional elements into physical modules which become the building blocks for the product of family of products.” [IEEE Std 1471-2000]

It is in the interaction or combination of system elements, or between system modules, where systems often generate their value. Therefore, these interactions should be well understood, defined and managed. Methods which assist in making system architectures more modular often simplify the architecture, improve reuse, and enhance flexibility and adaptability. These characteristics are very important in reducing system life cycle costs in military systems.

This section will outline architecture management tools and methods to help identify opportunities for defining and creating modules and common interfaces by the identification of interactions between system elements and grouping these elements into mutually exclusive or minimally interacting subsets. The approach will also take as inputs the uncertainties identified in the previous section to provide a powerful ability to help optimize system architectures for adaptability while minimizing the impacts of change.

Design Structure Matrix

Over the years, methods to reduce system complexity have taken many forms. The Design Structure Matrix (DSM) is one methodology that has proven very effective in the analysis, management, and integration of complex system architectures. DSM enables the user to model, visualize, and analyze the dependencies among the entities of any system—and derive suggestions for system optimization. DSM provides this understanding in a compact and clear representation. In reference to this paper DSM is especially powerful in the identification of component and architectural drivers of cost.

Design Structure Matrix (DSM), a simple and insightful yet powerful Systems Engineering and Integration (SEI) methodology for managing and developing complex system architectures. DSM has been successfully applied in the automotive, aerospace, construction, microprocessors, electronics and other industries as well as in the U.S. Air Force, U.S. Navy and NASA.

The use of matrices in system modeling, as done with Design Structure Matrix, can be traced back to the 1960s and '70s with Donald Stewart and John Warfield. However, it wasn't until the 1990s that the method received widespread attention. Much of the credit in its current popularity is accredited to MIT's research in the design process modeling arena by Dr. Steven Eppinger.

DSM—also known as the dependency structure matrix, dependency source matrix, and dependency structure method—is a square matrix that shows relationships between elements within a system. Since the behavior and value of many systems is largely determined by interactions between its elements, DSMs have become increasingly useful and important in recent years.

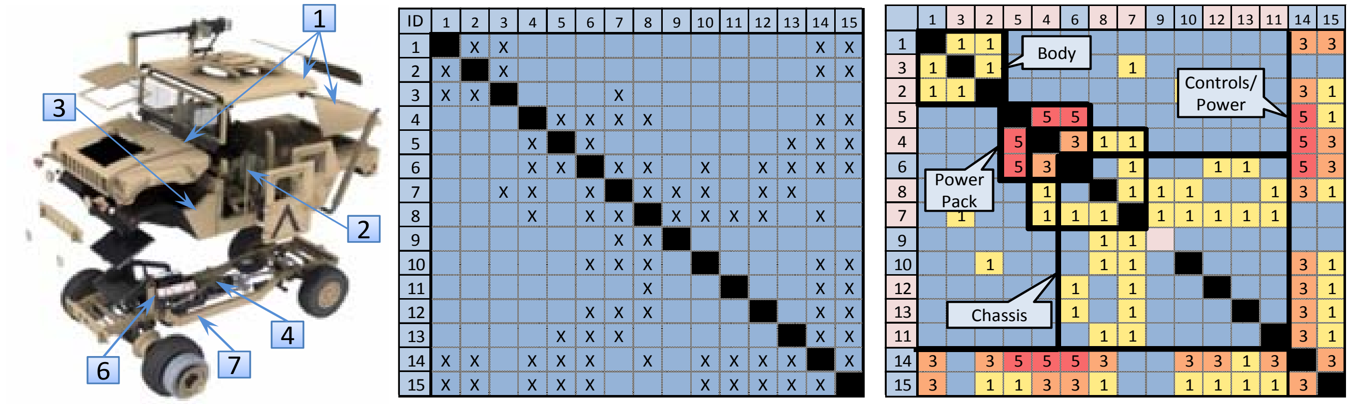
The DSM is related to other square-matrix–based methods, such as: a dependency map, a precedence matrix, a contribution matrix, an adjacency matrix, a reachability matrix, and an N-square diagram, and also related to non-matrix–based methods such as directed graphs, systems of equations, and architecture diagrams and other dependency models.

