Tradespace and Affordability – Phase 1

A013 - Final Technical Report SERC-2013-TR-039-1

July 9, 2013

Principal Investigator: Dr. Barry Boehm, University of Southern California

Research Team:

Air Force Institute of Technology

Georgia Institute of Technology

Massachusetts Institute of Technology

Naval Postgraduate School

Pennsylvania State University

Stevens Institute of Technology

University of Southern California

University of Virginia

Wayne State University

Contract Number: H98230-08-D-0171

TO 0031, RT 046

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
09 JUL 2013		**00-00-2013 to 00-00-2013**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Tradespace and Affordability - Phase 1	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **University of Southern California,Systems Engineering Research Center,Los Angeles,CA,90089**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**115**	

This material is based upon work supported, in whole or in part, by the U.S. Department of Defense through the Systems Engineering Research Center (SERC) under Contract H98230-08-D-0171. SERC is a federally funded University Affiliated Research Center managed by Stevens Institute of Technology

Motivation and Context

One of the key elements of the SERC's research strategy is transforming the practice of systems engineering – "SE Transformation." The Grand Challenge goal for SE Transformation is to transform the DoD community's current systems engineering and management methods, processes, and tools (MPTs) and practices away from sequential, single stovepipe system, hardware-first, outside-in, document-driven, point-solution, acquisition-oriented approaches; and toward concurrent, portfolio and enterprise-oriented, hardware-software-human engineered, balanced outside-in and inside-out, model-driven, set-based, full life cycle approaches.

These will enable much more rapid, concurrent, flexible, scalable definition and analysis of the increasingly complex, dynamic, multi-stakeholder, cyber-physical-human DoD systems of the future. Four elements of the research strategy for SE Transformation are the following:

1. Make Smart Trades Quickly: Develop MPTs to enable stakeholders to be able to understand and visualize the tradespace and make smart decisions quickly that take into account how the many characteristics and functions of systems impact each other

2. Rapidly Conceive of Systems: Develop MPTs that allow multi-discipline stakeholders to quickly develop alternative system concepts and evaluate them for their effectiveness and practicality

3. Balance Agility, Assurance, and Affordability: Develop SE MPTs that work with high assurance in the face of high uncertainty and rapid change in mission, requirements, technology, and other factors to allow systems to be rapidly and cost-effectively acquired and responsive to both anticipated and unanticipated changes in the field

4. Align with Engineered Resilient Systems: Align research to leverage ERS and contribute to it; e.g., ERS efforts to define new approaches to tradespace.

For strategy 3, "Systems" covers the full range of DoD systems of interest from components such as sensors and effectors to full systems that are part of net-centric systems of systems and enterprises. "Effectiveness" covers the full range of needed system quality attributes or ilities, such as reliability, availability, maintainability, safety, security, performance, usability, scalability, interoperability, speed, versatility, flexibility, and adaptability, along with composite attributes such as resilience, sustainability, and suitability or mission effectiveness. "Cost" covers the full range of needed resources, including present and future dollars, calendar time, critical skills, and critical material resources.

RT-46, Tradespace and Affordability, is a major SERC initative within SE Transformation. It particularly focuses on the tradespace among a system's Ilities, or non-functional requirements. Its project name is ilities Tradespace and Affordability Project (iTAP).
The ilities differ from functional requirements in that they are systemwide properties that specify *how well* the system should perform, as compared to functions that specify *what* the system should perform. Adding a functional requirement to a system's specification tends to have an incremental, additive effect on the system's cost and schedule. Adding an ility requirement to a system's specification tends to have a systemwide, multiplicative effect on the system's cost and schedule. Also, ilities are harder to specify and evaluate, as their values vary with variations in the system's environment and operational scenarios.

Further, the satisfaction of their specifications is much harder to verify than placing an X in a functional traceability matrix, as the verification requires considerable effort in analysis across a range of environments and operational scenarios. As a result, it is not surprising that problems in satisfying ility requirements are the source of many DoD acquisition program cost and schedule overruns. Also, with some exceptions such as pure physical systems and pure software systems, there is little technology in the form of scalable methods, processes, and tools (MPTs) for evaluating the satisfaction of multiple-ility requirements and their associated tradespaces for complex cyber-physical-human systems.

The increasingly critical DoD need for such capabilities has been identified in several recent studies and initiatives such as the National Research Council's "Critical Code" Report (NRC, 2010), the SERC "Systems 2020" Report (SERC, 2010), the "Manual for the Operation of the Joint Capabilities Integration and Development System" (JROC, 2011), and the DoD "Engineered Resilient Systems (ERS) Roadmap" (Holland, 2012). The particular need for Affordability has been emphasized in several USD(AT&L) and DepSecDef "Better Buying Power" memoranda (Carter et al., 2010-2013) and research-need studies such as the AFRL "Technology Horizons" report (Dahm, 2010).

Phase 1 Objectives, Approach, and Results

The major objectives of the initial 5-month Phase 1 activity have been:

- To lay strong foundations for ITAP Phase 2, including knowledge of Department of Defense (DoD) ility priorities; foundations and frameworks for ITAP analysis; extension and tailoring of existing ITAP methods, processes, and tools (MPTs); and exploration of candidate Phase 2 pilot organizations for ITAP MPTs.

- To help develop an Ilities Tradespace and Affordability community of interest via collaborative activities with the DoD Engineering Resilient Systems (ERS) program and counterpart working groups in the International Council for Systems Engineering (INCOSE), the Military Operations Research Society (MORS), and the National Defense industry Association (NDIA).

Four activities were pursued in achieving these objectives:

1. Ility Definitions and Relationships. Phase 1 included a discovery activity to identify and analyze DoD and other ility definitions and relationships, and to propose a draft set of DoD-oriented working definitions and relationships for the project.

2. iTAP Foundations and Frameworks. This effort is helping to build iTAP foundations by elaborating key frameworks (process-based, architecture-based, means-ends based, value-based), anticipating further subsequent elaboration via community efforts. These elaborations would enableDoD projects to better establish an integrated set of iTA capabilities.

3. Ility-Oriented tool demos and extension plans. This effort created initial demonstration capabilities from strong existing ITA analysis toolsets and explored piloting by user organizations in the DoD Services.

4. Program management and community building. Considerable effort is involved in ensuring rapid and effective research results across three activity areas and eight SERC collaborators, as well as coordinating efforts with complementary initiatives in the DoD ERS, INCOSE, MORS, and NDIA communities – but the resulting payoffs will be worth the effort.

The Phase 1 results for activities 1 and 2 are presented in Section 1 on Technical Foundations of ility Tradespace Analysis. They include sets of views relevant to ilities tradespace and affordability analysis that are intended to provide a common framework for reasoning about ilities, similar in intent to the various views provided by SysML for product architectures and DoDAF for operational and architectural views. Section 1.1 summarizes results on ility relationship views, including definitions, stakeholder value-based and change-oriented views, views of ility synergies and conflicts resulting from ility achievement strategies, and a representation scheme and support system for view construction and analysis.

Section 1.2 summarizes results on process-oriented views, including an incremental epoch-era approach to deal with change and uncertainty; differences between point-based, outside-in and set-based, inside-out architecting processes; and evidence-based decision processes. Section 1.3 summarizes results on means-ends views, including hierarchies of means for achieving the ends of affordability and timeliness. Section 1.4 addresses domain-oriented views and their advantages of speed and interoperability when composing system elements within the domain.

Section 1.5 addresses aspects of system interoperability and composability within systems of systems and enterprises, including the challenges of interoperability of multi-domain elements and the tradespace between investments in portfolio or product line assets, and the savings involved in their resulting reusability. And Section 1.6 summarizes results in systems

engineering for affordability in the context of the series of DoD memoranda on Better Buying Power.

Section 2 presents results in adapting and extending the team's previously-developed methods, processes, and tools (MPTs) for tradespace and affordability analysis, and their demonstration at iTAP workshops at INCOSE IW in Jacksonville in January 2013 and at CSER in Atlanta in March, and in two presentation sessions at the Army Engineer Research and Development Center, the lead organization for the DoD Engineered Resilient Systems initiative, in Vicksburg, MS in January and April 2013. These are summarized in Section 2.1, along with the capabilities explored for adaptation and extension with DoD early adopters described in more detail in Sections 2.2 through 2.5.

Section 2.2 discusses iTAP university MPTs in the ground vehicle domain, including the FACT system developed at Georgia Tech, being applied at the US Marine Corps, and prospectively for the CREATE-SHIPS program; and the Wayne State ground vehicle MPTs being developed and applied at the US Army TARDEC. Section 2.3 summarizes iTAP Phase 1 explorations by NPS and Wayne State to apply their capabilities to the CREATE-SHIPS program, now being followed up in Phase 2. Section 2.4 summarizes the MIT incremental epoch-era approach to deal with change and uncertainty MPTs developed for the space domain, and being explored in Phase 2 for application to the Army ERDC logistics domain. And Section 2.5 summarizes an exploratory USC-NPS initiative to create a next-generation full-coverage satellite system cost estimation capability in concert with USAF-SMC, NRO, and the Aerospace Corporation.

Section 3 summarizes the overall plans developed in Phase 1 for the pursuit of Phases 2 and 3 objectives. The 7.5-month Phase 2 will extend and refine the initial foundation capabilities discussed in Section 1, and will conduct exploratory MPT applications and extensions of the capabilities discussed in Section 2. Phase 3 will extend these to more general and robust capabilities for further and more general pilot application, evaluation, refinement, and extensions.

References

(Carter et al., 2010-2013) Carter, A., et al., Better Buying Power Memoranda, http://bbp.dau.mil/references.html.

Dahm, 2010) Dahm, W., Technology Horizons, AF/ST-TR 10-01-PR, 15 May 2010.

(Holland, 2012) Holland, J. "ERS Overview," Proceedings, NDIA SE Conference, October 2012.

(JROC, 2011) Joint Requirements oversight Council, "Manual for the Operation of the Joint Capabilities Integration and Development System," Updated 31 January 2011.

(NRC, 2010) Scherlis, W. et al., Critical Code: Software Producibility for Defense NAS Press, 2011.

(SERC, 2010) B. Boehm, J. Bayuk, A. Desmukh, R. Graybill, J. Lane, A. Levin, A. Madni, M. McGrath, A. Pyster, S. Tarchalski, R. Turner, and J. Wade, Systems 2020 Strategic Initiative, Final Technical Report SERC-2010-TR-009, August 29, 2010.

RESEARCH TEAM

Air Force Institute of Technology	Dr. David Jacques Dr. John Columbi Dr. Erin Ryan
Georgia Institute of Technology	Dr. Tommer Ender Mr. Michael Curry
Massachusetts Institute of Technology	Dr. Donna Rhodes Dr. Adam Ross, Co-PI
Naval Postgraduate School	Dr. Ray Madachy
Stevens Institute of Technology	Dr. Roshanak Nilchiani Dr. Stan Rifkin
University of Southern California	Dr. Barry Boehm, PI Dr. JoAnn Lane, Co-PI Mr. Nupul Kukreja Mr. Daniel Link
University of Virginia	Dr. Kevin Sullivan, Co-PI Ms. Xi Wang
Wayne State University	Dr. Walter Bryzik Dr. Gary Witus

TABLE OF CONTENTS

FIGURES AND TABLES

1.1 ILITY RELATIONSHIP VIEWS

1.1.1 VALUE-ORIENTED ILITY HIERARCHY (USC)

Traditionally, ility requirements have been specified as single values, such as for Reliability, "The system shall have a Mean Time Between Failure (MTBF) of 10,000 hours," or equivalently, "shall have a probability of non-failure of 0.9999 per hour." This practice has been the source of numerous problems, for the following three reasons:

1. The requirement above says nothing about the amount of stress that the system will undergo with variations in the system's environment and operational scenarios. Extremes in environmental conditions and high-risk operational scenarios will make the MTBF requirement highly unlikely or extremely expensive to be satisfied.
2. Different operational stakeholders will be relying on the system in different ways. Some will only be concerned with liveness. Others will be highly dissatisfied with a system that has an MTBF for liveness of 10,000 hours, but produces garbled or dropped messages that they are relying on more than once a day.
3. In a world in which tradeoffs among various ilities are becoming increasingly important, specifying a single numerical value leaves no way to define a tradespace that systems engineers can work within to better produce acceptable ility levels among several stakeholder-desired ilities. In preference, having a requirement that specifies a liveness MTBF desired level of 10,000 hours and an acceptable level of 5,000 hours across a weighted set of operational environments and scenarios; and a similar correct-message-delivery MTBF desired level of 2 months and an acceptable level of 1 month; will create a tradespace in which a liveness MTBF of 8,000 hours and a correct-message-delivery MTBF of 1.5 months will be a mutually acceptable solution for these mission-critical stakeholders.

Clearly, there will be a nontrivial amount of effort involved in creating such specifications. But just as clearly, this is the kind of thought and effort that systems engineers should be undertaking (and being budgeted for) in defining what a future mission-critical system should be able to do, and how well it should be able to do it.

Overview of ilities Defined

After analyzing the current primary definitions of ilities and their relationships, such as ISO/IEC 9126 and the 25000 series, the Joint Capabilities Integration and Development System Combat Commanders' ility priorities, and the Air Force Risk Identification: Integration and Ilities Guidebook, we found that none had the necessary coverage of DoD needs, or had a fully satisfactory rationale for their hierarchical decomposition of ilities. The most satisfactory hierarchical decomposition of ilities that we analyzed was one organized about the various sources of value that DoD stakeholders have for the ilities. These sources were Mission Effectiveness and Resource Utilization, which combine to define current cost-effectiveness; Protection and Robustness, which combine to ensure that the cost-effectiveness remains capable across various natural or adversary disruptions; and Flexibility and Composabilty, which combine to ensure that a system's current cost-effectiveness can be maintained or increased as the system's environment and operational scenarios undergo change.

Table 1 provides a top-level definition of the various ilities. For each ility in the left column, the right column summarizes its effect on the system and its stakeholders, subject to variations in the system's environment, workload level, and primary operational scenarios. An effort has been made to define hierarchies of ilities that are mutually exclusive and exhaustive, although neither of these attributes are perfectly achievable for complex systems and for ilities such as Mission Effectiveness. Some key ilities are combinations of the categories below, and are summarized at the bottom of the table. For example, Resilience is defined as the union of Protection, Robustness, and Flexibility. This is followed by a next level of detail of the definitions. Lower hierarchical levels become much more complicated by many-to-many relationships. For example, Testability supports almost all of the higher-level ilities.

Table 1. Top-Level System Ilities Definitions

Ility	**Effect on DoD Operational System**
Mission Effectiveness	Stakeholders-satisfactory balance of Speed, Delivery Capability, Accuracy, Usability, Scalability, and Versatility
Speed	Distance or work accomplished per unit of time
Delivery Capability	Amount of needed payload weight, capacity, energy, bandwidth, throughput, data storage, etc. provided
Accuracy	Closeness to target
Usability	Ease of learning, ease of use, difficulty of misuse
Scalability	Sustainability of system capability across a range of system or environmental scales
Versatility	Range of functions provided
Resource Utilization	Ability to deliver other ilities within constraints on limited resources
Cost	Amount of funding to complete delivery
Duration	Amount of calendar time to complete delivery
Key Personnel	Shortfalls in number of personnel with needed skills

Other scarce resources	Shortfalls in amount of needed Delivery Capability provided
Robustness	Ability of the system to continue to deliver stakeholder-desired capabilities
Reliability	Probability that the system will continue to deliver stakeholder-desired capabilities
Availability	Fraction of the time that the system will deliver stakeholder-desired capabilities
Maintainability	Expected amount of time required to restore stakeholder-desired capabilities
Survivability	Ability of the system to continue to deliver partial stakeholder-desired capabilities
Flexibility	Ability of the system to be rapidly and cost-effectively changed
Modifiability	Flexibility via external reconfiguration
Tailorability	Flexibility via prespecified parameter setting, configuration directives, or services interfaces
Adaptability	Flexibility via internal reconfiguration
Composability	Ability of the system to be rapidly and cost-effectively composed with other systems
Interoperability	Composability via continuing negotiation and evolution of interfaces
Openness	Composability via open standards compliance
Service-Orientation	Composability via published-service interfaces and assumptions
Composite ilities	
Comprehensiveness	All of the above
Resilience	Protection, Robustness, Flexibility
Dependability	Mission Effectiveness, Protection, Robustness
Affordability	Mission Effectiveness, Resource Utilization

More Detailed ility Definitions

Mission Effectiveness involves a stakeholder-satisfactory balance of the component ilities of Speed, Delivery Capability, Accuracy, Usability, Scalability, and Versatility across a representative range of environments, operating scenarios, and system characteristics. The best balance of these will vary by mission scenario and by the value propositions of the system's success-critical stakeholders. Most systems will need to operate across a range of operational scenarios; the best one can do is to evaluate system alternatives via a scenario-weighted average of values or to decide that multiple system versions are preferable to a one-size-fits-all system.

Speed involves how rapidly and completely the system can deliver its needed capability. As examples, a mission's outcome may be improved by the speed of delivery of facilities, combat platforms, weapons, support materiel, personnel, or information. As above, and also applicable to the ilities below, Speed is to be evaluated with respect to the mission-critical stakeholders' desired and acceptable levels across a weighted set of representative operational environments and scenarios. The level of detail of the evaluations should be risk-driven: if there is clear evidence that previous systems and/or commercial technology has been able to meet the stakeholders' acceptable speed levels across a representative set of scenarios without adverse side effects on other ilities, just citing the evidence is sufficient.

Delivery Capability involves how much of a needed resource the system can provide, and for how long and how far. The greater the range, weight, capacity, or levels of other needed resources that the system can provide, the more likely the mission outcome will be improved. Again and for the rest of the criteria, the level of detail of the evaluations should be risk-driven.

Accuracy involves how close the system comes to locating, tracking, or engaging its target. Again, this will vary by scenario and the nature of the environment. The metric for accuracy may be absolute distance, acceptable distance, or various probabilistic measures such as confidence ellipses, or probability of being below a desired accuracy level. For moving targets, target position and velocity should be time-stamped.

Usability involves how easy it is for a system's designated users to learn how to use and to use the system, along with how difficult it is for them to misuse it. Again, this will vary by the range of designated users, including substitutes, and by the environment and operational scenarios of its use, especially including off-nominal scenarios such as recovery from accidental misuse. Evaluation must be done by actual system users, along with test engineers. Metrics should include time to learn and degree of successful performance under various representative conditions of environment or users.

Scalability involves the system's ability to provide stakeholders acceptable levels of its other –ilities as the system's size, complexity, or workload increase or as the speed, capacity, battery power, or display size decreases. Other quantities may be applicable, such as the number of nodes in a scalable-up mobile network or the limited size of a scalable-down mobile platform.

Versatility involves the range of capabilities provided by a system as it is currently configured. A good example is the number and types of blades provided by a Swiss Army knife, but (as of today) the blades cannot be reprogrammable to perform other functions, or generally to be concurrently applied.

The rest of the definitions will be similarly elaborated in Phase 2.

1.1.2 CHANGE-ORIENTED VIEW (MIT)

One of the fundamental challenges for developing a clearer understanding of the semantics of "ilities" is the current ambiguity in these terms. Many of these terms are used colloquially and therefore inherit informal meaning. Additionally, the terms, as currently used, display polysemy and synonymy. Polysemy is "the property of [a term] having multiple meanings that are semantically related" (Akmajian et al. 2001, p. 585). An example of polysemy is two different, but related meanings for flexibility: "able to be changed" and "able to satisfy multiple needs." (For a broad discussion of multiple related meanings for flexibility see Saleh, Mark, and Jordan (2009).) In contrast to polysemy, synonymy is "the property of multiple terms having similar meaning." An example of synonymy is the interchangeable use of flexibility (able to be changed) and changeability (able to be changed or change itself).

One of the reasons for this ambiguity in the technical usage of ilities is that typically ilities are mostly considered one at a time in the literature. As an example, consider flexibility in Saleh, Mark, and Jordan (2009), Nilchiani (2005), and de Neufville and Scholtes (2011). Some work has been done on sets of ilities, such as for changeability in Fricke and Schulz (2005), and Ross, Rhodes, and Hastings (2008).

One of the key papers asserting a relationship amongst a set of ilities is the one by Fricke and Schulz (2005) that uses the concept of "changeability" as a higher order ility that encompasses four key ilities: adaptability, robustness, flexibility, and agility, illustrated in Figure 1.