Relative to other system modeling methods, DSM has two main advantages that differentiate it from the others:

- DSM provides a simple and concise way to represent a complex system.
- DSM is capable of powerful analyses techniques—which will be discussed in subsequent sections of this paper.

The general DSM modeling approach consists of the following steps:

1) Define the system boundary
2) Describe important interfaces
3) Decompose the system into simpler elements
4) Define the characteristics of the elements
5) Characterize the element interactions
6) Analyze the system architecture (structure):
    a) System model behaviors
    b) Potential element arrangements/integrations.



ID	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1		X	X											X	X
2	X		X											X	X
3	X	X					X								
4					X	X	X	X						X	X
5				X		X							X	X	X
6				X	X		X	X		X		X	X	X	X
7			X	X		X		X	X	X		X	X		
8				X		X	X		X	X	X	X		X	
9							X	X							
10						X	X	X						X	X
11								X						X	X
12						X	X	X						X	X
13					X	X	X							X	X
14	X	X		X	X	X		X		X	X	X	X		X
15	X	X		X	X	X				X	X	X	X	X	

	1	3	2	5	4	6	8	7	9	10	12	13	11	14	15
1		1	1											3	3
3	1		1					1							
2	1	1												3	1
5					5	5								5	1
4				5		3	1	1						5	3
6				5	3			1			1	1		5	3
8					1			1	1	1			1	3	1
7		1			1	1	1		1	1	1	1	1		
9							1	1							
10			1				1	1						3	1
12						1		1						3	1
13						1		1						3	1
11							1	1						3	1
14	3		3	5	5	5	3			3	3	1	3		3
15	3		1	1	3	3	1			1	1	1	1	3	

1 Decompose System Elements or Functions

2 Document all interactions in a preliminary Binary DSM (Static Product DSM)

3 Modularize and characterize interactions (Advanced DSM)

Figure 2: Notional Tactical Wheeled Vehicle, Binary DSM and Modularized and Weighted DSM

DSM Application:

To illustrate, consider the basic static DSM example in the Figure 2 above.

(1) The system architecture is decomposed into functions or components (as shown).
(2) A matrix is constructed where a system element is associated with both a row and a column. A mark in a cell of the matrix indicates an interaction between the components for that cell's row and column.
(3) Interactions can then be classified based on integration risk (as shown), functional need, type of interaction (spatial, energy, information, material) etc.
(4) Matrix based analysis techniques then rearrange rows and columns so as to cluster together components that strongly interact with one another while being less connected to other components. This facilitates identifying functional subsystem modules (body, power pack etc.) and distributed subsystems (Controls/Power).

Modularization

Clustering is a powerful technique for manipulating a DSM. The goal is to find subsets of system elements (i.e., clusters or modules) that are mutually exclusive or minimally interacting subsets (i.e., clusters as groups of elements that are interconnected among themselves to an important extent while having few connections to the rest of the system). This process is referred to as *clustering.* In other words, clusters absorb most, if not all, of the interactions internally, and the interactions or links between separate clusters are eliminated or at least minimized. Managing a system's architecture in this way can help to minimize change propagation, enable technology insertion and enable a platform based strategy making systems less sensitive to the uncertainties described in the previous section.

Change Propagation

To accommodate uncertainties engineering changes are often necessary and the implementation of these updates are often suboptimal due to architectural constraints. In such situations changes can balloon in an uncontrolled fashion. This drives the need to understand and model up front how such changes might propagate in order to minimize the future impact. DSM can also aid in identifying how changes propagate through an architecture so second, third and fourth order impacts are discovered. Early discovery of such ”propagation paths” as described by Koh, Caldwell and Clarkson 2009, can have a significant impact on total life cycle cost. This analysis aids engineers and designers in isolating or controlling change propagation due to upgrades. Managing elements classified as change "multipliers" will reduce the cost of future upgrades as well as limit their ability to compromise system performance.

Technology Insertion

Modularization and architecture management using DSM also enables technology infusion and modular system upgrades. The use of a delta DSM (deWeck, 2007) or Multi-Domain Matrix to assess technology integration risks and opportunities within an architecture can quickly be assessed once the Systems and Technology DSMs have been created. Such an effort can also provide metrics on technology invasiveness as outlined by deWeck.

Platform Engineering

One fundamental approach to driving down cost in complex systems is to maximize reuse or commonality. Doing this to the largest extent possible provides economies of scale, reductions in component complexity etc. However, to meet wide varying or potentially highly dynamic user needs systems often require components which are unique to a particular mission need. A platform engineering approach seeks to balance flexibility, unique capability and commonality within systems and it has proven to be effective in delivering cost effective future capabilities under uncertainty.

Some of the fundamental elements of this approach include using the methods outlined above as well as identifying:

- ***Common Elements*** which are uniform across variants in a product family and are less sensitive to unexpected changes over time.
- ***Flexible Elements*** that can be interchanged at a lower cost to accommodate uncertainties
- ***Unique Elements*** which are not easily changed without redesign.
- ***Tunable Design Parameters*** or the design variables used to dial in capability
- ***Interfaces*** or points of connection between entities, made modular and open where appropriate, to enable `plug and play' capability

Within this context a Sensitivity DSM (SDSM) can specify the degree to which systems elements are sensitive to anticipated change or uncertainty. Elements most subject to change are tentatively classified as unique elements while those insensitive to change become platform elements (Kalligeros, K., de Weck, O., de Neufville, 2006). Modeling functional requirements along with design variables in an SDSM at various levels of abstraction provides an effective way to manage system architectures. This method can also be enhanced to identify and prioritize candidate flexible elements based upon cost and/or performance. This could reduce downstream manufacturing complexity for a more affordable design.

Summary

In summary, DSM aids architecture management by providing an effective representation for system components and their relationships. It helps to modularize systems, manage change propagation and technology insertion as well as provide a foundation for platform engineering methods. Using DSM together with the uncertainty management methods outlined in the previous section promotes the development of systems which are more robust to change through the identification and management of cost drivers in complex systems.

DECISION ANALYSIS

Overview

Uncertainty management reduces the uncertainties affecting the architecture while architecture management minimizes the overall impact of uncertainty on the overall system architecture. The management of these two areas is tightly coupled and they must be managed together. While it is clear that the outputs of uncertainty management will affect the decisions made for the system architecture, it is also true that the outputs of architecture management will change the decisions made in how to manage uncertainty. Hence, the two techniques must be used iteratively to explore the trade space for an optimal solution.

Nonetheless there will typically be some remaining uncertainty not cost-effectively addressable through either uncertainty management or architecture management. Hence decision analysis is necessary. The goal of decision analysis

There are two important cases in applying decision analysis:

(1) Uncertainties only appear in the objective function. In this case, we represent our willingness to trade off performance against risk by specifying a utility function over different values of the objective. We then use a decision tree to describe the possible outcomes of a decision. Finally we use the expected utility criteria to identify the `optimal' decision given the uncertainty.

(2) Uncertainties appear in the objective function and constraint. In this case, we redefine the problem. Instead of maximizing the objective function subject to constraints, we maximizing the joint probability of both
   a. satisfying all constraints
   b. having the objective function exceed an uncertain threshold. The uncertain threshold is defined so that the cumulative probability of the uncertainty is, in fact, equal to the previously mentioned utility function.

This second approach is often called reliability-based design optimization in structural design. .This involves specifying an upper bound on the maximum tolerable risk and optimizing an objective function subject to that upper bound. Unfortunately, RBDO can ignore negligibly more costly solutions which lead to dramatically smaller risks This property, which violates commonsense (and even product liability law), is avoided by Bordley and Pollock (2009)'s improved RBDO algorithm.

Target-Oriented Utility

Doing optimization when there is only uncertainty in both the objective function requires the replacement of the objective function by its utility, where the utility function automatically adjusts to allow for the client's attitude toward risk. When there is uncertainty in both the constraints and the objective function, we must replace the objective function by the probability of the objective function exceed some uncertain threshold. In other words, the utility function is replaced by a probability distribution over an uncertain threshold. While it can be shown that these replacement is always mathematically acceptable when there is only one criterion of interest, a natural question is how to extend this result to the case of multiple criteria.

For example, suppose our mission is to satisfy our customer for the foreseeable future. But because of future uncertainties, not only about the environment, but also about what future threats and preferences, we simply do not know what the customer will require in that future. For example, NASA wishes to design a rocket but does not know what the rocket's requirements should be. To solve this problem, we define our utility as one if we achieved our mission by successfully meeting what the customer actually requires at some future time. Otherwise we get utility zero.