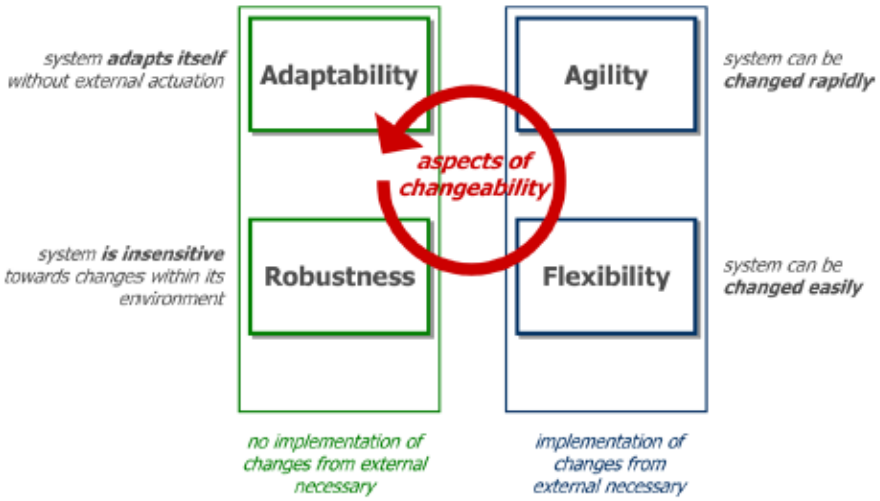


Figure 1. Changeability as four ilities (figure from Fricke and Schulz 2005)

In this work, a number of other ilities are mentioned in the context of "architecture principles" for achieving the changeability-related ilities. These include: simplicity, independence, modularity, integrability, autonomy, scalability, non-hierarchy, decentralization, and redundancy. Implicit in the paper is evolvability. The prescriptive nature of the framework for

relating the ilities into change-type and architecture-type is based on the authors' research and experiences in German product development (e.g. BMW).

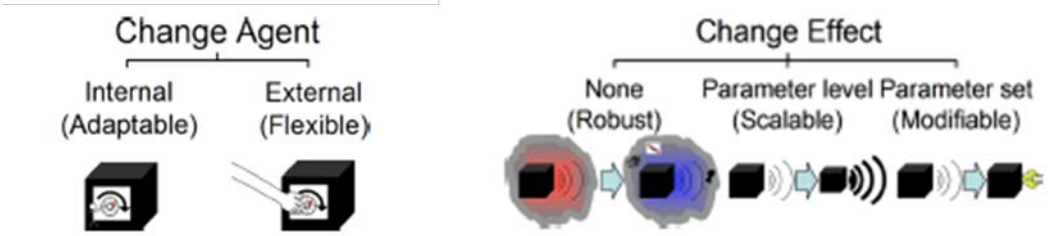


Figure 2. Change agent and effect dichotomy for changeability (figure from Ross, Rhodes and Hastings 2008)

Another changeability-related work is that of Ross (2006) and Ross, Rhodes, and Hastings (2008), which similar to Fricke and Schulz, asserts "changeability" as an overarching ility, with five underlying related ilities along two relationship dimensions: adaptability, flexibility (change agent), and robustness, scalability, and modifiability (change effect) (Figure 2). The concepts of change agent, change effect, and change mechanism are introduced as a means to generalize the concept of changeability and to provide a simple basis for deriving the other five ilities (i.e. adaptable scalability is an internal change agent instigating a change in the level of a system parameter). The "adaptable" vs. "flexible" ility label is dependent on whether the change agent is internal or external to the system boundary. This work introduces the concept that changeability is relative to one or more parameters of a system. In this way a system could display all five ilities. Additionally, this work introduced the concept of a "verifiable changeability statement" as a first step towards concretizing the concepts of these ilities into actionable project engineering and requirements (Figure 3).

The system shall be	_____	_____	in	_____	for less than	_____.
	(change agent type)	(change effects)		(system parameter)		(resources)
	flexibly or adaptably	scalable, modifiable		with range		

Figure 3. Template for verifiable changeability requirement (from Ross, Rhodes, and Hastings 2008)

Recently, an initial exploratory study sought to uncover potential means-ends hierarchical relationships amongst a set of ilities (de Weck, Ross, and Rhodes 2012). In this study, four groups of graduate students constructed means-ends hierarchies from a given set of ilities. Each of the four groups independently derived distinct "hierarchies." In addition to connecting ilities with "means-ends" links, each of the four groups independently proposed a second grouping criterion called "level" or "depth" and structured their group hierarchies to display this quality. Figure 4 below illustrates the aggregate of the four hierarchies, with solid lines indicating 3 or 4 groups in agreement of means-ends relationship between two ilities, and dashed lines indicating 2 groups in agreement. The vertical placement of each ility corresponds to the median level assigned to that ility across the four groups. The lack of consensus and emergent "depth" criterion suggests that more than "means-ends" relationships exist amongst the ilities.

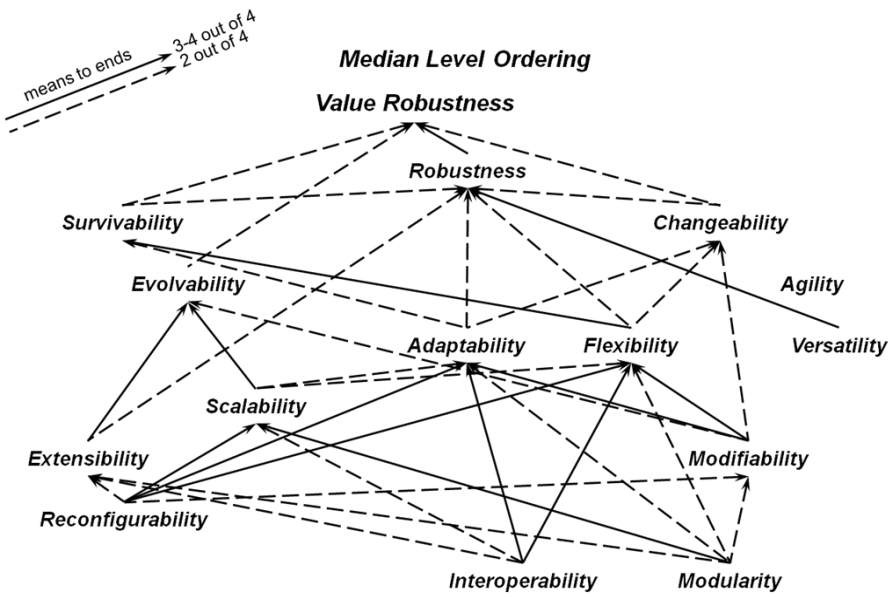


Figure 4. Median level ordering means-ends hierarchy from study (de Weck, Ross, Rhodes 2012)

Feedback from the study indicated that the "bottom" ilities contained some different "sense" than the higher ilities. In particular, modularity and interoperability were viewed as a different perspective on a system lifecycle property than the others and implied particular architectural choices. This insight corresponds to the "architecture principle" concept asserted by Fricke and Schulz (2005) and implies that these ilities may belong to a different semantic field than the others.

Building upon the insights from the various approaches for describing ilities above, what follows describes an initial approach for creating a prescriptive semantic basis for consistently representing ilities within a particular semantic field. At this time, the semantic basis, made up of fourteen categories, is believed to span the *change-type* ility semantic field and excludes the *architecture-type* semantic field that includes "bottom" ilities (de Weck, Ross, and Rhodes 2012) and "architecture principles" (Fricke and Schulz 2005) described above.

Beginning with the change agent and change effect as two categories for defining a change and the resulting applicable ilities, a larger set of categories are proposed for defining a larger set of possible changes for a system. The fourteen categories, which together form the semantic basis, are intended to collectively define a change in a system, thereby creating a consistent basis for specifying change-type ilities in formal statements. A system can be verified to display the quality described in the statement and therefore be traceable to a desired higher order system property. (An earlier version of this basis is described in Beesemyer 2012).

The fourteen categories are: cause, context, phase, agent, impetus, impetus nature, impetus parameter, impetus destination state size, impetus aspect, outcome effect, outcome parameter, outcome destination state size, outcome aspect, level of abstraction, and value

qualities of the change. Unique choices for each of these categories, when applied to a particular system parameter will formulate the change-type ility statement. The fourteen categories are illustrated in Figure 5.

The semantic basis aims to capture the essential differences among change-type ilities through specification of the following general change statement with regard to a particular system parameter:

In response to "cause" in "context", desire "agent" to make some "impetus parameter change" in "system" resulting in "outcome parameter change" that is "valuable."

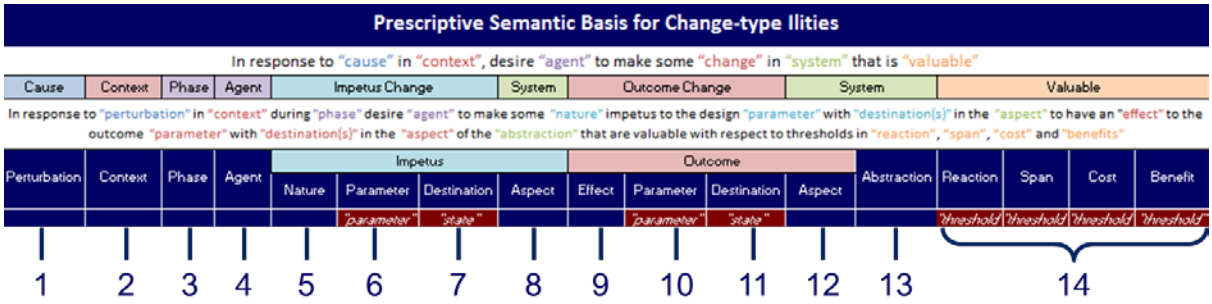


Figure 5. Change-type prescriptive semantic basis in 14 categories. (Ross, Beesemyer, Rhodes 2011)

Application of the semantic basis begins with a user generating a change statement. The change statement is refined and assigned categorical choices within the basis, with the intention that the applicable ilities will emerge from the specified change statement. In this way, a user does not need to use a particular ility label a priori, thereby avoiding the semantic ambiguity in the terms. If the basis accurately and completely describes the underlying categories for change-type changes, then a user should be able to describe any change-type ility through the basis.

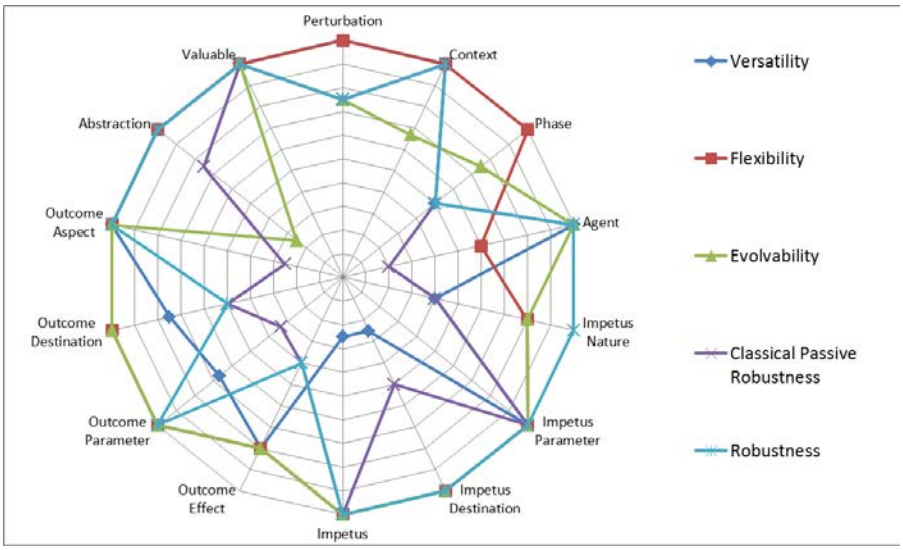
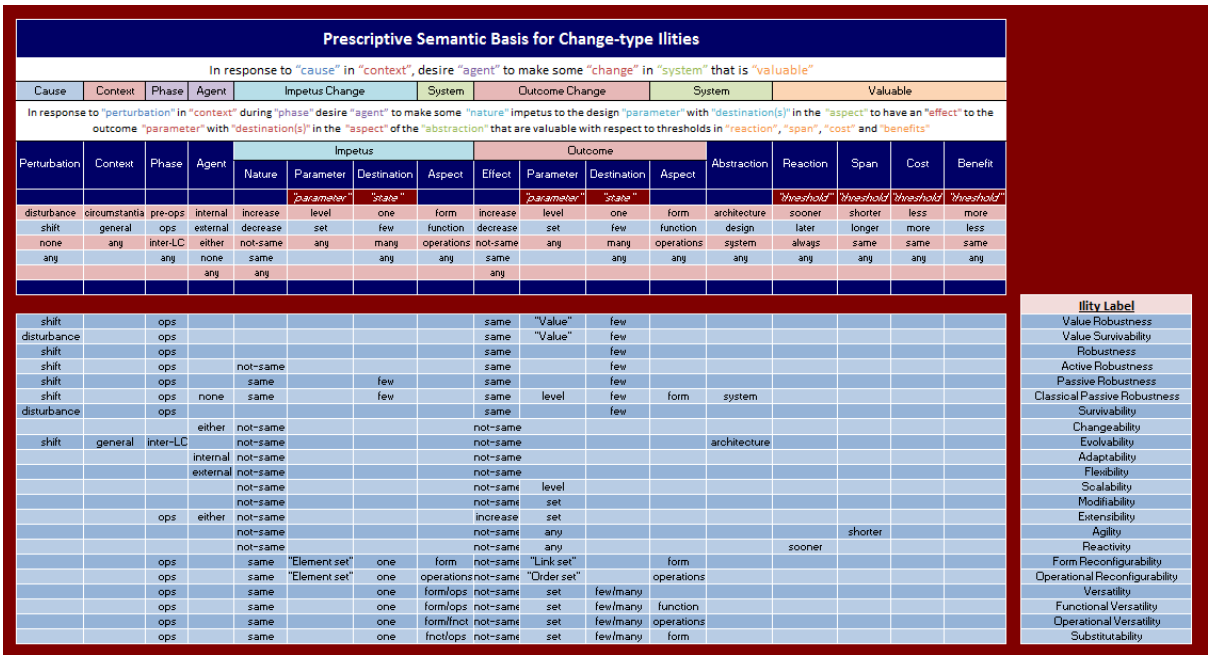


Figure 6. A radar plot depiction of change-type ilities in a 14 category basis

These fourteen categories can be visualized using a radar plot, with different ilities tracing a different shape (Figure 6).

In order to validate the proposed use of the basis, two activities need to be accomplished: refinement of the basis itself, both in terms of complete categories and in terms of choices within those categories. Additionally, if the use of particular ility terms is ultimately desired, a mapping between patterns in the basis and ility terms needs to be generated. If the latter is accomplished, then any usage of the basis for specifying particular change statements will result in consistently derived ility term labels. It is hypothesized that particular change-type ilities will correspond to particular choice(s) in this basis. In this way, a less ambiguous mechanism for specifying ilities can be created.

An example of using the semantic basis for mapping ility terms is illustrated in Figure 7. The results in the figure are presently a work in progress and are meant for illustration purposes only.



Prescriptive Semantic Basis for Change-type Ilities

In response to "cause" in "context", desire "agent" to make some "change" in "system" that is "valuable"

Cause	Context	Phase	Agent	Impetus Change	System	Outcome Change	System	Valuable

In response to "perturbation" in "context" during "phase" desire "agent" to make some "nature" impetus to the design "parameter" with "destination(s)" in the "aspect" to have an "effect" to the outcome "parameter" with "destination(s)" in the "aspect" of the "abstraction" that are valuable with respect to thresholds in "reaction", "span", "cost" and "benefits"

				Impetus				Outcome								
Perturbation	Context	Phase	Agent	Nature	Parameter	Destination	Aspect	Effect	Parameter	Destination	Aspect	Abstraction	Reaction	Span	Cost	Benefit
					"parameter"	"state"			"parameter"	"state"			"threshold"	"threshold"	"threshold"	"threshold"
disturbance	circumstantia	pre-ops	internal	increase	level	one	form	increase	level	one	form	architecture	sooner	shorter	less	more
shift	general	ops	external	decrease	set	few	function	decrease	set	few	function	design	later	longer	more	less
none	any	inter-LC	either	not-same	any	many	operations	not-same	any	many	operations	system	always	same	same	same
any		any	none	same		any	any	same		any	any	any	any	any	any	any
			any	any				any								

shift		ops						same	"Value"	few						
disturbance		ops						same	"Value"	few						
shift		ops						same		few						
shift		ops		not-same				same		few						
shift		ops		same		few		same		few						
shift		ops	none	same		few		same	level	few	form	system				
disturbance		ops						same		few						
			either	not-same				not-same								
shift	general	inter-LC		not-same				not-same				architecture				
			internal	not-same				not-same								
			external	not-same				not-same								
				not-same				not-same	level							
				not-same				not-same	set							
		ops	either	not-same				increase	set							
				not-same				not-same	any					shorter		
				not-same				not-same	any				sooner			
		ops		same	"Element set"	one	form	not-same	"Link set"		form					
		ops		same	"Element set"	one	operations	not-same	"Order set"		operations					
		ops		same		one	form/ops	not-same	set	few/many						
		ops		same		one	form/ops	not-same	set	few/many	function					
		ops		same		one	form/fnct	not-same	set	few/many	operations					
		ops		same		one	fnct/ops	not-same	set	few/many	form					

Ility Label
Value Robustness
Value Survivability
Robustness
Active Robustness
Passive Robustness
Classical Passive Robustness
Survivability
Changeability
Evolvability
Adaptability
Flexibility
Scalability
Modifiability
Extensibility
Agility
Reactivity
Form Reconfigurability
Operational Reconfigurability
Versatility
Functional Versatility
Operational Versatility
Substitutability

Figure 7. Using the semantic basis consistently identify ility term labels (Ross, Beesemyer, Rhodes 2011)

It is well-recognized in systems engineering that ambiguity in requirements contributes to program failures (NDIA 2010). Increasingly, capability and requirements documents include ilities statements. In our empirical studies we have found that such ilities requirements (e.g., The system shall be flexible and robust") are often dismissed as it is unclear what is meant and not possible to verify these. Further research specifying change-type ilities, which results in verifiable and unambiguous requirements statements will make a significant impact for intentionally and consistently designing in change-type ilities into systems.

References

Akmajian, A., Demers, R.A., Farmer, A.K., and Harnish, R.M. (2001), *Linguistics*. MIT Press: Cambridge, MA.

Beesemyer, J.C. (2012), *Empirically Characterizing Evolvability and Changeability in Engineering Systems*, Master of Science Thesis, Aeronautics and Astronautics, MIT, June 2012.

de Neufville, R. and Scholtes, S. (2011), *Flexibility in Engineering Design*, MIT Press: Cambridge MA.

de Weck, O.L., Ross, A.M., and Rhodes, D.H. (2012), "Investigating Relationships and Semantic Sets amongst System Lifecycle Properties (Ilities)", *3rd International Engineering Systems Symposium*, CESUN 2012, TU Delft, 18-20 June 2012.

Fricke, E. and Schulz, A.P. (2005), "Design for Changeability (DfC): Principles to Enable Changes in Systems Throughout their Entire Lifecycle," *Systems Engineering*, Vol. 8, No. 4, pp. 342-359.

NDIA, Top Systems Engineering Issues in the Defense Industry, Systems Engineering Division Task Group, September 2010.

Ross, A.M. (2006), “Managing Unarticulated Value: Changeability in Multi-Attribute Tradespace Exploration,” *PhD in Engineering Systems*, MIT, Cambridge, MA.

Ross, A.M., Beesemyer, J.C., and Rhodes, D.H., (2011) "A Prescriptive Semantic Basis for System Lifecycle Properties", SEAri Working Paper WP-2011-2-1, MIT, Cambridge, MA, http://seari.mit.edu/papers.php.

Ross, A.M., Rhodes, D.H., and Hastings, D.E. (2008), “Defining Changeability: Reconciling Flexibility, Adaptability, Scalability, Modifiability, and Robustness for Maintaining Lifecycle Value,” *Systems Engineering*, Vol. 11, No. 3, pp. 246-262.

Saleh, J. H., Mark, G.T., Jordan, N.C. (2009), "Flexibility: a multi-disciplinary literature review and a research agenda for designing flexible engineering systems," *Journal of Engineering Design*, Vol. 20, No. 3. pp. 307-323.