For simplicity, first consider the case where there are only two requirements. Suppose the probability of achieving our mission is u{G|1,2} if both requirements are met. Let p{12} be the probability that both requirements are satisfied.

Likewise suppose the probability of achieving the mission is u{G|1} if only the first condition is met. The probability of only the first condition being met is p(1). Similarly the probability of achieving the mission is u{G|2} if only the second requirement is met with p(2) being the probability of only satisfying the second requirement. Finally let u(G|0) be the probability of achieving the mission if none of the conditions are met. The probability of no requirements being satisfied is p(0). Thus we allow for the possibility of failure if we satisfy both requirements as well as the possibility of success if we satisfy none of our requirements. Then our overall chance of achieving our mission is

$$u(G\}=u(G|1,2)\ p(12)+u(G|1)p(1) + u(G|2)p(2) + u(G|0)p(0)\}\ p\_0$$

Let F(12) be the probability that both conditions are met. Let F(1) be the probability that the first condition is met. Let F(2) be the probability that the second condition is met. Letting

$$w(12)\}=u(G|12)-u(G|1)-u(G|2)+u(G|0)$$

$$w(1)=u(G|1)-u(G|0) \text{ and}$$

$$w(2)=u(G|2)-u(G|0) \text{ implies}$$

$$u(G)-u(G|0) = w(12)F(12) + w(1)F(1)+w(2)F(2)$$

Which can be generalized to an arbitrarily large number of requirements.

This corresponds to a multilinear multiattribute utility which reduces to the standard additive utility when only w(1),...w(n) are nonzero. Another important case arises when the mission can only be achieved by achieving all requirements. In this case, we maximize F(1,...n). When there is independence among the requirements, the logarithm of F(1,...n) can be treated as an additive utility function.

CONCLUSIONS

The tools and methods outlined within this paper have been successfully employed for Uncertainty Management Architecture Management and Decision Analysis within the government and industry. This paper illustrates an interactive use of these methods for identifying the component and architectural divers of cost in military systems. This is ultimately achieved by designing more flexible and adaptable systems with architectures which are less sensitive to uncertainties. The approach aids conceptualization and provides teams with an in-depth understanding of the system and its potential emergent behaviors, which fundamentally arise out of the integration of elements within the system. Furthermore, the system models built in using these methods, both from a decision analysis and architecture perspective, will establish a documented knowledge base for making decisions and recording key design rules, which are fundamental needs in effectively evolving platforms with extended life cycles. The resulting analysis provides a baseline for continuous improvement and the reduction of costs across the life cycle.

REFERENCES

[1] Bordley, R. and C. Kirkwood.(2004.) “Multiattribute Preference Analysis with Performance Targets.” Operations Research. 52(6), 823-835.

[2] Bordley, R. and S. Pollock. (2009). “A Decision Analytic Approach to Reliability-Based Design Optimization.” Operations Research. 57(5), 1262-1270.

[3] Kalligeros, K, DeWeck, O., deNeufville, R. and A. Luckins.(2006). ``Platform Identification in DSM.’’ 16th Annual Int’l INCOSE Symposium , Orlando, Florida, 8-4, July.

[4] Morgan, G. and M. Henrion. (1990). Uncertainty. Cambridge Univ.Press, New York.

[5] Suh, E., O. DeWeck, D. Chang (2007). ``Flexible Product Platforms: Framework and Case Study.” Research in Engineering Design. 18(2), 67-89.

[6] Ulrich, K and S. Eppinger. (2000). Product Design and Development. 2nd edition, Irwin McGraw-Hill.

[7] Rinkevich, D. and Samson, F. P., “An Improved Powertrain Attributes Development Process With the Use of Design Structure Matrix”, MIT Master’s Thesis, Cambridge, MA, Feb. 2004

[8] Sharman, D. and Yassine, A., "Characterizing Complex Product Architectures," Systems Engineering, Forthcoming, 2004.

[9] Steward, D.V. “The Design Structure System: A Method for Managing the Design of Complex Systems.” IEEE Transactions on Engineering Management, Vol. 28, pp. 71-74, 1981.