1.1.3 Strategy-Based ility Synergies and Conflicts (USC)

A common project practice for addressing a high-priority ility such as Security is to set up an Integrated Process Team (IPT) to ensure that the system is highly secure. Often, the IPT will be so focused on Security that it will propose strategies that have adverse effects on other ilities. As some examples from project practice, a Security IPT proposed a single-agent key distribution system to minimize probability of compromise, only to have a Reliability engineer identify this as a system-level single point of failure. Another IPT proposed numerous protection layers that would have consumed 80% of the processing Speed capability. On the other hand, Security emphases on integrity will generally enhance aspects of Reliability, and Security defenses against denial-of-service attacks will generally enhance Speed.

In general, this implies that individual ility IPTs should be completed by a ilities IPT that addresses the synergies and conflicts implied by the individual IPT strategies. However, many of the cross-ility synergies and conflicts are not well understood, and this research area is developing a first cut at identifying them. Some initial examples are provided below; more will be provided in Phase 2.

Flexibility Strategy Synergies and Conflicts

Flexibility Arch. Strategy	Synergies	Conflicts
High module cohesion; Low module coupling	Interoperability Reliability	High Speed via Tight coupling
Service-oriented architecture	Composability, Usability, Interoperability, Testability	High Speed via Tight coupling

Autonomous adaptive systems	Affordability via task automation; , Interoperability, Speed	Excess autonomy reduces human Controllability
Modularization around sources of change	Interoperability, Usability, Reliability, Availability	Extra Duration on critical path of rapid fielding projects
Multi-layered architecture, open standards, plug-ins	Reliability, Availability, Interoperability	Lower Speed due to layer traversal overhead
Many built-in options, entry points	Functionality, Accessibility	Reduced Usability via options proliferation; harder to Secure
User programmability	Usability, Mission Effectiveness	Full programmability causes Reliability, Safety, Security risks
Spare/expandable capacity	Performance, Reliability	Added Cost
Product line architecture, reusable components	Cost, Duration, Reliability	Some loss of Speed vs. optimized stovepipes

Interoperability Strategy Synergies and Conflicts

Interoperability Arch. Strategy	Synergies	Conflicts
High module cohesion; Low module coupling	Flexibility Reliability	High Speed via Tight coupling
Service-oriented architecture	Composability, Usability, Flexibility, Testability	High Speed via Tight coupling
Autonomous adaptive systems	Affordability via task automation; Flexibility, Response time	Excess autonomy reduces human Controllability
Modularization around sources of change	Flexibility, Usability, Reliability, Availability	Extra Duration on critical path of rapid fielding projects
Multi-layered architecture, open standards, plug-ins	Flexibility, Reliability, Availability	Lower Speed due to layer traversal overhead
Wrappers, mediators, distributors, procedure calls	Composability, Reliability	Lower Speed due to interface overhead
Domain interface knowledge utilization	Usability, Mission Effectiveness	Cross-domain Interoperability
Product line architecture, reusable components	Cost, Duration, Reliability	Cross-product line Interoperability
Asset interface and assumption metadata analysis	Composability, Reliability, Rework cost savings	Added Cost of metadata creation, update

Reliability Strategy Synergies and Conflicts

Reliability Arch. Strategy	Synergies	Conflicts
General	Availability, Safety, Security, Privacy	Cost, Duration
Input checking	Usability, Interoperability	Speed on large inputs
Autonomous adaptive systems: Self-test, trend analysis	Affordability via task automation; Response time, Accuracy	Speed; May reduce human Controllability
Data redundancy		Maintainability, Security
Hardware redundancy		Size, Weight, Power, Maintainability
N-version components	.	Speed, Maintainanbility. Not a guarantee of Reliability
Limited component variability	Usability	Versatility
Formal specification and verification	Interoperability	Cost, Usability, Scalability, Maintainability
Recovery blocks		Cost, Speed

1.1.4 VIEW RELATIONSHIP REPRESENTATION (UVA)

Success in the development and operation of today's complex systems depends on our ability to manage comprehensively and effectively (to select, specify, achieve, verify, and evolve) non-functional system properties (or ilities), including affordability, dependability, evolvability,and usability. While we do have limited knowledge, methods and tools to manage certain isolated ilities, such as reliability, we lack the capability to manage others, and we clearly lack the scientific foundations and engineering know-how adequate to support comprehensive treatment of ilities: integrating and making tradeoffs among all key ilities across the whole system life-cycle. Major projects thus fail at an alarming rate, producing massive delays and cost overruns, outright cancellations, and operational systems that are dangerously inadequate.

The root problem is found in (1) an under-developed science of non-functional properties, (2) inadequate engineering methods and tools for managing them, and (3) poor dissemination and application of the knowledge we do have. Symptoms include a shocking lack of agreement on unambiguous and well validated definitions, models, and measures of ilities, relationships among them, threats to their realization, and mechanisms for prescribing and meeting comprehensive ility requirements with assurance.

What is needed is a comprehensive science and engineering approach to managing ilities, including definitions, notations, models, measurements, tradeoffs, and means of gaining assurance across the system lifecycle. Such a framework would enable designers and decision-makers to express and design against comprehensive ility specifications, to include acceptable tradeoffs. An enormous amount of work has been done on system modeling and on individual ilities such as dependability. Yet we are still far from having the knowledge and tools needed for *comprehensive* treatment of ilities and tradeoffs. Popular system modeling notations, for example, do not provide capabilities to express the *full* range of evolvability, safety, security, performance, cyber-physical functionality, and other properties of modern systems in a manner that can support a disciplined and comprehensive approach to managing ilities and tradeoffs across the system lifecycle.

The University of Virginia team, which also leads the iTAP sub-project on scientific foundations, is working with the rest of the iTAP team to develop such a framework. The long-term aim is to develop a scientific and engineering framework that includes explicit treatment of at least the following elements:

- Precise, validated, accepted definitions of relevant ilities, understood as observable properties of given *super-systems*. By this term we mean not only end products (e.g., vehicles), but also the meta-systems that develop them, the range of environments in which systems and meta-systems might operate and evolve, and the evolutionary and tradeoff spaces that these systems and meta-systems inhabit. The design and analysis of super-systems is essential to disciplined engineering of ilities and tradeofs. For example, the specification of system (e.g., vehicle) evolvability has to account for possible future operating environments (outside of the system), fitness functions that incorporate end-user utility, including affordability (a meta-system issue), and tradeoffs against other properties (such as total cost of ownership).
- Formal languages (including data types) and related functions for modeling systems, meta-systems, environments, and trade-spaces; for prescribing and predicting ility properties; for formulating propositions about system ilities; and for organizing evidence in support of such propositions.
- Taxonomies of threats and risks to the achievement of ilities and strategies for achieving ilities in the face of such threats and for mitigating such risks, including characterizations of the impacts of given strategies across the range of relevant system ilities (e.g., how a given strategy for improving security might impact usability).
- Software tools for comprehensive ility management across the system lifecycle.
- Engineering methods for using applying scientific and engineering knowledge and tools in practice
- The specialization and extension of definitions, models, languages, methods and tools to particular domains and environments
- Mechanisms for dissemination and adoption of comprehensive ility management for complex systems and meta-systems.

In response to this need, the University of Virginia is conducting research to strengthen the science of comprehensive for ility and tradespace management. Phase I of the iTAP project supported concept exploration, project planning, and demonstration prototype development. The University of Virginia undertook the following initial efforts.

Docility for View Relationships.

There is already a plethora of inconsistent and often inadequate definitions of ilities and related concepts. A goal of this project is to provide an online resource that documents important existing definitions in order to foster awareness of, improvement in, and eventual convergence on precise and well validated views and definitions of ilities, threats, strategies, impacts of strategies on ilities, and key tradeoffs among ilities. Our approach is to catalyze such convergence and adoption with an enterprise-quality, web-based tool for documenting, evolving and disseminating definitions of and relational propositions about ilities, strategies, and key relationships among these elements. The tool supports a growing database, populated by researchers in the iTAP project, and eventually available for validation and use by practitioners. In Phase I, UVa designed, produced, and deployed a solid, working, initial version of such a tool.

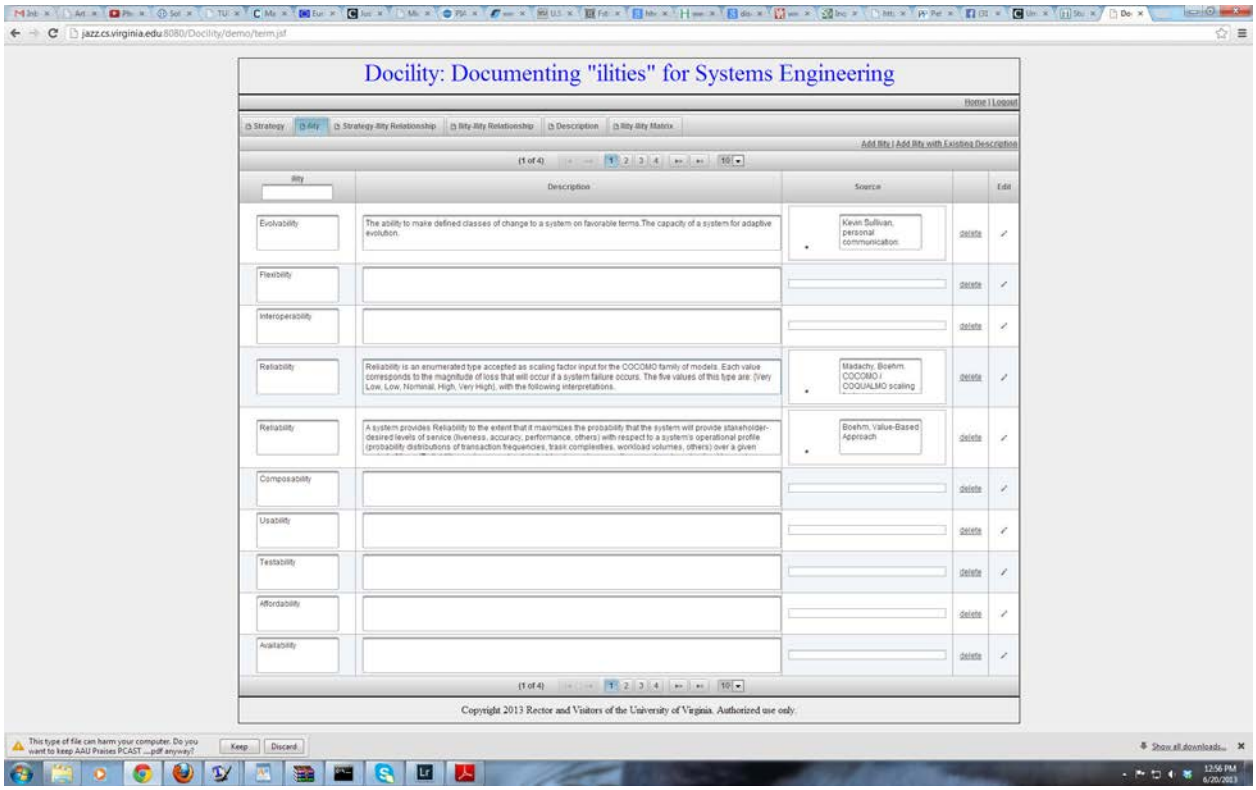


Figure 8: A Screenshot of Docility showing a list of partially completed ility definitions along with tabs for accessing other functions of the tool.

Formal methods for ility and relationship modeling and analysis.

Natural language and quasi-mathematical definitions of ilities, measures, analysis functions, propositions, and related artifacts are inadequate for the development of an effective framework for the comprehensive management of system ilities. This project aims to develop and test the notion that what is needed are mathematically precise, rigorously validated definitions of all such terms. We are thus employing a range of formal methods borrowed from the fields of software engineering and programming language design to specificy, verify and validate computational models of ilities, strategies, relationships among them, models, analysis functions, propositions, and evidence structures. Automated mechanisms that we are exploring include proof assistants, model checkers, Boolean satisfiability solvers , and related advanced technologies for the specification, debugging, verification and validation of computational concepts.

Our approach in Phase I has been to apply a particular "flavor" of formal methods—a constructive logic with inductive definitions and an automated proof assistant, namely Coq [1]—to formalize and then assess the model of changeability and related ilities of our team members, Ross et al. [2]. This work revealed both the feasibility and scientific and practical value of such an approach: in resolving ambiguities, identifying the kinds of gaps and errors that almost inevitable in quasi-formal models, in establishing a precise basis for model validation, and in providing a specification from which software can be synthesized automatically. We have made very considerable progress in this direction, and we are increasingly confident that it is a good, and necessary, approach to building an genuine science for comprehensive engineering management of ilities and tradeoffs.

Structure and Interpretation of Ility Assurance Cases.

Simply providing the means to express comprehensive ility specifications (including acceptable tradeoffs) is not enough to enable successful ility engineering. Nor does it suffice to develop related methods for achieving ilities and tradeoffs. What is needed, in addition, are rigorous approaches to structuring and evaluating *bodies of evidence* to support precise, detailed *propositions* asserting that systems have (or, if developed as specified, will have), ilities that satisfy given ility specifications.

We call such bodies of evidence *ility assurance cases.* Today, the science of ility assurance cases is extremely immature, and engineering methods and tools are woefully underdeveloped. Some of the best work on this topic is being done in the area software dependability, particularly software safety. This work is being driven in part by increasing demands from regulatory agencies for rigorous approaches to the certification of safety-critical systems, such as medical devices (FDA).

However, little, if any, work is being done on generalizing the approaches being developed by the dependability community to the full range of critical system ilities (e.g., affordability, evolvability, usability). Moreover, even within the dependability area, the science and engineering remains highly immature. Furthermore, it is not clear that approaches currently being pursued for dependability are right for the more general case of comprehensive ility management. In particular, safety cases are often structured around results of *hazard analyses.* Such an analysis produces an enumeration of threats to system safety. The propositions that are then supported by evidence are of the form *for all i, the i'th hazard is adequately addressed by the system design.* Comprehensive ility management will generally involve much richer forms of specification, requiring much more interesting and complex approaches to structuring evidence.

The key idea behind the approach that we have developed in Phase I of the iTAP project is that the comprehensive ility specifications should be documented in a formal logic language, and that the form of the evidence presented in support of such a proposition should be structured in parallel with the ***logical*** structure of the ility proposition being asserted. We are now developing an approach in which a highly expressive constructive logic and dependent type theory is used to write comprehensive ility specifications, and in which corresponding constructive logic "proof" techniques are used to organize the evidence in support of such propositions,

When formal propositions are about mathematical structures such as software programs, evidence comes in the form of actual mathematical proofs. The structures of such proofs are then determined by the forms of the propositions that they address. For example, the proof of a proposition, *A and B (*e.g., the system as designed is affordable and safe) is an *ordered pair of proofs* whose elements are a proof of A and a proof of B. A proof of *A or B,* is a pair containing either a proof of A or a proof of B or both. A proof of a proposition that an object having certain properties exists is a pair, the first element of which is an object and the second, a proof that *that* object has the given properties.
When the proposition to be proved has a rich logical structure, the *argument* (evidence) in support of the proposition has a correspondingly rich structure. When the argument is a formal proof developed in a proof checking system like Coq, its soundness as a proof can be checked automatically and efficiently using a form of type checking. An argument in support of a comprehensive ility proposition for a complex system will *not* be in the form of a formal proof. Rather, it will assemble highly heterogeneous evidence, such as signatures of engineers, predictions of cost models, the results of physical simulations, and software and system test and evaluation results. Clearly then the interpretation of such an argument will usually not be carried out automatically. Yet, such evidence will have to be structured and interpreted by experts.

The concept that we have developed and that we plan to develop further, is that the we *can* give precise logical expression to comprehensive ility specifications, and that we *can* using

corresponding constructive logic approaches to structuring "proofs" (evidence in support) of such propositions, *even though* the content that will inhabit such "proofs" will generally be informal.

A key expected benefit of this approach is that it provided a basis for constructing the schemas that will be needed to build automated tools to support storage, manipulation, presentation, and interpretation of complex, extensive, and evolving bodies of evidence in support of comprehensive ility assurance cases for complex systems.

View Relationship Representation.

In conclusion, the University of Virginia team is taking a multi-pronged approach to the development of a comprehensive approach to ility and tradeoff management, involving collaborative software; formal definitions of ilities, strategies, and relationships; and formal expression of ility specifications and corresponding structuring of ility assurance cases, in support of propositions that ility specifications are satisfied, which could mean that the achieved ilities are consistent with acceptable tradeoffs. If we are truly to develop a science of relationships among ilities, contingent on such factors as system types and application domains, then it is vitally important that we take: (1) a mathematically rigorous approach to modeling of ilities for purposes of ility and ility relationship definition; (2) a rigorous approach to expressing and assuring of comprehensive ility specifications for given systems; and (3) have a precise and detailed basis for developing powerfully supportive tools and methods for these purposes. The work that UVa pursued with its SERC colleagues in the iTAP project, Phase I, aimed to establish a strong initial position that would with high likelihood significantly advance our capabilities for rigorous, comprehensive ility and tradeoff management for complex systems.

References

1. Coq Development Team, "Coq Reference Manual," version 8.4pl2, available online at http://coq.inria.fr/refman/index.html

2. Ross, A.M, Rhodes, D.H, Hastings, D.E., "Defining Changeability: Reconciling Flexibility, Adaptability, Scalability, Modifiability, and Robustness for Maintaining System Lifecycle Value", Systems Engineering Journal, 2008 (pages 246-262)

1.2 PROCESS-ORIENTED VIEWS

1.2.1 EPOCH-ERA VIEW (MIT)

Epoch-Era Analysis (EEA), originally proposed in Ross (2006) and Ross and Rhodes (2008), is a multi-stage approach for identifying, structuring, and evaluating the impact of changing contexts and needs on systems. The approach combines two key concepts: "epochs" and "eras." The "epochs" part refers to the short run possible futures that may be experienced by a system. Described as a pair of possible contexts and needs, the *epochs* encapsulate one possible environment, among many, within which a system may find itself. A technically sound system may fail when confronted by unanticipated or harsh epochs. A particular time-ordered sequence of epochs is a possible system *era*. The path dependency of how epochs unfold over time may have a large impact on the time-varying success of a system. Strategies for delivering value over time can be considered for a system across possible eras. EEA can be viewed as consisting of two complimentary levels of analysis: Epoch-level (with both Single Epoch Analyses, Multi-Epoch Analysis, see Figure 9), and Era-level (with both Single Era Analyses, Multi-Era Analysis). These two levels require different levels of data availability and effort to conduct, but also provide different insights in terms of system success sensitivity to changes in context and needs within (single epoch analyses) and across (multi-epoch analysis) the uncertainty space, as well as the impact of path dependency of the uncertainties (era-level).

EEA can be used as a computational scenario planning method that provides a structured way to analyze the temporal system value environment. EEA decomposes the lifecycle of a system (comprising an "era") into sequential epochs that each have fixed context and value expectations (see Figure 10).

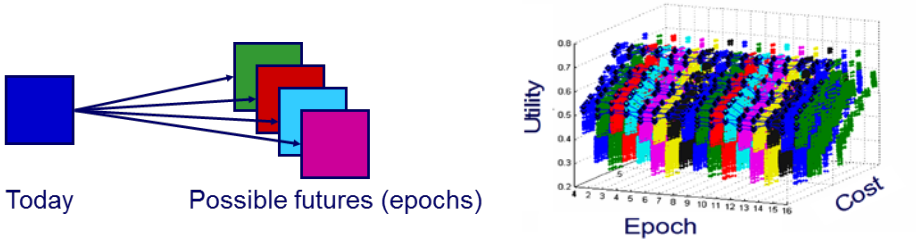


Figure 9. Epochs as Alternative "Point" Futures (l) and Multi-Epoch Analysis (r)

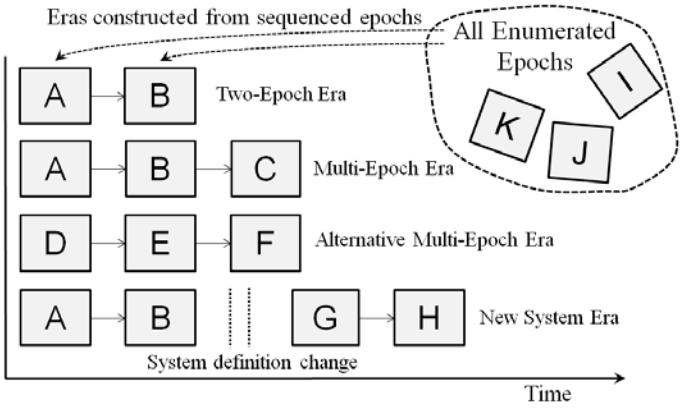


Figure 10. An era spanning a system lifecycle is subdivided into epochs which define alternative future value expectations and contexts (Rader, Ross and Rhodes 2010)

A key difficulty in implementing changeability in design has been the justification of the extra cost of its inclusion, as it typically requires longer development and/or additional technology. The benefits changeability gives are extremely difficult to extract and value in a static context, which has led to a systematic favoring of systems employing passive robustness. The EEA framework provides a means with which to intuitively explore system performance over time and across different contexts; it is the goal of ongoing research to find and implement a method to investigate and quantify changeability value using EEA, allowing it to be compared effectively to passive robustness in the design process.

EEA was designed to clarify the effects of time and context on the value of a system in a structured way. The base unit of time in the method is the *epoch*, which is a period of time defined by a fixed set of variables describing the context in which the system operates. These variables can encompass any exogenous circumstances that have an effect on the usage and value of the system: weather patterns, political scenarios, financial situations, operational plans, and the availability of other technologies are all potential *epoch variables*. The complete set of epochs, differentiated using these variables, can then be assembled into *eras*, ordered sets of epochs creating a description of a potential progression of contexts over time. This approach provides an intuitive basis upon which to perform analysis of value delivery over time for systems under the effects of changing circumstances and operating conditions, an important step to take when evaluating large-scale engineering systems with long lifespans. As system-exogenous changes trigger the start of a new epoch, the system may need to transform in order to sustain value, or else it may fail to meet expectations as defined for this new epoch, as illustrated in Figure 11. System Needs versus Expectations across Epochs of the System Era (Ross and Rhodes 2008).

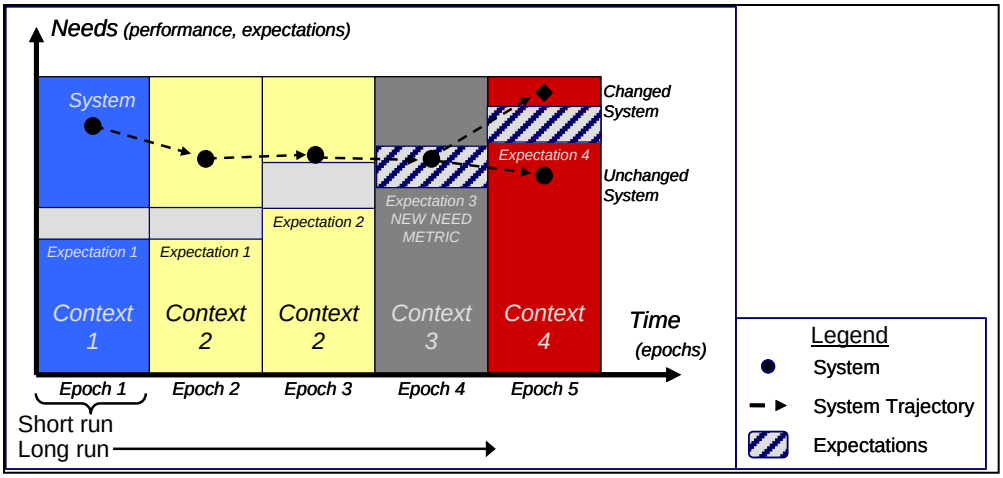


Figure 11. System Needs versus Expectations across Epochs of the System Era (Ross and Rhodes 2008)

Figure 11 illustrates the temporal evolution of a system as needs and contexts change. A system exists in Context 1 in Epoch 1 and has performance exceeding expectations. Expectations are represented by a band capturing the range from minimally acceptable to the highest of expectations. In Epoch 2, the context changes to Context 2 and the system when entering this context finds its performance is degraded. Yet, expectations are still met with the same system, so the system is relatively *robust* to the change in context. A change in expectation is shown in Epoch 3, with the context remaining the same as the second epoch; now the still unchanged system exhibits *value robustness* since it maintains value delivery in spite of changes in expectations. In Epoch 4, the system shows *versatility* by continuing to satisfy expectations despite the introduction of a new metric of need. Notice that even though the system no longer exceeds all expectations, it still does exceed the minimally acceptable level and thus is still successful. Finally, in Epoch 5, a change in context and a boost in expectations are too much for the system as-is; in this case the system must change in order to remain successful. If the system is capable of changing at acceptable cost, it is deemed *flexible* or *adaptable*, depending on the type of change desired (McManus et al., 2007).

The original application of Epoch-Era Analysis was to provide a temporal extension to Multi-Attribute Tradespace Exploration (MATE). MATE allows for the investigation of an extremely large design space, rating designs with a utility function that is constructed from nonlinear functions of multiple performance attributes (Ross et al 2004). The design space is populated by a computer model that evaluates the performance of an enumerated design vector. The potential design space becomes combinatorially large as the number of design variables considered increases. However, large design spaces can be used to generate a more complete understanding of the breadth of options available than would be given by a point-design study. Each of the designs can then be investigated across the epochs in EEA to provide insight into their performance in different contexts, and eras can be constructed to check lifetime performance across changing contexts.

Epoch-Era Analysis is not limited to tradespace exploration applications, as it employs a conceptual framework for considering the progression of time. Thus, EEA is equally applicable as a means of exploring lifecycle value for point-design studies. As long as there are exogenous variables that change over time and affect the performance or perceived value of the system, EEA can be used to define epochs of static context and eras of stochastically sampled epochs, which gives a wide range of potential projects and studies for EEA to support.

Effective summarizing metrics are key for understanding performance across uncertainties. These allow system designers and architects to quickly compare alternatives without having to manually proceed through design performance in each epoch. At the Epoch-level of analysis, there are two types of metrics: those that are cross-epoch, and those that are within-epoch. Cross-epoch metrics summarize system value across the alternative unordered epochs, while within-epoch metrics summarize the impact of that particular epoch on the systems. *Normalized Pareto Trace* is an example cross-epoch metric, while *Yield* is an example within-epoch metrics. Both types can be useful for gaining insights into the impact of uncertainties and the effectiveness and efficiency of system responses (e.g. robustness, changeability, or evolvability). Additionally, metrics relate to the two "strategies" depicted in Table 2 change (i.e. "changeability") or no-change (i.e. "robustness" or "versatility").

Table 2 below lists a number of multi-epoch metrics, with type indicated, as well as "value aspect" usually related to ilities. For clarity, a *K-percent fuzzy Pareto front* includes all designs within both *K* percent of the total cost range, and *K* percent of the total utility range, of a Pareto front design. Additionally, low yields indicate difficult conditions or demanding needs; epochs with these characteristics may require extra attention.

Table 2. Example Multi-Epoch Metrics

Value Aspect	Type	Acronym	Stands For	Definition
Degree of changeability	Within	OD	Outdegree	# outgoing transition arcs from a design
Degree of changeability	Within	FOD	Filtered Outdegree	Above, considering only arcs below a chosen cost threshold
Epoch difficulty	Within	YN	Yield	Fraction of design space considered valid within an epoch
"Value" gap	Within	FPN	Fuzzy Pareto Number	% margin needed to include design in the fuzzy Pareto front
"Value" of a change	Within	FPS	Fuzzy Pareto Shift	Difference in FPN before and after transition
Robustness via "no change"	Cross	NPT	Normalized Pareto Trace	% epochs for which design is Pareto efficient in utility/cost
Robustness via "no change"	Cross	fNPT	Fuzzy Normalized Pareto Trace	Above, with margin from Pareto front allowed
Robustness via "change"	Cross	eNPT, efNPT	Effective (Fuzzy) Normalized Pareto Trace	Above, considering the design's end state after executing a change option
"Value" of a change across epochs	Cross	FPS Dist	Fuzzy Pareto Shift Distribution	Epoch frequency of FPS scores for a design across epochs

Example case studies using these metrics can be found in Fitzgerald and Ross (2012a), Fitzgerald and Ross (2012b), Ross, Rhodes, and Hastings (2009), and Ross et al (2009).

References

Fitzgerald, M.E. and Ross, A.M., "Mitigating Contextual Uncertainties with Valuable Changeability Analysis in the Multi-Epoch Domain," 6th Annual IEEE Systems Conference, Vancouver, Canada, March 2012.

Fitzgerald, M.E. and Ross, A.M., "Sustaining Lifecycle Value: Valuable Changeability Analysis with Era Simulation," 6th Annual IEEE Systems Conference, Vancouver, Canada, March 2012.

McManus, H.M., Richards, M.G., Ross, A.M., and Hastings, D.E., "A Framework for Incorporating "ilities" in Tradespace Studies," AIAA Space 2007, Long Beach, CA. September 2007.

Rader, A.A., Ross, A.M., and Rhodes, D.H., "A Methodological Comparison of Monte Carlo Methods and Epoch-Era Analysis for System Assessment in Uncertain Environments," 4th Annual IEEE Systems Conference, San Diego, CA, April 2010.

Ross, A.M., and Rhodes, D.H. (2008), "Using Natural Value-Centric Time Scales for Conceptualizing System Timelines through Epoch-Era Analysis," INCOSE Int'l Symposium 2008, Utrecht, the Netherlands, June 2008.

Ross, A.M., Hastings, D.E., Warmkessel, J.M., and Diller, N.P., "Multi-Attribute Tradespace Exploration as Front End for Effective Space System Design," Journal of Spacecraft and Rockets, Vol. 41, Jan/Feb 2004, pp. 20-28.

Ross, A.M., McManus, H.L., Rhodes, D.H., Hastings, D.E., and Long, A.M., "Responsive Systems Comparison Method: Dynamic Insights into Designing a Satellite Radar System," AIAA Space 2009, Pasadena, CA, September 2009.

Ross, A.M., Rhodes, D.H., and Hastings, D.E., "Using Pareto Trace to Determine System Passive Value Robustness," 3rd Annual IEEE Systems Conference, Vancouver, Canada, March 2009.

1.2.2 SET-BASED AND INSIDE-OUT VIEWS (NAVSEA; WSU)

1.2.2.1 Philosophy of Set-Based Design

Set-based design is also known as "set-based concurrent engineering." It is often contrasted to point-based design. Both set-based and point-based design require a prior understanding of the generic system architecture and options, tools to assess compatibility/feasibility of combinations of choices, and implications for performance.

The philosophy of set based design is to keep options open and delay decisions until it is clear that some options are infeasible or dominated by other options, or until schedule pressures

require a decision. The idea is to succeed by avoiding failure, not to build an "optimal" design. The principle is to involve multiple interest groups concurrently, for them individually to reject infeasible regions of the tradespace and identify dominated solutions. Set-based design incrementally narrows the tradespace by progressively integrating the views from different interest groups finding mutually feasible regions, and to expand or restrict the tradespace by adjusting the "threshold and objective" performance levels. The basic concepts of set-based by design are:

- Defer decisions until they have to be made
- Analyze the tradespace simultaneously from different perspectives, find the feasible region from each perspective – nested set of feasible regions at different capability levels
- Eliminate infeasible and inferior regions to narrow the tradespace by considering the intersections of feasible regions
- End with one or more (disjoint) regions
- Then decide which (one or more) to proceed with.

Set-based design is often contrasted to point-based design. While there are many ways to implement point-based design, the general idea is to begin by making the decisions on those aspects of the architecture that (1) most constrain the design space, i.e. have the most long-reaching implications, and (2) are most important for achieving the desired performance, i.e., most enabling or limiting. This basic step is iterated until the design is complete, or there is a conflict or incompatibility, or shortfall relative to desired performance. If there is a performance shortfall, the required capability level may be reduced. If there is an incompatibility, the early decisions are reconsidered.

1.2.2.2 A Simple Example Contrasting Set-Based and Point-Based Design

Consider the problem of scheduling a meeting in a large organization.

Using point-based design, I would pick a time, send out an email inviting the people I want to have participate asking if they can make that time. If enough say "yes", I schedule the meeting, and ask those who cannot make it if they can send an alternate, or if there was some way they could juggle their schedule. If not enough say "yes", I pick another time and try again. I might suggest a few times in the initial email, then pick the time with the largest consensus. The other participants might suggest an alternate time, and might suggest some other people who should be invited. The potential problems are that some people who should attend may not be invited, and there may have been a better time at which more of the principles and/or their alternates could have attended. Not a lot of people are involved.

Using set-based design, I would send an announcement of the meeting to all the department heads inviting them to send a representative if their department should be represented, and a scheduling form for each designated representative to make the times that it would be difficult to attend. When I get the forms back, I would look for the intersection to find feasible times. If there is a large feasible region, I could resend the scheduling form with a higher threshold: exclude times that would be inconvenient to attend. If there were no feasible times for everyone, I could resend form with a lower threshold: exclude times that would be difficult but not impossible to attend. Alternatively, I could open up the design space by asking everyone to identify a potential alternate and ask them to fill out the scheduling survey. This takes more up-front work, but it ensures no departments are overlooked, and maximizes attendance.
In order to apply point-design, I needed to know who I wanted to attend and my minimum attendance threshold. In order to apply set-based design, I needed to know the organization (departments). The department heads needed to be able to decide if their department needed to be represented, and, if so, to identify a representative. I also needed a tool to evaluate joint feasibility (in this case, the trivial intersection of times). In both approaches, each individual responder needed to be able to evaluate their regret function for any given time, and, potentially, to be able to identify an alternate.

1.2.2.3 Hopes For Set-Based Design

Hoped-for benefits from set-based design include the following:

- •More rapid and efficient response to changes in requirements, context, needs and priorities during the development process
- •Lower risk of not meeting affordability, capability and suitability objectives
- •Designs with fewer unexpected incompatibilities
- •Designs that require less rework – fewer and less extensive engineering change orders – to fix problems, correct oversights and incompatibilities, and improve reliability
- •Designs that are more resilient, i.e. more robust with respect to use and operating conditions, and more versatile with respect to future mission needs and technology opportunities (The term “versatile” is used as a general term encompassing flexible, adaptable, changeable, extensible, etc.)

By itself, set-based design does not ensure more versatile designs. “Inside-out” design is one step in this direction (described in the following section). Versatility also requires that the systems be designed with sufficient reserve capacity (also called “design margin”) for future modifications and upgrades: electrical power for additional equipment, drive power for increased weight, cooling for increased thermal load, volume and surface area to install

equipment, structural strength for greater loads and shocks, etc. Further analysis of the needs and methods to ensure the versatility of long-lived systems are needed.

1.2.2.4 NAVSEA and Set-Based Design

On February 4, 2008 Admiral Paul Sullivan, Commander of the Naval Sea Systems Command, sent out a letter entitled: Ship Design and Analysis Tool Goals. The purpose of the widely distributed memorandum was to state the requirements and high-level capability goals for NAVSEA design synthesis and analysis tools. In this memo, Admiral Sullivan expressed the need for evolving models and analysis tools to be compatible with, among other things, Set-Based Design.

The recent and typical practice has been to estimate the weight and volume of the of everything that will go into the ship, design the hull form based on the weight and volume, then later try to configure all the components within the hull in so they can work together. Problems attributed to this "outside-in" design include:

(1) non-optimum hydrodynamic hull form designs which significantly increase energy consumption and the fuel bills for the Fleet;
(2) difficulty in maintaining and repairing ships due to space limitations and the "tightness" of the ship arrangements;
(3) insufficient service-life allowances for weight and/or space, thus increasing modernization costs;
(4) significant reductions in terms of years of the economical service-life of ships;
(5) possible operational restrictions due to the inability to develop robust designs.

In contrast, "inside-out" design first creates the functional arrangement of spaces, then sizes the hull to fit the functional arrangement. There are different functional arrangements depending on the granularity of subsystem decomposition.

Combining "inside-out" design with set-based design takes this a step further by considering the design space of alternative functional arrangements, and hull configurations for each, then rejecting combinations with inferior functional capability, and hydrodynamic mobility and stability. This approach favors hull configurations that support the greatest variety of functional arrangements. However it does not address the issue of reserve capacity (design margin) in weight and volume capacity, engine/drive power, cooling, compute power, or communications bandwidth to enable the insertion of new capabilities and/or mission modules.

1.2.3 EVIDENCE-RISK-BASED PROCESS VIEW (USC)

Having evidence serve as the principal decision criterion at milestone decision reviews is a considerable step forward from traditional schedule-based or event-based reviews. An initial step forward in systems engineering and acquisition guidance was to progress from schedule-based major project reviews (the contract says that the Preliminary Design Review is scheduled for September 30, so we'll have it then, ready or not) to event-based reviews (the PDR will be held when there is a preliminary design to review). This is better, but frequently leads to "Death by PowerPoint and SysML" reviews. These present much design detail, but there is little time to determine whether or not the design will meet the system's key performance parameters or ilities. Such evidence of feasibility is generally desired, but is considered as an optional appendix and not a project deliverable. Thus, it is often neglected when budgets are tight, and contractual progress payments and award fees are based on producing a design and not evidence of its feasibility.

In an evidence-based review, the feasibility evidence is a first-class deliverable. As such, its planning and preparation becomes subject to earned value management and is factored into progress payments and award fees. Investments in feasibility evidence have been found to pay off significantly in development rework avoidance. The key content of a Feasibility Evidence Description is shown in Figure 12.

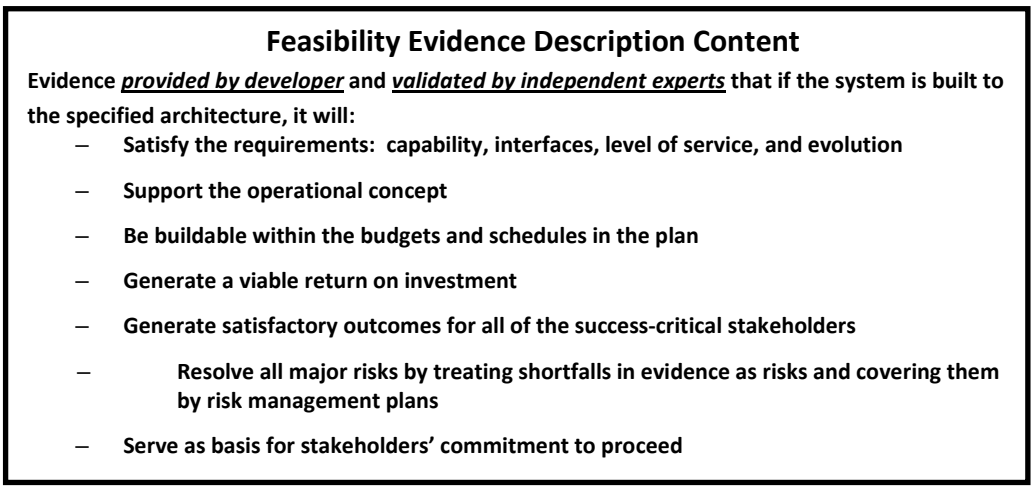


Figure 12. Feasibility Evidence Description (FED) Content

What Feasibility Evidence Development Isn't

Feasibility evidence development is not Evidence Appreciation. Frequently, systems engineers involved in developing models and simulations to evaluate feasibility become so engrossed in the elegance and detail of their models that they do not complete them until after the key decision milestone for which they are needed is past. As with many other systems engineering

and development "how much is enough" questions, the best way to address this is to balance the risks of doing too little evidence development vs. the risks of doing too much.

Some key drivers in this regard are the project's size and criticality, which point toward having more evidence; and the project's rate of requirements and technology change; which point away from generating a lot of evidence that will quickly become obsolete. These decision drivers are quantified in Figure 13 below.

Note the difference between having and generating evidence. Often, a well-prepared phone call to a representative previous-project user of a prospective COTS product being evaluated for project use will produce superior evidence of feasibility or infeasibility than will weeks of COTS product exercise. As a first-class deliverable, the FED's development should be preceded by careful planning and evaluation of alternative ways of obtaining evidence.

Feasibility evidence development is also not Analysis Paralysis. Often, COTS or cloud services evaluations become caught in a delay loop in which a new vendor announcement becomes an excuse to delay making a decision until the actual glories of the vendor announcement are available and can be evaluated. If users need new capabilities soon or the product being developed has a short market window, the project can't afford to wait. At best, it should do a quick assessment of how much of the announcement is likely to be vaporware, and how the system could be architected to accommodate the new COTS product or service if and when its glories become reality.

How Much Feasibility Evidence Development is Enough?

Size, criticality, and volatility are key decision drivers for focusing on agile or architected approaches. But critical questions remain about how much architecting and feasibility evidence development is enough for a particular project. Here we provide a quantitative approach that has helped projects address this question. It extends the ROI of SE analysis described in [Boehm-Valerdi-Honour, 2008].

The graphs in Figure 13 show the results of a risk-driven "how much feasibility evidence is enough" analysis, based on the COCOMO II Architecture and Risk Resolution (RESL) factor. This factor was calibrated along with 22 others to 161 project data points. It relates the amount of extra rework effort on a project to the percent of project effort devoted to software-intensive system architecting and feasibility evidence development. The analysis indicated that the amount of rework was an exponential function of project size.

A small (10 thousand equivalent source lines of code, or KSLOC) could fairly easily adapt its architecture to rapid change via refactoring or its equivalent, with a rework penalty of 14% between minimal and extremely thorough architecture and risk resolution. However, a very large (10,000 KSLOC) project would incur a corresponding rework penalty of 91%, covering such

effort sources as integration rework due to large-component interface incompatibilities and critical performance shortfalls.

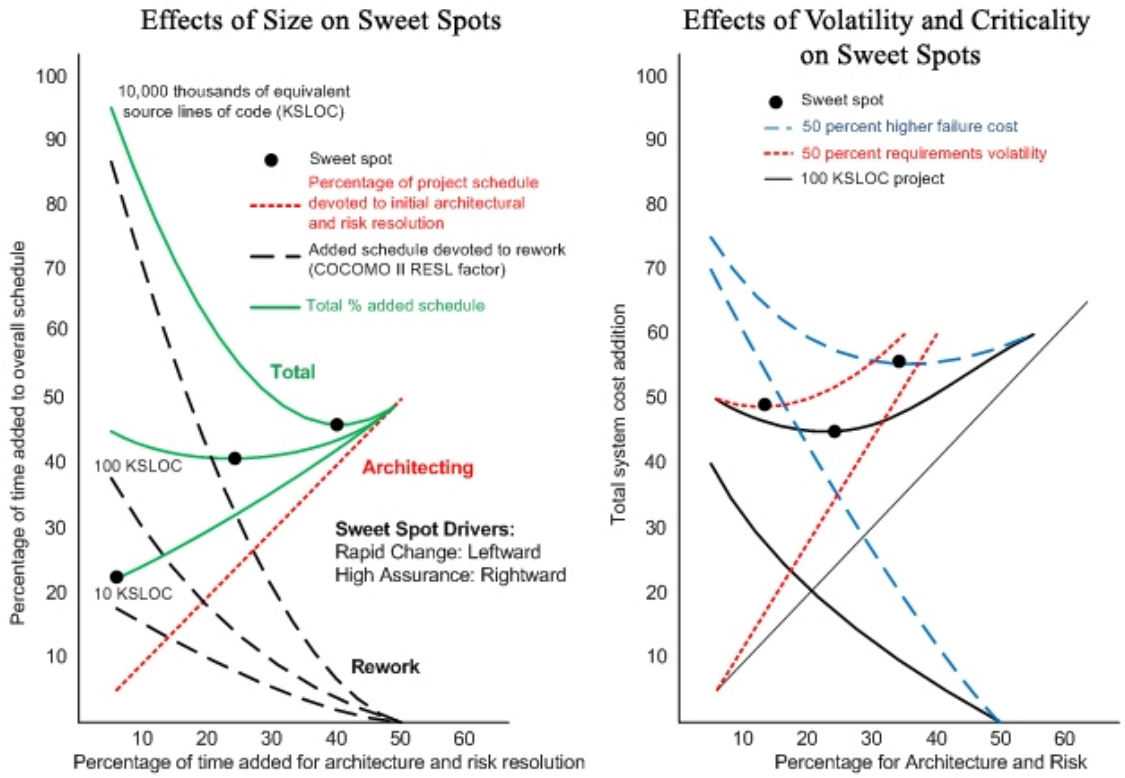


Figure 13: Size, Volatility, Criticality, Effects on Feasibility Evidence Sweet Spots

Actually, the RESL factor includes several other architecture-related attributes besides the amount of architecting investment, such as available personnel capabilities, architecting support tools, and the degree of architectural risks requiring resolution. Also, the analysis assumes that the other COCOMO II cost drivers do not affect the project outcomes.

The effects of rapid change (volatility) and high assurance (criticality) on the sweet spots are shown in the right hand graph. Here, the solid black lines represent the average-case cost of rework, architecting, and total cost for a 100-KSLOC project as shown at the left. The dotted red lines show the effect on the cost of architecting and total cost if rapid change adds 50% to the cost of architecture and risk resolution. Quantitatively, this moves the sweet spot from roughly 20% to 10% of effective architecture investment (but actually 15% due to the 50% cost penalty). Thus, high investments in architecture and other documentation do not have a positive return on investment due to the high costs of documentation rework for rapid-change adaptation.

The dashed blue lines at the right represent a conservative analysis of the effects of failure cost of architecting shortfalls on the project's effective business cost and architecting sweet spot. It assumes that the costs of architecture shortfalls are not only added rework, but also losses to the organization's operational effectiveness and productivity. These are conservatively assumed to add 50% to the project-rework cost of architecture shortfalls to the organization. In most cases for high-assurance systems, the added cost would be considerably higher.

Quantitatively, this moves the sweet spot from roughly 20% to over 30% as the most cost-effective investment in architecting for a 100-KSLOC project. It is good to note that the "sweet spots" are actually relatively flat "sweet regions" extending 5-10% to the left and right of the "sweet spots." However, moving to the edges of a sweet region increases the risk of significant losses if some project assumptions turn out to be optimistic.

Again, the effects of other factors may affect the location of a given project's "how much evidence is enough" sweet spot. A good cross-check is to use the Constructive Systems Engineering Cost Model, COSYSMO [Valerdi, 2008], to estimate the project's amount of needed systems engineering effort. A third approach is to use the risk-based decision heuristic of balancing the risks of doing too little evidence generation with the risks of doing too much (the balance between "Look before you leap" and "He who hesitates is lost.").

1.2.3.1 Evidence Development Planning and Control

Table 3 outlines a process that can be used for developing the Feasibility Evidence Description (FED). The process clearly depends on Step A: having the appropriate work products for the phase (these are basically the system operational concept, requirements specification, architecture, and life cycle plan). Since these are not fully developed in the early phases, the work products and the FED are developed concurrently and influence each others' content and preparation activities. To reflect this, the "Steps" are denoted by letters rather than numbers to indicate that many are done concurrently.

Table 3. Steps for Developing the Feasibility Evidence Description (FED)

Step	Description	Examples/Detail
A.	Develop phase work-products/artifacts	See the ICM Anchor Point Milestone Content charts provided Chapters 6, 8, and 9 for each phase and anchor point milestone.
B.	Determine most critical feasibility assurance issues	Issues for which lack of feasibility evidence is program-critical
C.	Evaluate feasibility assessment options	Cost-effectiveness, risk reduction leverage/ROI, rework avoidance Tool, data, scenario availability

D.	Select options, develop feasibility assessment plans	
E.	Prepare FED assessment plans and earned value milestones	Try to relate earned value to risk-exposure avoided rather than budgeted cost
F.	Begin monitoring progress with respect to plans	Also monitor project/technology/objectives changes and adapt plans
G.	Prepare evidence-generation enablers	Assessment criteria Parametric models, parameter values, bases of estimate COTS assessment criteria and plans Benchmarking candidates, test cases Prototypes/simulations, evaluation plans, subjects, and scenarios Instrumentation, data analysis capabilities
H.	Perform pilot assessments; evaluate and iterate plans and enablers	
I.	Assess readiness for Commitment Review	Shortfalls identified as risks and covered by risk mitigation plans Proceed to Commitment Review if ready
J.	Hold Commitment Review when ready; adjust plans based on review outcomes	
NOTE: "Steps" denoted by letters rather than numbers to indicate that many are done concurrently.		

In Step B for each phase, feasibility assurance issues are prioritized based on their criticality to the success of the system development program. A risk-based approach such as the framework and tools provide in Appendix B can help determine the priorities. In Steps C and D, the feasibility assessment options may include stakeholder need identification and prioritization techniques, prototypes, models, simulations, benchmarking of candidate solution elements, and cost and schedule analyses. The costs and schedules of each option are also needed to determine their relative cost-effectiveness in reducing project risks. Chapter 11 elaborates on each of these steps.

In Step E, the FED activities are scheduled, resourced, and assigned an appropriate earned value. Advanced earned value techniques would base the earned value on the expected risk exposure cost reduction to be provided by the activity, not the activity's budgeted cost. The cost estimates for each FED development activity will help in providing a bottom-up cost estimate for the project's system engineering activity. The percentage-of-project-time estimates shown in Figure 13 can help, but they are just based on the project's size, criticality, and dynamism. Other factors may influence the amount of time and effort required to develop feasibility evidence, such as the degree of understanding of the requirements, architecture,

personnel capability and experience, and reusability of existing models, simulations, testbeds, and results. The Constructive Systems Engineering Cost Model (COSYSMO) [Valerdi 2008] can be helpful in providing additional cost perspective.

1.2.3.2 Preparing for Evidence-Based Reviews

Figure 14 highlights the activities that need to be performed in preparation for the review, the actual review, as well as the post-review activities and follow-up. The entry criteria include ensuring that the feasibility evidence preparation has been successfully tracking its earned value milestones. The inputs include identifying committed expert reviewers for each of the review questions, and familiarizing them with the review process.

The review meeting will include not only the developer SEs and the expert reviewers, but also the stakeholder upper-management decision-makers, who will need some context-setting before the developer responses to reviewer issues are discussed. The review exit criteria and tasks include key stakeholder concurrence on the way forward and commitment to support the next phase, as well as action plans and risk mitigation plans for the issues identified.

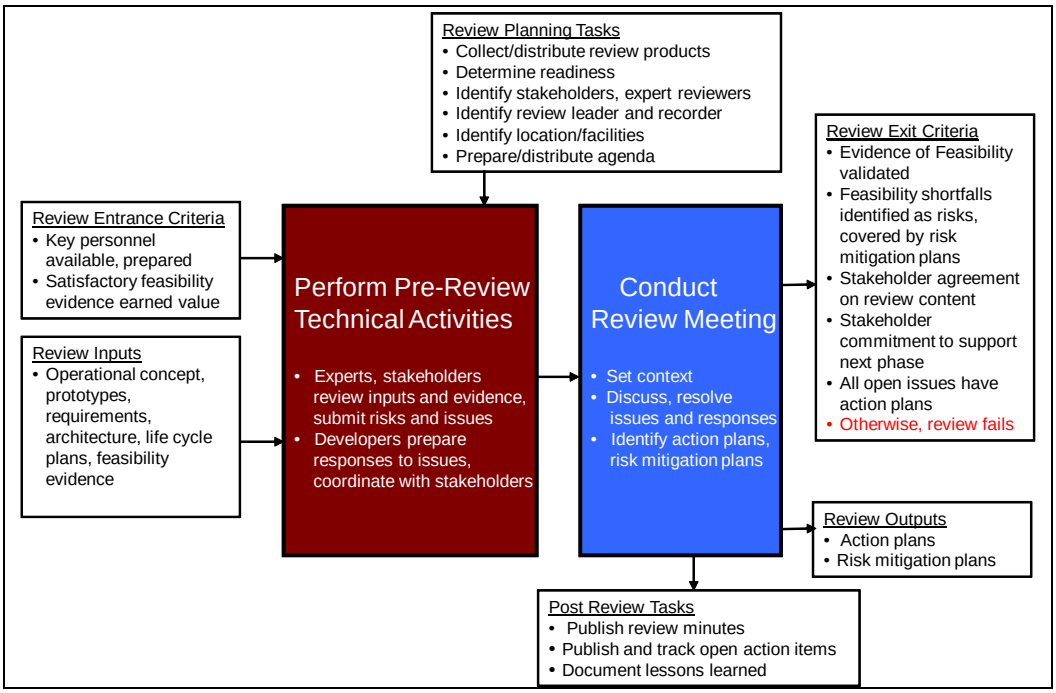


Figure 14. Overview of Commitment Review Process

1.3 MEANS-ENDS VIEWS

1.3.1 AFFORDABILITY EXAMPLE (USC)

Many approaches to improving system affordability focus on one or two strategies (e,g., automation, outsourcing, repurposing, reuse, process maturity), and miss the opportunity for further improvements. Often, they focus on one phase (e.g., acquisition) at the expense of other factors that increase total ownership cost (TOC).

Based on several related research projects, we have developed, applied, and evolved an orthogonal framework of strategies for improving Affordability. The framework includes:

- Get the best from people (Staffing, Teambuilding, Facilities, Kaizen)
- Make tasks more efficient (Tools, Work and Oversight Streamlining, Collaboration Technology)
- Eliminate tasks (Lean and Agile Methods, Automation, Model-Based Product Generation)
- Eliminate scrap and rework (Early Risk and Defect Elimination, Evidence-Based Decision Gates, Modularity around Sources of Change, Evolutionary Development, …)
- Simplify products: KISS (Risk-Based Prototyping, Value-Based Capability Prioritization, Satisficing vs. Optimizing)
- Reuse components (Domain Engineering, Composable Components and Services, Legacy Repurposing)
- Reduce O&S costs (Automate Tasks, Design for Maintainability, Streamline Supply Chain, Anticipate/Prepare for Change)
- Perform value-based tradespace analysis among the above.

The research presented will also summarize the use of calibrated quantitative cost models for reasoning about the strategy elements and their tradeoffs.

The INCOSE Systems Engineering Handbook [1] defines affordability as, “The balance of system performance, cost, and schedule constraints over the system life cycle, while satisfying mission needs in concert with strategic and organizational needs.” Similar definitions are cited by the National Defense Industrial Association (NDIA) and Military Operations Research Society

(MORS) Affordability Working Groups. The most significant aspect of this definition is that it emphasizes not only cost reduction but also addition to stakeholder value.

Figure 15 illustrates this perspective on affordability. It indicates that one's objective is to achieve combinations of added benefits and reduced costs that enable their organization to advance its current state further and further away from the origin (the organization's original

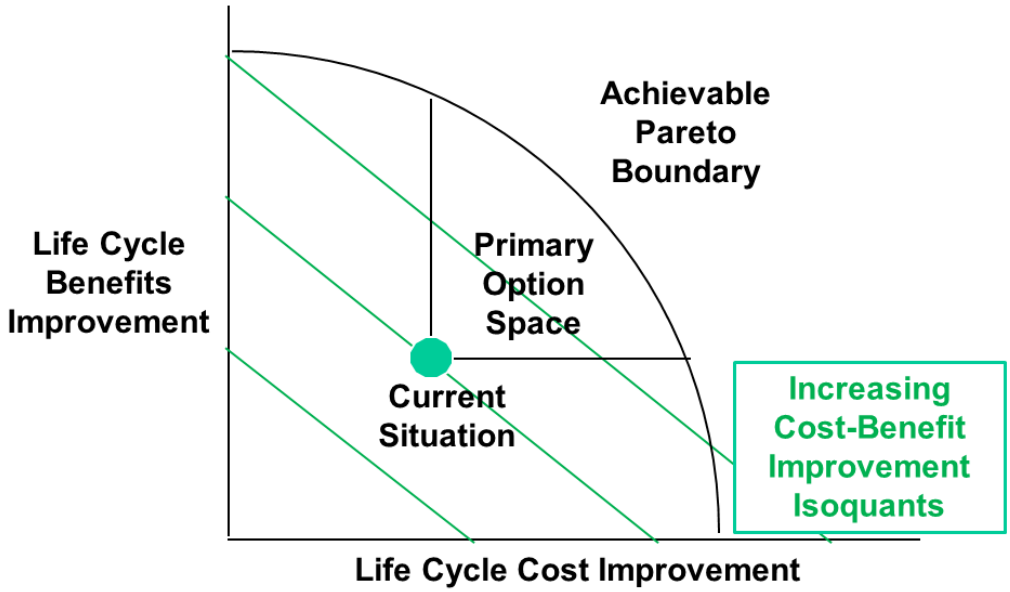


Figure 15 Affordability Improvement Options

costs and benefits) toward its Pareto boundary of achievable combinations of increased benefits and reduced costs. Its primary option space will include strategies that increase benefits at no loss in cost savings, reduce costs at no loss in increased benefits, or mixed strategies in between.

However, there are also common special cases, such as an overall-organization 10% budget cut. Too often, such cases are handled by imposing 10% across-the-board cuts in each area; singling out easy-to-cut costs such as travel, equipment, and education; or dismissing employees with the least or most seniority. These usually have serious long-term negative effects. More far-seeing organizations have affordability engineering functions that pursue more long-term strategies that enable them to execute budget cuts in more cost-effective ways.

1.3.1.1 The Orthogonal Framework for Improving Affordability

In this context, Figure 16 shows the orthogonal (in terms of classes of options) framework for improving affordability. It has evolved over several decades of related industrial and academic research and development [2]. Each class has several options that have been found to be cost-effective across many application domains. For each option, an organization can assess its current state with respect to the identified improvement candidates, and can determine which

candidates are the current best fit for pursuing. For several of the options, quantitative data are available for assessing the effects of improvements, as will be discussed below.

Getting the Best from People

The first major option is to get the best from people through improvements in staffing, teambuilding, facilities, and involving its people in identifying and executing Kaizen (good ideas can come from anyone, particularly performers) approaches for increasing benefits or reducing costs. Staffing improvements involve initiatives for identifying and prioritizing current and future needs for personnel knowledge, skills, and abilities; and realizing these needs through improved staffing, retention, and career path facilitation via education, training, and mentoring. For systems engineering (SE), competency models are available for assessing and improving SE competency, such as the INCOSE-UK SE Competencies Framework 2010-0205 Issue 3 [1] and the Mitre SE Competency model [3].

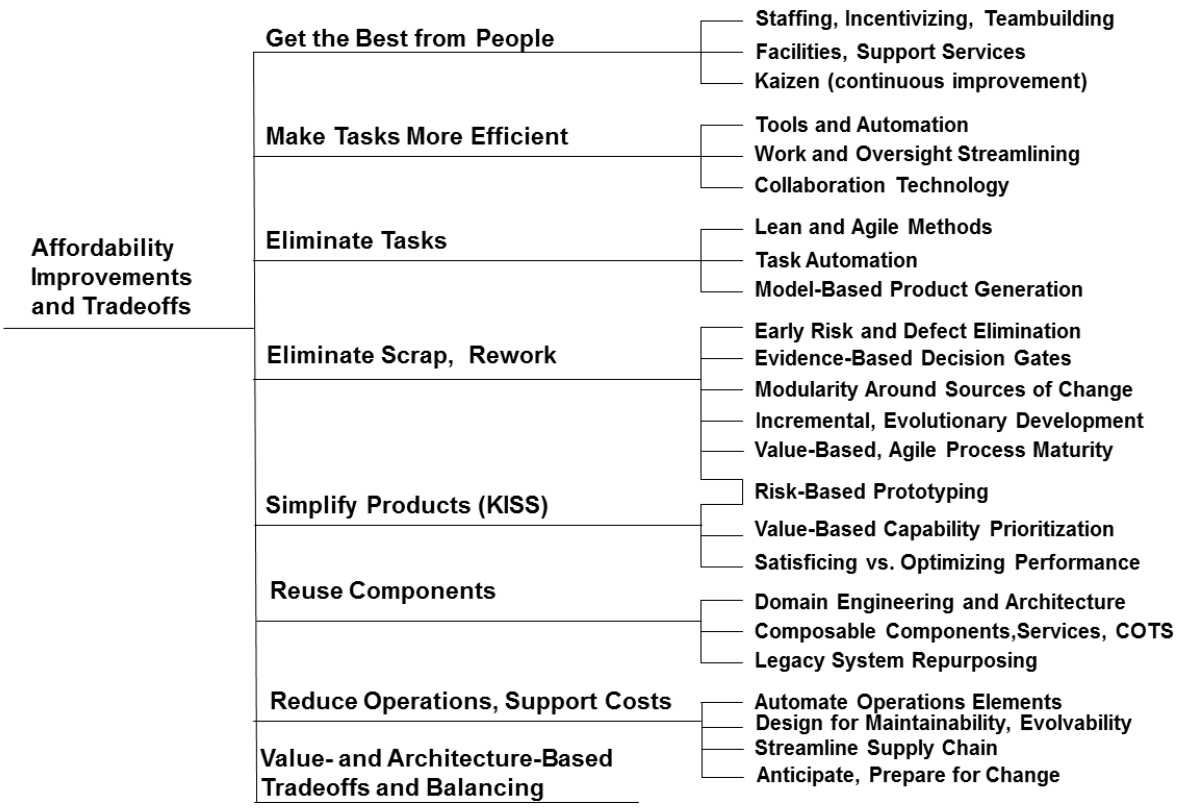


Figure 16 The Affordability and Tradespace Options Framework

The People Capability Maturity Model [1] provides a good detailed framework and set of practices for continuous improvement of an organization's human resources. Facilities and support services for knowledge workers will include techniques to reduce distractions and increase thought flow, as described in sources such as Peopleware [5].

Incentivizing can include reducing project turnover via project completion bonuses and flowdown of project award fees to individuals. At a group level, it involves balancing individual incentives with group incentives. At a cross-organizational level involving suppliers, distributors, or strategic partners, it can involve such practices as Vested Outsourcing or shared-destiny contracting [6,7], in which organizations determine each other's' value propositions or win conditions, and develop win-win arrangements in which collaborative efforts provide positive outcomes with respect to each participant's value propositions [8,9,10]. Such practices also contribute to teambuilding; a good source for further teambuilding practices is [11].

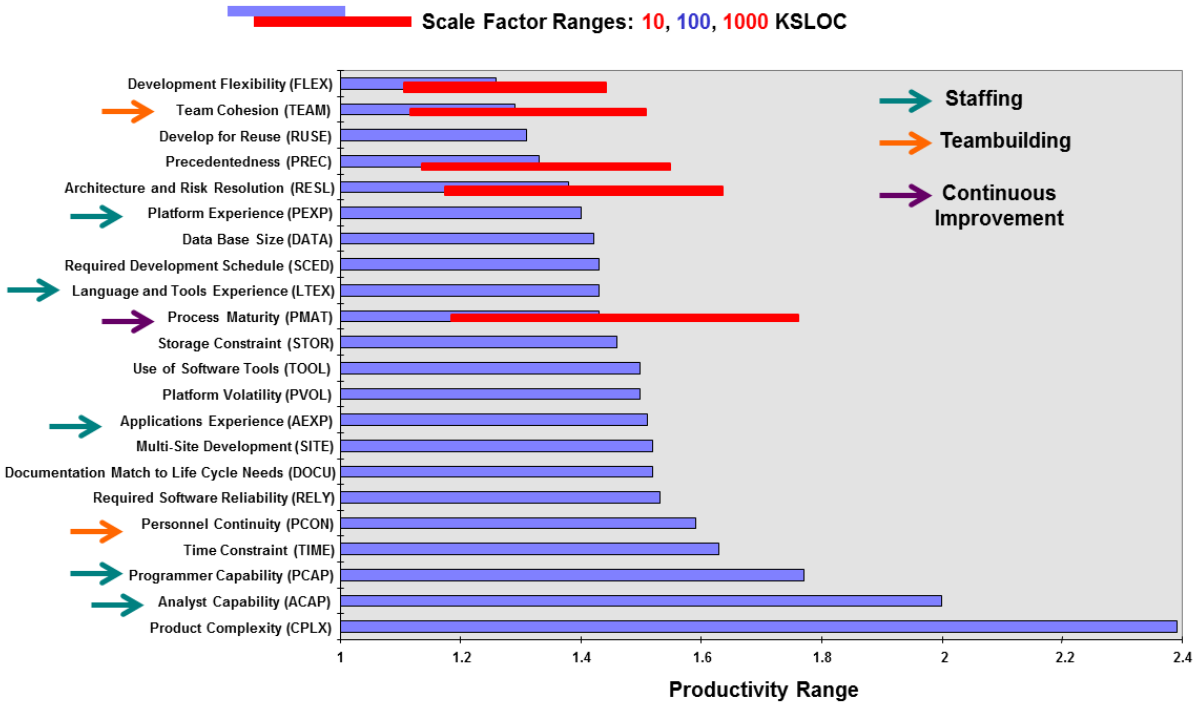


Figure 17 Quantitative Software Cost Improvement Insights from COCOMO II

For software projects, a quantitative framework for identifying the project's current status and estimating potential strategies with respect to improving software productivity and affordability via people factors is shown in Figure 17. The green arrows represent COCOMO II [12] productivity ranges (the ranges in effect on project cost of having very poor or very good ratings) for staffing factors such as analyst capability, programmer capability, applications experience, platform or infrastructure (hardware, operating system, data management system, etc.) experience, and programming language and tool experience. For each of these factors, an organization can identify their projects' average COCOMO II rating level, which identifies the project's current location on the productivity range, and can then identify strategies for increasing the rating level and determining the resulting productivity or affordability increase associated with the increase in rating level.

Thus, for example, if an organization had High (75%) levels of Programmer Capability but Low (35%) levels of Analyst Capability, investment in systems engineering staffing, education, career pathing, and mentoring that raised the Analyst Capability level to Nominal (55%) would reduce relative effort from 1.19 to 1.00, for an affordability gain of 19%. Further initiatives to raise the level to High would reduce the relative effort to 0.85, for an affordability gain of 40%. A framework and tool for performing such assessments is the COPROMO tool described in chapter 5 of [12].

Corresponding productivity ranges for improvements in incentivizing and teambuilding are shown by the orange arrows in Figure 17, representing benefits resulting from improved personnel continuity and team cohesion. Some of the COCOMO II productivity ranges such as team cohesion have exponential effects as a function of project size. The productivity difference between a Very Low and Extra High level of team cohesion is a factor of 1.13 for a 10 thousand source lines of code (KSLOC) project, and 1.46 for a 1000 KSLOC project. The productivity ranges were determined by a Bayesian combination of group expert judgment and a multiple regression analysis of the contributions of each factor to the overall productivity of 161 projects in the COCOMO II database. Productivity and affordability gains due to advances in continuous process improvement (designated by the purple arrow) were determined by using the project's Capability Maturity Model level [13,14] as a rating scale. The resulting productivity ranges are 1.20 for a 10-KSLOC project and 1.71 for a 1000-KSLOC project.

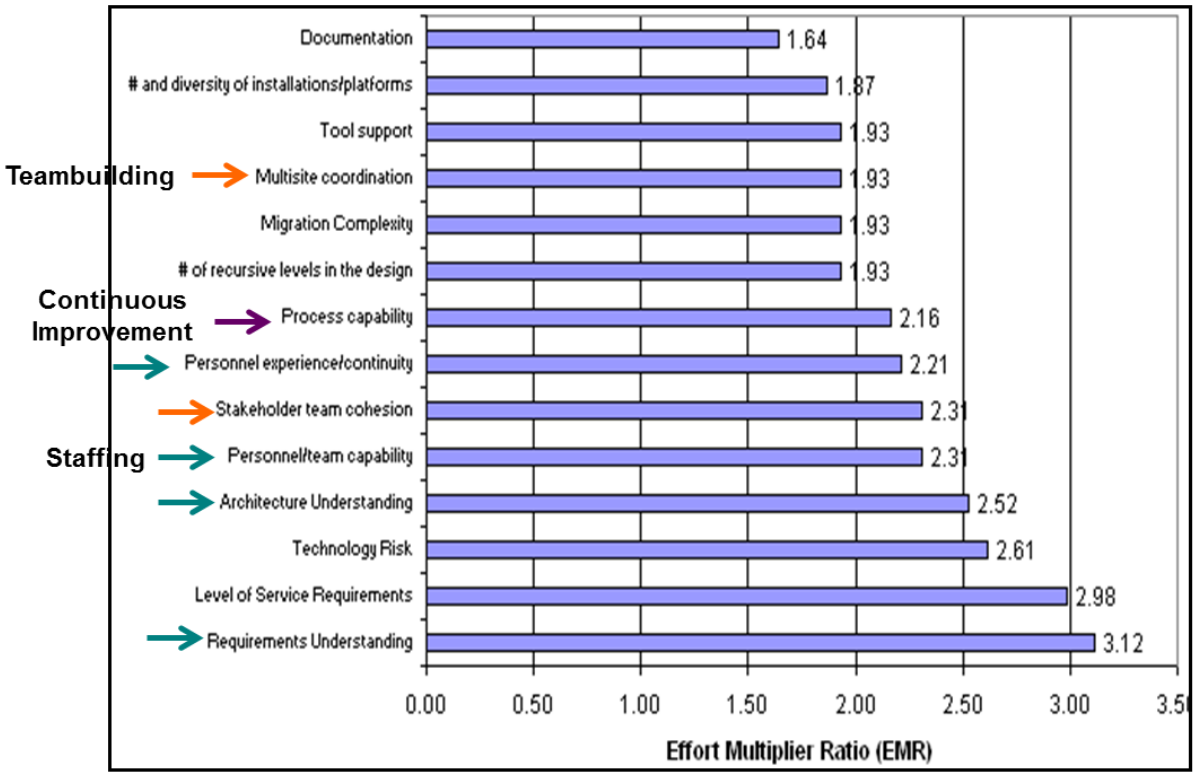


Figure 18 Quantitative SE Cost Improvement Insights from COSYSMO

A similar quantitative framework for identifying the project's current status and estimating potential strategies with respect to improving systems engineering (SE) productivity and affordability is shown in Figure 18, based on the calibrated factors in the COSYSMO cost model [15]. The people factors are shown in corresponding green, orange, and purple arrows in Figure 18. The productivity ranges are different, as COSYSMO is calibrated to the SE of hardware as well as software projects, but the can be used similarly for organizations to assess their current SE status and estimate likely SE productivity and affordability gains via improvements in staffing, teambuilding and performer-involved continuous process improvement. Of course, one must avoid SE cost reductions that reduce SE effectiveness, since those increase life-cycle scrap and rework. This topic will be addressed in Section 2.4 on Eliminating Scrap and Rework. Having provided a thorough discussion of the first (and arguably the most important) people option, space limitations do not permit comparably detailed discussions of the remaining seven options. However, the figures provided will provide context and quantitative relationships that will strengthen the later discussions.

Making Tasks More Efficient

A major source of affordability improvement involves investments in improved tools to improve the cost-effectiveness of the various life-cycle sources of effort such as system scoping, definition, human interface prototyping, system architecting, development, qualification, deployment, manufacturing, operations, and life cycle support, plus general tools for program

management, supply chain management, and stakeholder collaboration support. For software engineering, Figure 17 shows a productivity range of 1.50 for going from a minimal set of tools to a mature and well-integrated toolset with full life cycle coverage. A subsequent analysis confirmed that tool coverage, maturity, and integration were essential to very high tool support [16]. For SE, the corresponding productivity range for tool support in Figure 18 is 1.93.

Another source of affordability improvement is automation, in which computing and software or devices replace more expensive personnel. This has happened significantly in information processing, robots in manufacturing, and financial systems needing microsecond response times. However, there are several hazards in trusting computers to properly recognize and handle off-nominal situations, such as with several stock market crashes that caused major financial losses. And there are risks that overautomation penalizes human interaction, as with several banks which discontinued strongly automated teller support systems because they were turning the bank tellers into computer peripherals, rather than their previous roles as mini-bankers satisfying their customers.

Work and oversight streamlining generally involves undoing the effects of creeping bureaucracy, such as requiring 14 signatures to order a COTS product, or having a progress report undergo six levels of management review before it is presented to the customer. Often, performer-initiated suggestion boxes and continuous process improvement systems will help in streamlining processes.

Collaboration technology has made significant improvements in team performance across different locations, organizations, cultures and timezones. Many project support systems are still organized around individual vs. group use, causing frequent slowdowns or incompatible decisions to be made because of insufficient information sharing or lack of wideband collaboration support across interdependent groups. The COCOMO II Multi-Site Development factor in Figure 17 has a productivity range of 1.53 for variations in collaboration support across widely distributed teams. The corresponding SE Multisite Coordination factor in Figure 18 has a productivity range of 1.93.

Eliminating Tasks

Lean and agile methods have several ways to improve affordability via eliminating tasks. Lean methods emphasize elimination of non-value-adding processes such as the creeping bureaucracy examples above, or of non-value-adding products such as unnecessary documentation. Figures 17 and 18 show the relative productivity ranges for software (1.52) and SE (1.93) effects of documentation level on cost. Often, a risk-based guideline such as "If it's risky to leave it out, put it in; if it's risky to put it in, leave it out," will help make such decisions. A further approach growing in use is the workflow-oriented Kanban method of limiting work in progress and prioritizing new features described in [17].

Agile methods such as Extreme Programming have guidelines such as Simple Design, that

restrict design documentation to what has already been built [18]. However, many agile projects have found that the risk of leaving out an overall architectural definition of the system to be developed is very high as projects grow larger, such as the ThoughtWorks Lease Management project described in [19].

Agile methods also include timeboxing as a way of improving affordability by eliminating tasks. This involves having the customers prioritize their desired features, with the option of dropping low-priority features if the project is running out of time or budget, or if more valuable features need to be added during development. This has the added advantage of carrying a risk reserve in terms of lower-priority features vs. unspent funds, which are easier to take away. Eliminating tasks via automation has been discussed under tools and automation in Section 2.2.

Model-based product generation has been successfully applied in the software field, particularly for domain-specific models. It has traditionally been applied to hardware elements via numerical control systems, and its recent extension to three-dimensional printing has been identified as the beginning of a third industrial revolution by such journals as The Economist [20].

Eliminating Scrap and Rework

Most analyses of scrap and rework costs find a Pareto distribution that shows 80% of the rework costs coming from 20% of the problems. Figure 19 shows such distributions resulting from analyzing the problem reports of two TRW projects. In both cases, the main cause was the lack of attention to off-nominal use cases that turned out to be architecture-breakers: network failover for Project A and extra-long messages for Project B. Actually, the root cause for both projects was an inadequate budget, schedule, and focus on thorough architecture definition and risk resolution during the preliminary design phase.

The quantitative impact of shortfalls in architecture definition and risk resolution are also shown in Figure 17 as a function of project size. A 10 KSLOC software system will have a productivity range of 1.18, as compared to a productivity range of 1.63 for a 1000 KSLOC project. This result has been used to determine an increasingly positive return on investment in systems engineering as a function of project size in [21]. Such investments include not only the definition of a system's requirements, architecture, and plans, but also evidence that a system built to the architecture would satisfy the requirements, and be buildable within the budgets and schedules in the plans. Such evidence was not produced in Projects A and B of Figure 19.

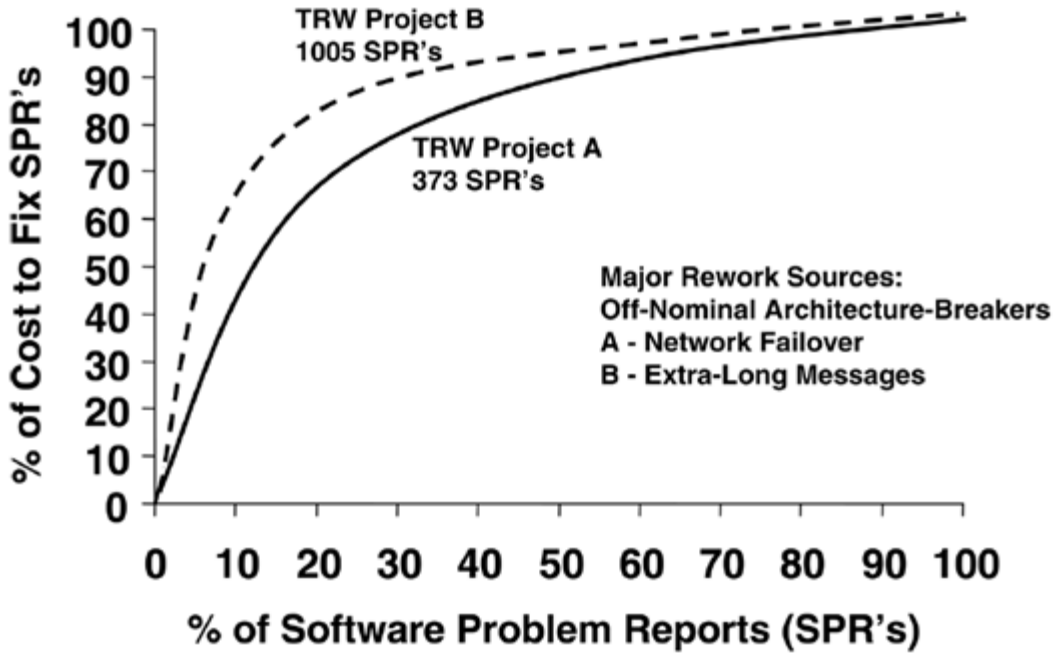


Figure 19 Pareto Distribution of Project Rework Costs

Another valuable result of analyzing the Project A and B data was the identification of the most frequent sources of change, as these can be used to modularize the architecture around the sources of change, thus reducing the costs of change effects that would otherwise ripple through the entire product [22]. These insights led to the development of an evidence and risk-driven process model and architecture framework used in the highly successful CCPDS-R project described in Appendix D of Walker Royce's book, Software Project Management [23]. It used risk-based prototyping of the user interfaces and actual development of its network operating system to provide evidence of feasibility at its PDR, which was held in month 14 rather than the usual 4 months after contract start. Its rework costs per change stayed relatively constant, compared to the usual high escalation of cost of change vs. time.

A further source of high rework costs is the premature specification of requirements before fully understanding the difficulty of achieving them. Rather than prespecify all the system's requirements, incremental and evolutionary processes enable projects to defer commitments until their implications are better understood. As with agile methods, this approach enables projects to prioritize the content of future increments based on better-understood stakeholder value.

Simplify Products: KISS (Keep It Simple, Stakeholders)

The risk-based prototyping and value-based capability prioritization enabled CCPDS-R to simplify its products, in one case reducing its size from 12 KSLOC to 5 KSLOC without reducing

essential capability. Another key approach is to use risk and evidence-based decision criteria to converge on a mutually satisfactory vs. an optimized set of performance requirements. A good example is the TRW project described in [24], whose customer specified a 1-second response time requirement to handle an extremely large workload. After finding that COTS products could not scale up to the workload and deliver a 1-second response time, the TRW engineers devised a complex custom architecture that would meet the 1-second response time, but that would cost $100 million. The customer's budget was only $30 million (maybe it's not a requirement if you can't afford it), and it was decided to prototype some usage scenarios to determine whether a 1-second response time was needed. The result was that a 4-second response time was workable 90% of the time, enabling TRW to provide a simpler $30 million COTS-based solution.

Reuse Products

A good example of investment in product line architectures and reusable components was provided by Hewlett-Packard, which in the late 1980s found that its average market lifetime for products was 2.7 years, while developing the software for the products was taking 4 years. An example of HP's experience in investing in a product line architecture and set of reusable components is provided in Figure 20. It shows that the first two 1987 projects required roughly 5 years to create the architecture and components, but that by 1992 their projects were finishing in roughly 1 year each [25].

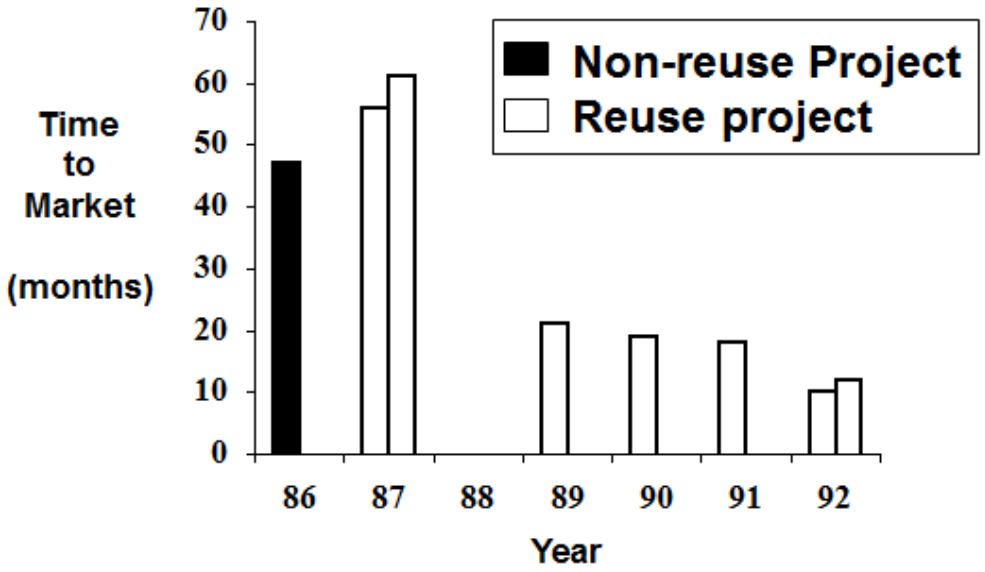


Figure 20 Hewlett-Packard Experience in Product Line Reuse

Use of COTS products vs. custom development was covered in the TRW example in the Simplify Products section. Another increasingly attractive option for reuse is the reuse of full legacy systems by repurposing them for new missions. Good examples are the DoD B-52 and C-130

aircraft, which have been repurposed to support wide ranges of new missions over time periods of over 50 years.

Reducing Operations and Support Costs

Many projects engage in short-term thinking to minimize acquisition costs, or (as often with agile methods) to rapidly produce early capabilities, but thereby end up with brittle or unscalable architectures that significantly increase life cycle costs. Table 4 summarizes data from [26] and [27] on the percentage of life cycle costs expended on operations and support. It shows the general dominance of post-deployment costs, and the need for better preparation for and execution of life-cycle support.

Table 4. Percentage of Post-Deployment Life Cycle Cost

Hardware [*26*]	Software [*27*]
• 12% -- Missiles (average) • 60% -- Ships (average) • 78% -- Aircraft (F-16) • 84% -- Ground vehicles (Bradley)	• 75-90% -- Business, Command-Control • 50-80% -- Complex platforms as above • 10-30% -- Simple embedded software

As discussed in Section 2.2, automating operational tasks can both decrease operational costs and increase operational effectiveness. However, there may be risks in taking humans out of the decision loop. Similarly, design for maintainability can improve both life cycle cost and effectiveness, as discussed in Section 2.4 on the Parnas principle, and for hardware in sources such as [28]. Streamlining supply chains for such approaches as just-in-time manufacturing can again improve both life cycle cost and effectiveness, as discussed in [29]. And given the increasing pace of change in competition, technology, marketplaces, and organizations, pro-active investments in anticipating and preparing for change are increasingly valuable for both reducing costs and increasing beefits.

Value and Architecture-Based Tradeoffs and Balancing

As shown in Figure 15, affordability can involve numerous combinations of options for decreasing life cycle costs and increasing life cycle effectiveness. Evaluating these combined options often involves complex tradeoffs among affordability and other –ilities such as reliability, availability, maintainability, usability, adaptability, interoperability, scalability, and others such as safety, security, reliance, and timeliness. For software systems, a considerable literature on ility tradeoffs has evolved, from [30] through [31,32], to the [33] Architecture Tradeooff Analysis Method. Current systems engineering approaches include real options analysis [34, 35], total cost of ownership analysis [36], incremental-commitment decision-space narrowing, [37], and physical tradespace analysis [38]. All of the approaches face significant challenges in multi-criteria decision analysis, but the need for improved capabilities continues to increase.

1.3.1.2 Conclusions

Affordability is not only about cost reduction, but also about preserving or enhancing both near-term and long-term benefits. Many cost reduction efforts focus only on a few easy ways to eliminate costs, and not only unnecessarily eliminate benefits, but also increase long-term costs. More far-seeing organizations have affordability engineering functions that pursue more long-term strategies that enable them to execute budget cuts in more cost-effective ways.

The orthogonal Life Cycle Affordability Framework, evolved over several decades of related industrial and academic research and development, provides a wider set of options for addressing both near-term and long-term costs and benefits. Cost models such as COCOMO II, COSYSMO, and hardware cost models can enable organizations to determine their current status with respect to the cost driver ratings, and to determine the potential cost savings achievable by improving their ratings. Challenges for the future primarily include the development of similar methods, models, processes and tools for estimating benefits, and for evaluating tradeoffs among affordability and other desired system ilities.

References

1. INCOSE, Systems Engineering Handbook, version 3.2.2. San Diego, CA, USA: International Council on Systems Engineering (INCOSE). INCOSE-TP-2003-002-03.2, December 2011.
2. B. Boehm, Improving Software Productivity, IEEE Computer, Volume 20, Issue 9, September 1987, pp. 43-57.
3. Mitre Corporation, MITRE Sysems Engineering Competency Model, version 1.13E, September 1, 2007, The Mitre Corporation, 2007.
4. B. Curtis et.al, The People Capability Maturity Model, Addison Wesley, 2002.
5. T. DeMarco and T. Lister, Peopleware, Dorset House, 1987.
6. K. Vitasek, M. Ledyard, and K. Manrodt, Vested Outsourcing, Palgrave Macmillan, 2010.
7. K. Vitasek, J. Crawford, N. Nyden, and K. Kawamoto, The Vested Outsourcing Manual., Palgrave Macmillan, 2011.
8. R. Fisher and W. Ury, Getting to Yes, Houghton Mifflin, 1981.
9. S. Covey, The Seven Habits of Highly Effective People, Simon and Schuster, 1989.
10. B. Boehm and R. Ross, Theory-W Software Project Management: Principles, and Examples, IEEE Trans. SW Engr., July 1989, pp. 902-916.
11. W. Dyer, Team Building, Addison Wesley, 1987.
12. B. Boehm, C. Abts, A.W. Brown, S. Chulani, B. Clark, E. Horowitz, R. Madachy, D. Reifer, and B. Steece, Software Cost Estimation with COCOMO II, Prentice Hall, 2000.
13. M. Paulk, C. Weber, B. Curtis, and M. Chrissis, The Capability Maturity Model, Addison Wesley, 1994.
14. M. Chrissis, M. Konrad, and S. Shrum, Capability Maturity Model-Integrated (CMMI), 2nd Ed., Addison Wesley, 2007.
15. R. Valerdi, The Constructive Systems Engineering Cost Model (COSYSMO), VDM Verlag Dr. Muller, 2008.
16. B. Boehm, B. Steece, J. Baik, Disaggregating and Calibrating the CASE Tool Variable in COCOMO II, IEEE Transactions on Software Engineering, Volume 28, Issue 11, November 2002, pp. 1009-1022.
17. D. Anderson, Kanban, Blue Hole Press, 2010.
18. K.. Beck, Extreme Programming Explained, Addison Wesley, 1999.

19. A. Elssamadisy and G. Schalliol, Recognizing and Responding to 'Bad Smells' in Extreme Programming, Proceedings, ICSE 2002, pp.617-622.
20. P. Markillie, A Third Indistrial Revolution,The Economist special report, April 21, 2012.
21. B. Boehm, R.Valerdi, E.Honour, The ROI of Systems Engineering: Some Quantitative Results for Software-Intensive Systems, Systems Engineering, Volume 11, Issue 3, April 2008, pp. 221-234
22. D. Parnas, Designing Software for Ease of Extension and Contraction, IEEE Trans. SW Engr., March 1979, pp. 128-137.
23. W.E. Royce, Software Project Management: A Unified Framework, Addison Wesley, 1998.
24. B. Boehm, Unifying Software Engineering and Systems Engineering, Computer, March 2000, pp. 114-116.
25. S. Kan, Metrics and Models in Software Quality Engineering, Addison Wesley, 1995.
26. Q. Redman, Weapon System Design Using Life Cycle Costs, Raytheon Presenatation, 2008.
27. J. Koskinen, Software maintenance fundamentals, in Encyclopedia of Software Engineering, P. Laplante, Ed., Taylor & Francis Group, 2009
28. B. Blanchard, D. Verma, and E. Peterson, Maintainabilty: A Key to Effective Servicability and Maintenance Management. Wiley 1995.
29. D. Blanchard, Supply Chain Management Best Practices, 2nd. Edition, John Wiley & Sons, 2010.
30. B. Boehm, J. Brown, and M. Lipow, Quantitative Evaluation of Software Quality, Proceedings, ICSE 2, pp. 592-605, October 1976.
31. T. Gilb and S. Finzi, Principles of Software Engineering Management. Addison-Wesley, 1988.
32. L. Chung, B. Nixon, E. Yu, and J. Mylopoulos, Non-Functional Requirements in Software Engineering, Kluwer, 1999.
33. P. Clements, R. Kazman, and M. Klein, Evaluating Software Architectures, Addison Wesley, 2002.
34. J. Mun, Real Options Analysis: Tools and Techniques for Valuing Strategic Investment and Decisions, 2nd Edition, Wiley Finance, 2005.
35. C. Baldwin and K. Clark, Design Rules, Vol. 1: The Power of Modularity, MIT Press, 2000.
36. B. Boehm, J.A. Lane, R. Madachy,Total Ownership Cost Models for Valuing System Flexibility, Proceedings, CSER 2011, March 2011.
37. R. de Neufville and S. Scholtes, Flexibility in Engineering Design, MIT Press, 2011.
38. A. Ross and D. Hastings, The Tradespace Exploration Paradigm, Proceedings, INCOSE 2005.

1.3.2 TIMELINESS EXAMPLE (USC)

A similar means-ends framework is provided next for sources of cycle time reduction. It can be used for assessing various mixed strategies for tailoring a systems engineering approach to a given organization's environment, culture, technology, and constraints. Its orthogonality enables the organization to compound its systems engineering calendar time savings by concurrently addressing each major source of savings. A Rapid Application Development version of the framework was provided in the SERC Systems 2020 Strategic Initiative Final Technical Report (Boehm et al. 2010). as an approach for significantly reducing calendar time for systems development. It was originally applied to rapid software application development in the CORADMO extension of the COCOMO II software cost estimation model (Boehm et al., 2000), and subsequently extended in USC work performed jointly on RT-34 (Expedited Systems Engineering) and RT-46 Phase 1 (Ingold et al., 2013).

The orthogonal framework is developed in the context of systems engineering as an activity network of tasks with backtracking, as illustrated in Figure 21.

This model is a bit oversimplified, given that the real world has partial dependencies and more complex constraints, but these do not cause major complications with respect to the use of the model to identify sources of calendar time reduction.

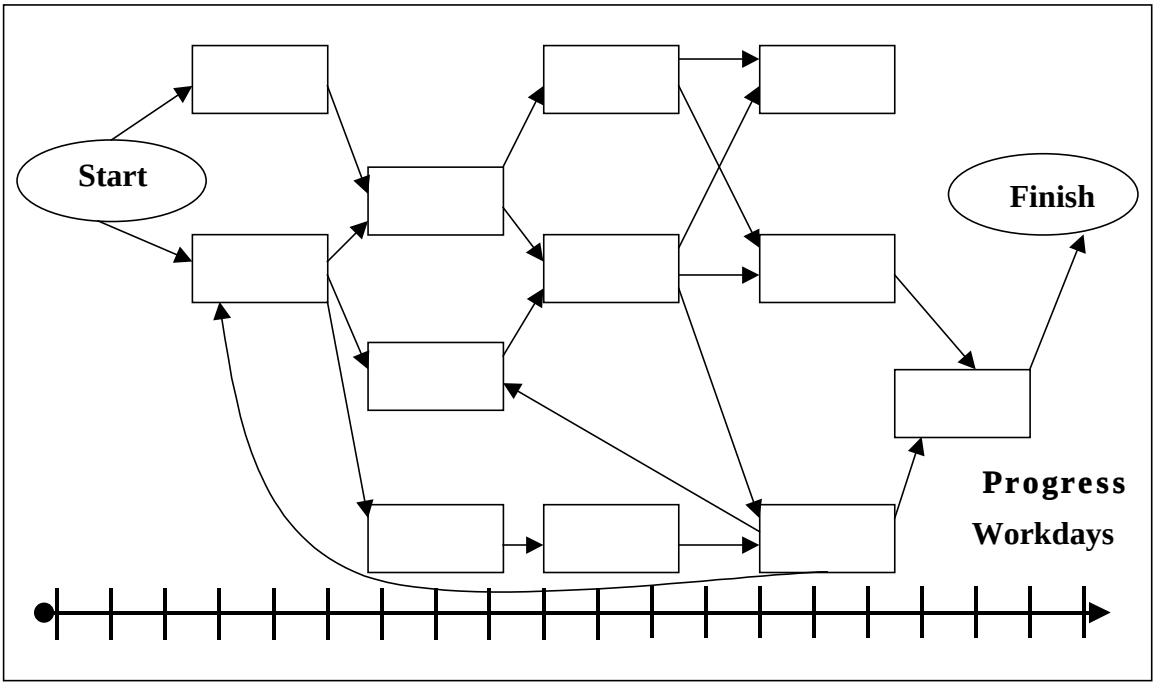


Figure 21 Activity Network with Backtracking

With respect to Figure 21, the SE Acceleration Opportunity Tree of sources of calendar time reduction is presented in Figure 22. Each major source is elaborated below.

Eliminating tasks

Business process re-engineering or lean thinking (Womack-Jones, 2003) can discover and eliminate non value-adding tasks, such as unnecessary coordination cycles, purchase approval signatures, or change control boards operating at too low a level. Reusing systems engineering assets, model-based systems engineering (MBSE) capabilities, and automated SE artifact generation have been shown in the COSYSMO 2.0 systems engineering effort estimation model (Fortune, 2009) to eliminate significant sources of SE effort. For example, the COSYSMO 2.0 reuse-related percentages determined by a combination of expert judgment and data analysis are an added 38% effort to design SE artifacts for reuse, and a savings of 35% for reuse with modification, 57% for reuse without modification but a need for testing, and 85% for reuse without modification or testing. Not all of this saved effort will be on the critical path, but much of it will be. However reuse will require up-front investment in domain engineering and structuring SE artifacts for reuse.

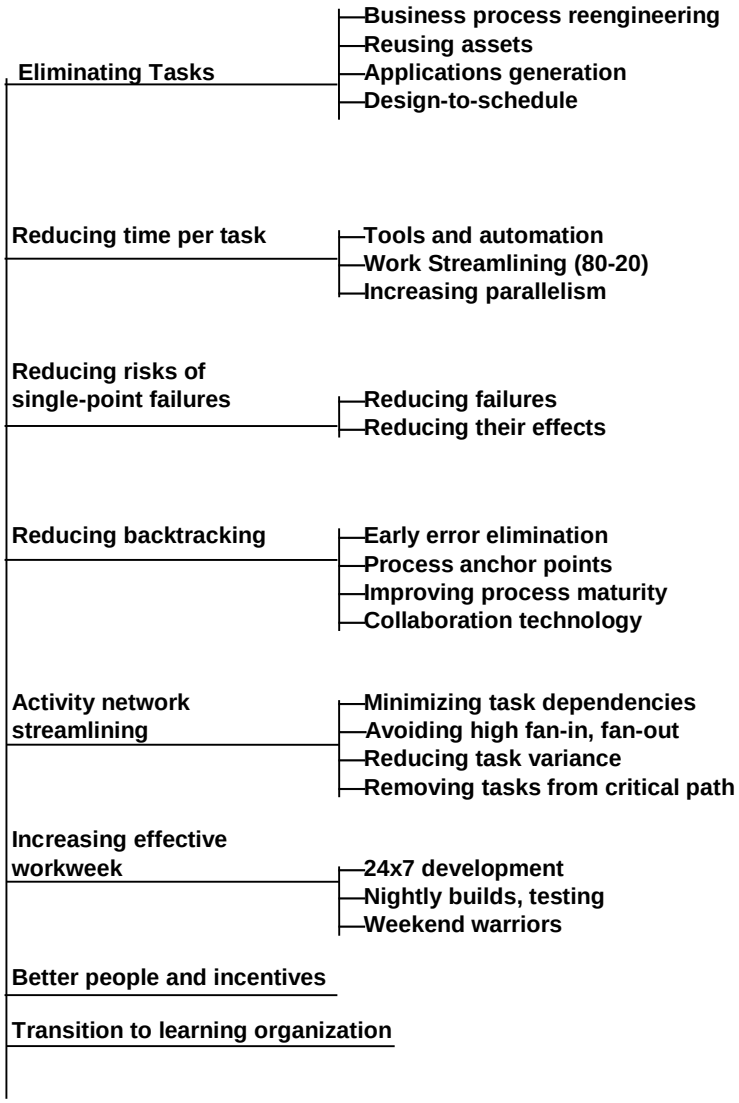


Figure 22. SE Acceleration Opportunity Tree

Evolutionary definition avoids definition of details best left to downstream increments, such as the details of decision aids for complex command and control systems. However, it is important to devote early effort to ensuring that the system's architecture will accommodate anticipated sources of system change and growth, thus avoiding easiest-first "mashup" early increments that foreclose options for growth and build up technical debt (Boehm-Bhuta, 2008).

Reducing time per task

Reducing time per task can be addressed through technology or management. Tools, models, and automation can accelerate SE tasks, as highlighted in the ODDR&E Rapid Capability Fielding Tools Study (Carlini et al., 2010). SE acceleration tools identified in the study included visual modeling, rapid prototyping, modeling and simulation, model-based artifact generation, architecture-based attribute and tradeoff analysis, and integrated SE environments with tool interoperability. Some good earlier sources for reducing time per task in the software area are (Arthur, 1992) and (McConnell, 1996).

On the management side, Pareto 80-20 analysis can be effective for work streamlining. For example, if 20% of the tasks cause 80% of the time delays, then task streamlining should be focused onto that 20% (e.g., defining and designing for critical off-nominal scenarios). Particularly for systems of systems, one way to increase effective parallelism in developing a number of components is to ensure precise, well-validated component interface specifications in advance of detailed component design. Then, the design and development effort for each component can proceed in parallel with minimal delays due to interface reconciliation or cross-component ripple effects. Often, management will try to save time by bringing the contributing-system providers aboard quickly without such interface specifications, but quantitative analyses have shown that such savings are eventually much more than wiped out by late rework (Boehm et al., 2004).

Avoiding single-point task failures

System development projects are sometimes prone to single point failures, which can negatively impact completion schedules. For example, SE support environments can go down or fail at untimely moments (a.k.a., "Murphy's Law"). Similarly, key project personnel such as lead system architects may leave the company or be pulled off to save another project. The basic way to reduce the risk of single point failures is to reduce both the probability of failure (e.g., by providing bonuses for staying with the project through completion) and the impact of failure (e.g., by spreading knowledge via task-sharing and peer reviews). The amount of risk exposure (Probability of failure times Impact of failure) is a good way to prioritize failure risk-avoidance efforts such as reducing the effects of personnel turnover.

Reducing backtracking

Rework is perhaps the most common form of time-sink that system development projects experience. Generally, rework does not add value. Thus, a major challenge and opportunity involves minimizing its occurrence and impact. Early and continuous SE verification and validation (V&V) via automated analysis tools, modeling and simulation, prototyping, and independent reviews catch SE defects earlier when they are easier and quicker to fix. Evidence-based decision milestones provide a management framework both to synchronize and stabilize concurrent SE effort and to generate and review the evidence that the proposed SE solutions will enable satisfaction of the requirements within the budgets and schedules of the plan.

Shortfalls in evidence translate into uncertainties and risks, which frequently translate into costly and time-consuming defects.

Investments in process maturity have been shown to reduce the incidence and negative impact of defects (Goldenson-Gibson, 2003). The CMMI process areas of Validation, Verification emphasize early defect identification and removal; and Causal Analysis and Resolution emphasizes root cause analysis to reduce future sources of defects. Finally, rework can be reduced by tightening convergence loops or by articulating where progress disconnects occur. For example, using collocation and virtual collaboration technology can reduce or surface misunderstandings among SEs, system stakeholders, and independent reviewers, and can avoid bad fixes through interactive discussions.

Activity network streamlining

As displayed in Figure 21 above, project activity networks may reveal many possible paths to project completion. PERT/CPM tools and techniques may help identify critical paths in workflow, resource dependency, or schedule. When activity networks get too "bushy" (when certain activities have a high fan-in of input paths or fan-out of output paths), then bottlenecks can occur.

For example, many early project review procedures require mixes of SE artifacts and numerous other derivative artifacts such as plans for system installation, data migration, training, cutover, maintenance, operations, and support to come together at a single formal review. This frequently requires SEs to spend much of their critical path time in tutorial discussions of the emerging system architecture, when they need to focus on getting the architecture ready. It is better to schedule less-formal SE technical reviews in advance of the large, many-artifact formal reviews. Some early work on the derivative artifacts can be done in advance, providing useful context for the SEs, but the main work on the derivative artifacts can then be done once based on a well-vetted architecture.

Some other ways to get time-consuming tasks off the critical path include task decomposition and parallelization, or through network reconfiguration. Examples include pre-positioning of facilities, components, tools, experts, or data, which may add somewhat to the cost but be worth it in schedule savings. A good example is "overinvestment" in reusable components as discussed under Eliminating Tasks.

Increasing the effective workweek

This does not mean getting SEs to work 80-hour weeks, which generally leads to staff burnout and untimely turnover. However, if SE work can be done in different shifts or time zones, it is possible to increase the effective workweek. This usually requires some amount of up-front investment in creating a shared product vision, establishing the ground rules for inter-group

collaboration, and ensuring consistent technical decision-making in order to succeed. Even then, it is best to have the activities be complementary, such as testing, interface checking, or other forms of independent V&V.

Better people and incentives

Clearly, every project would like its SE team and its Integrated Product Teams (IPTs) of stakeholders to consist of the best people possible. Often, though, "best" is interpreted as "technically strongest," which often leads to continuing arguments among polarized experts and slow progress. An excellent set of criteria for "best" SE and IPT team members is provided in Air Force Instruction 63-123, Evolutionary Acquisition for C2 Systems (AirForce, 2000):

- *empowered* – have the authority to negotiate for the organizations that they represent;
- *committed* – provide continuous representation of their constituencies and ensure performance of actions necessary to achieving group objectives;
- *representative* – represent their entire constituency, not just a part or their personal positions;
- *knowledgeable* – be sufficiently aware of group objectives and have organizational, technical, and management expertise to ensure informed and effective collaboration; and
- *collaborative* – operate as team players and work toward win-win solutions for all stakeholders.

As discussed under *Avoiding single-point task failures*, a critical success factor involves establishing incentives to attract and retain the best SE and IPT team members. These can include bonuses for staying with the project through completion (for evolutionary development, one does not dismiss the SEs after the first PDR), recognition of their key contributions, and SE career path progression.

Transition to learning organizations

A true CMMI Level 5 SE organization has accomplished the transition to a learning organization via the Organizational Innovation and Deployment process area. Learning organizations can do more than optimize and manage their processes. They have instead cultivated a culture of continuous improvement and process redesign as routine activities, rather than as uncommon events. They balance "skating to where the puck has been" process improvement via root cause analysis and improvement over previous projects, with "skating to where the puck is going" efforts to anticipate, monitor, and prepare for future trends.

A good example is in applying trends in agile methods to SE. The key to agility in complex systems is for the participants to be able to operate via tacit interpersonal knowledge and interpersonal trust (Nonaka, 1991; Nonaka-vonKrogh, 2009), as compared to basing collaboration on explicit documented knowledge among participants unfamiliar with working

with one another. This implies not only developing powerful virtual collaboration capabilities, but also involving the participants in realistic collaboration exercises that build up tacit knowledge, mutual understanding, and trust. Other good approaches for transforming SE into a learning organization are contained in (Wade, 2010).

Conclusions

Versions of the SE Acceleration means-ends framework have been and can be used as checklists for formulating SE and system development acceleration initiatives; a framework for reviewing SE and system development acceleration plans; and a balanced-scorecard framework for monitoring progress with respect to plans. It frequently opens up unconsidered avenues for accelerating project schedules. Its orthogonality enables organizations to compound their systems engineering calendar time savings by concurrently addressing each major source of savings.

References

(Air Force, 2000). Air Force Instruction 63-123, Evolutionary Acquisition for C2 Systems, 1 January 2000, page 25.

(Arthur, 1992). L Arthur, Rapid Evolutionary Development, Wiley 1992.

(Boehm et al. 2000). Boehm, B. et al., Software Cost Estimation with COCOMO II, Prentice Hall, 2000.

(Boehm et al., 2004). B. Boehm, A. W. Brown, V. Basili, and R. Turner, "Spiral Acquisition of Software-Intensive Systems of Systems," CrossTalk, May 2004.

(Boehm-Bhuta, 2008). Boehm, B., and Bhuta, J., "Balancing Opportunities and Risks in Component-Based Software Development," IEEE Software, November-December 2008, Volume 15, Issue 6, pp. 56-63.

(Boehm et al. 2010). B. Boehm, J. Bayuk, A. Desmukh, R. Graybill, J. Lane, A. Levin, A. Madni, M. McGrath, A. Pyster, S. Tarchalski, R. Turner, and J. Wade, Systems 2020 Strategic Initiative, Final Technical Report SERC-2010-TR-009, August 29, 2010.

(Carlini et al.,2010). Carlini, J. et al., Rapid Capability Fielding Tools Study, ODDR&E Report, March 2010.

(Fortune, 2009). J. Fortune, "Estimating Systems Engineering Reuse with the Constructive Systems Engineering Cost Model (COSYSMO 2.0)," PhD Dissertation,Department of Industrial and Systems Engineering, University of Southern California, December 2009.

(Goldenson-Gibson, 2003). D. Goldenson and D. Gibson, Demonstrating the Impact and Benefits of CMMI®: An Update and Preliminary Results, Special Report CMU/SEI-2003-SR-009, October 2003.

(Ingold et al, 2013). D. Ingold, B. Boehm, J. Lane, and S. Koolmanojwong, “A Model for Estimating Agile Project Process and Schedule Acceleration,” Proceedings, ICSSP 2013, May 2013.

(Lane et al. 2010). J. Lane, B. Boehm, M. Bolas, A. Madni, and R. Turner, "Critical Success Factors for Rapid, Innovative Solutions," Proceedings, ICSP 2010.
(McConnell, 1996). S. McConnell, Rapid Development, Microsoft Press, 1996.
(Nonaka, 1991). I. Nonaka, "The knowledge creating company". *Harvard Business Review* **69** (6 Nov–Dec): 96–104.
(Nonaka-vonKrogh, 2009). I. Nonaka, amd G.von Krogh, "Tacit Knowledge and Knowledge Conversion: Controversy and Advancement in Organizational Knowledge Creation Theory". *Organization Science* **20** (3): 635–652, 2009.
(Wade et al, 2010). J. Wade, A. Madni, Neill, C., Cloutier, R., Turner, R., Korfiatis, P., Carrigy, A., Boehm, B., Tarchalski, S., "Development of 3-Year Roadmap to Transform the Discipline of Systems Engineering." Final Technical Report – SERC-2009-TR-006, March 2010.
(Womack-Jones, 2003). J. Womack and D. Jones,. *Lean Thinking*. Free Press, 2003.

1.4 Domain-Oriented Views

1.4.1 Overview: Details Provided in Section 2

Particular domains will have aspects that help in setting ility priorities, and also in simplifying ility tradspaces. For example, space systems have a very high priority on Reliability, as it is generally uneconomic to access them to get them started again. But for the same reason, they do not need to be concerned with tradeoffs between Maintainability and other ilities, such as being designed to be easy to access and replace faulty components. Sections 2.2 through 2.6 describe several domain-specific approaches pursued in RT-46 Phase 1. Section 2.2.1 describes how the Georgia Tech FACT system draws on ground vehicle knowledge to enable rapid development of ground vehicle ility tradespace analyses. Section 2.2 describes a similar approach for ground vehicles developed and refined during Phase 1 by Wayne State. Section 2.3 summarizes Phase 1 work by NPS and Wayne State in preparation for prospective Phase 2 ility tradespace analysis in the ship domain. Section 2.4 summarizes the use of domain knowledge in the space domain in exploring satellite-vehicle design options using the Epoch-Era approach; its presentation at ERDC has stimulated interest in a Phase 2 effort by MIT to similarly address the logistics supply chain domain. Section 2.5 summarizes exploratory work done by USC in concert with Aerospace Corporation and USAF/SMC in identifying sources of total ownership cost for full space-oriented systems, including satellite bus and payload elements, ground system elements, and launch system elements, as an exploratory effort toward a potential RT-46 Phase 2 task.

1.5 System of Systems and Enterprise Participation Views

Increasingly, DoD systems will evolve away from being self-contained entities and toward receiving an increasing amount of their capability from their participation in systems of systems and enterprises. System of systems (SoS) and enterprise participation views are views that include multiple, interoperating systems that provide cross-cutting mission capabilities and organizational business functions, in which single system missions and functions participate. Some of the key ilities for these views that can have significant impact on costs and affordability are interoperability; constituent system flexibility, adaptability, and robustness; balanced with mission capability performance as captured by accuracy and precision, speed/velocity/response time, and usability (ease of use).

For the purposes of this technical report, we define systems of systems, enterprise systems, and product lines using definitions from the *Guide to the Systems Engineering Body of Knowledge (SEBoK)*, version 1.1 (http://www.sebokwiki.org/1.1/index.php?title=Main_Page):

> Systems of systems (SEBoK SoS Definition): "a*n assemblage of components which individually may be regarded as systems, and which possess two additional properties:*
>
> > *(a) Operational Independence of the Components: If the system-of-systems is disassembled into its component systems the component systems must be able to usefully operate independently. That is, the components fulfill customer-operator purposes on their own.*
> >
> > *(b) Managerial Independence of the Components: The component systems not only can operate independently, they do operate independently. The component systems are separately acquired and integrated but maintain a continuing operational existence independent of the system-of-systems.* (Maier 1998, 267-284)"
>
> Enterprise (SEBoK Enterprise Definition): "*one or more organizations sharing a definite mission, goals, and objectives to offer an output such as a product or service"*

A key to an enterprise's cost-effectiveness is its ability to identify the commonalities and variabilities across its products and services, and to organize the areas of commonality into portfolios or product lines. By investing in domain architectures in which the commonalities are developed and evolved once, with standard interfaces to the sources of variability, the enterprise can enjoy significant savings in cost and schedule during each system's development, and even greater savings and responsiveness during its operational lifetime. Product lines and their contribution to Affordability and Timeliness are discussed next, followed by a discussion of systems' participation in systems of systems and their resulting need for Interoperability, Flexibility, Adaptability, and Robustness.

1.5.1 PRODUCT LINES (NPS)

A product line approach provides multiple benefits with respect to ilities across all DoD domains. Affordability gains accrue from reusing common pieces in different systems/products that share features. Furthermore, systems can be fielded faster leading to increased overall mission effectiveness. Flexibility is enhanced increasing the option space. These benefits occur because previously built components reduce the effort and enable more rapid development.

For example, the Navy and Marine Corps adopted Naval Open Architecture (NOA) to reduce the rising cost of warfare systems and platforms while continuing to increase capability delivery on shortened demand timelines (DoD 2010). NOA employs business and technical practices to create modular, interoperable systems that adhere to open standards with published interfaces. This approach significantly increases opportunities for innovation and competition, enables reuse of components, facilitates rapid technology insertion, and reduces maintenance constraints.

Composeable systems allow for selecting and assembling components in different ways to meet user requirements. In order for a system to be composeable its components must also be reusable, interoperable, extensible, and modular.

A reusable artifact as one that provides a capability that can be used in multiple contexts. Reuse is not confined to a software component but any lifecycle artifact including training, documentation, and configuration. NOA is concerned with artifacts which relate to the design, construction, and configuration of a component.

Efficient product line architecting requires modularization of the system's architecture around its most frequent sources of change (Parnas 1979) as a key principle for affordability. This is because when changes are needed, their side effects are contained in a single systems element, rather than rippling across the entire system.

For modularization it is desirable to identify the commonalities and variability across the families of products or product lines, and develop architectures for creating (and evolving) the common elements once with plug-compatible interfaces for inserting the variable elements (Boehm, Lane, and Madachy 2010).
Efforts such as the Navy's IWS Product Line Approach for Surface Combat Systems are addressing these product line architecture technical and governance issues. A depiction of their Product Line Common Asset Library is shown in from (Emory 2010) for selected ship applications.

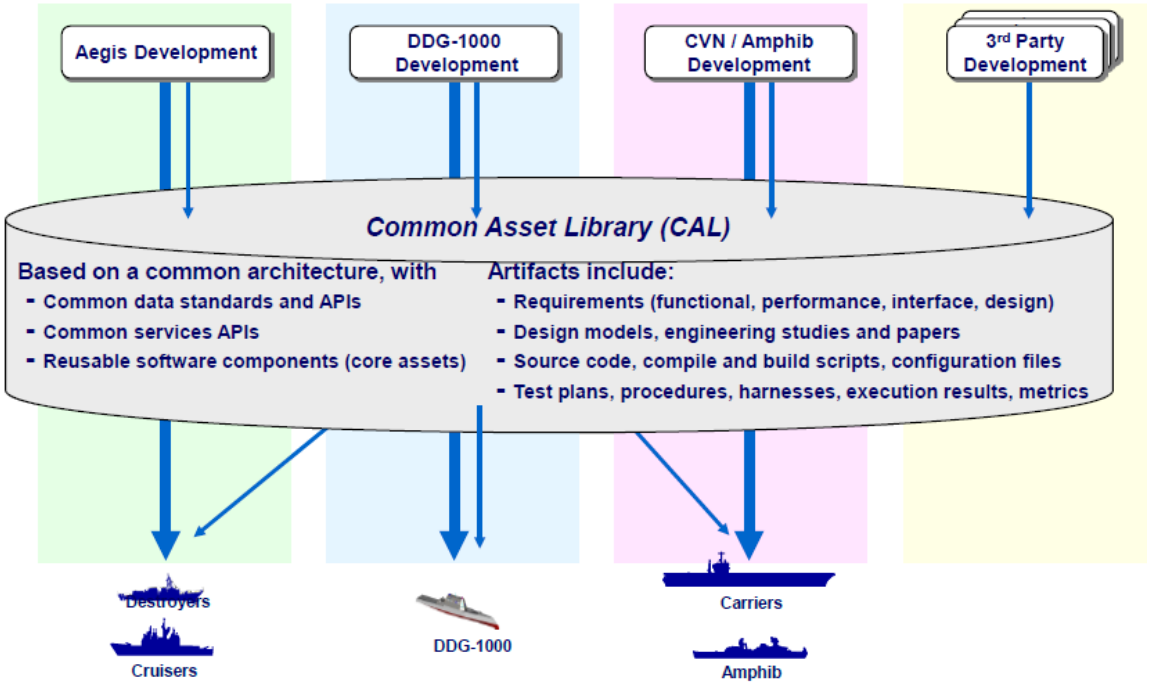


Figure 23: Surface Combat Systems Product Line Common Asset Library

The Navy's Surface Navy Combat Systems Software Product Line Architecture is defined in the Architecture Description Document (ADD) (PEO IWS 2009). It provides guidance for domain requirements and functional analyses across domains. System functional architectures must satisfy their own requirements while remaining in alignment with the ADD in order to successfully achieve commonality.

An example of establishing common product line requirements by applying the domains defined in the Navy's ADD is shown in **Figure 24** from [Shuttleworth et al. 2010]. This shortened example shows some domains, mission areas and non-function al ilities as attributes for sorting requirements to achieve commonality.

Domain	Mission Area	Nonfunctional
External Communications	Ballistic Missile Defense	Survivability
Display	Antiair Warfare (AAW)	Information Assurance
Vehicle Control	Surface Warfare (SUW)	Safety
Weapon Management	Undersea Warfare	Mobility
Sensor Management	Strike	Reliability
Track Management	Information Operations	Maintainability
Combat Control	Antiterrorism/ Force Protection	Availability

Figure 24: Example Navy Architecture Domain, Mission Area and Ilities

Relevant MPT frameworks for assessing product line aspects are described next. These parametric approaches determine the TOC for various levels of investment in product line architecting. The investment effort is the analysis of the commonalities and variabilities across a product line of similar systems, and building in flexibility to enable reuse or easy adaptation of common components, and plug-compatible interfaces for the variable components.

Product Line Modeling for Affordability and Ility Trades

The Constructive Product Line Investment Model (COPLIMO) is used to assess the costs, savings, and return on investment (ROI) associated with developing and reusing software product line assets across families of similar applications [Boehm et al., 2004]. COPLIMO is based on the well-calibrated COCOMO II model [Boehm et al., 2000] with 161 data points.
It includes parameters which are relatively easy to estimate early and be refined as further information becomes available. One can perform sensitivity analyses with the model to see how the ROI changes with different parameters.
Most product line cost models focus on development savings, and underestimate the savings in Total Ownership Costs (TOC). COPLIMO consists of a product line development cost model and an annualized post-development life cycle extension to cover full lifecycle costs. It models the portions of software that involve product-specific newly-built software, fully reused black-box product line components, and product line components that are reused with adaptation.
More elaborate versions of COPLIMO include additional reuse parameters while covering software maintenance as well as development. Additional features such as present-value discounting of future savings and Monte Carlo probability distributions have been added.
The COPLIMO framework has been instantiated and extended at the systems level, used to assess flexibility and ROI tradeoffs. Some of these extensions and applications are described next.

TOC Models for Valuing Product Line Flexibility

The following approaches extend COPLIMO for a TOC analysis for a family of systems. The value of investing in product-line flexibility using Return-On-Investment (ROI) and TOC is assessed with parametric models adapted from the basic COPLIMO model. The models are implemented in separate tools available to all SERC collaborators:

- System-level product line flexibility investment model.
- Software product line flexibility investment model. The detailed software model includes schedule time with NPV calculations.
-

Figure 25 shows the inputs and outputs for the system-level product line model.

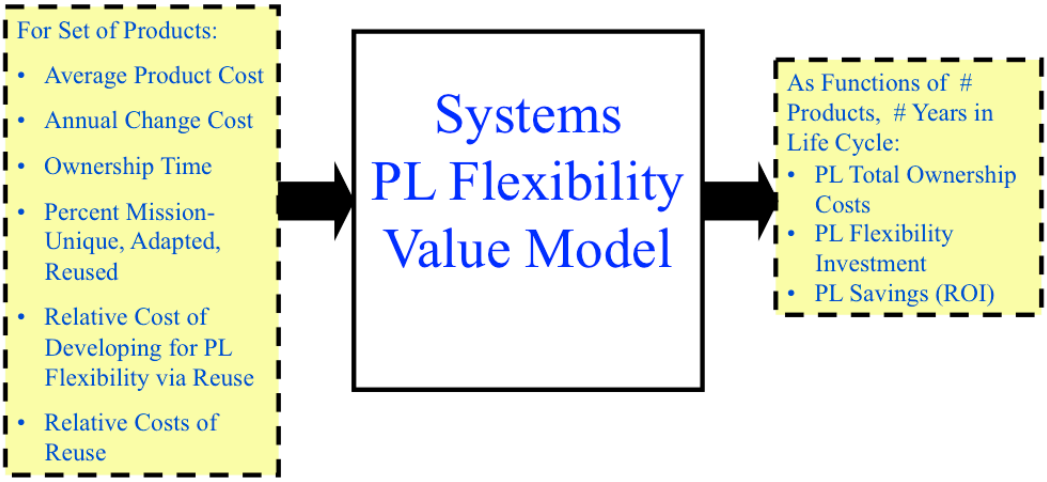


Figure 25: Systems product line flexibility value model (TOC-PL).

The cost of the first system is determined by multiplying the average product cost by the fraction of the product to be developed for reuse, (%Adapted + %Reused)/100, multiplying that by the relative cost of developing for product line flexibility reuse, and adding that to the system-unique cost (%Unique * Average Product Cost / 100) which does not have to be developed for reuse. For subsequent products, the cost of the unique system portion is the same, but the equivalent costs of adapted and reused portions are determined by their relative costs of reuse. For hardware, the relative costs of reuse should include not only the cost of adapting the reused components, but also the carrying costs of the inventory of reusable components kept in stock.

The net effort savings for the product line are the cost of developing separate products (#Products*Average Product Cost) minus the total cost of developing Product 1 for reuse plus developing the rest of the products with reuse. The ROI for a system family is the net effort savings divided by the product line flexibility investment, (Average Product Cost) * (%Adapted + %Reused) * (Relative Cost of Reuse + Carrying Cost Fraction – 1)/100. The TOC is computed for the total lifespan of the systems and normalized to net present value at specified interest rates. The example shown below represents a family of seven related systems with three-year ownership durations. It is assumed annual changes are 10% of the development cost. Within the family of systems, each is comprised of 40% unique functionality, 30% adapted from the product line and 30% reused as-is without changes. Their relative costs are 40% for adapted functionality and 5% for reused. The up-front investment cost in flexibility of 1.7 represents 70% additional effort compared to not developing for flexibility across multiple systems. **Figure 26** shows the consolidated TOC and ROI outputs.

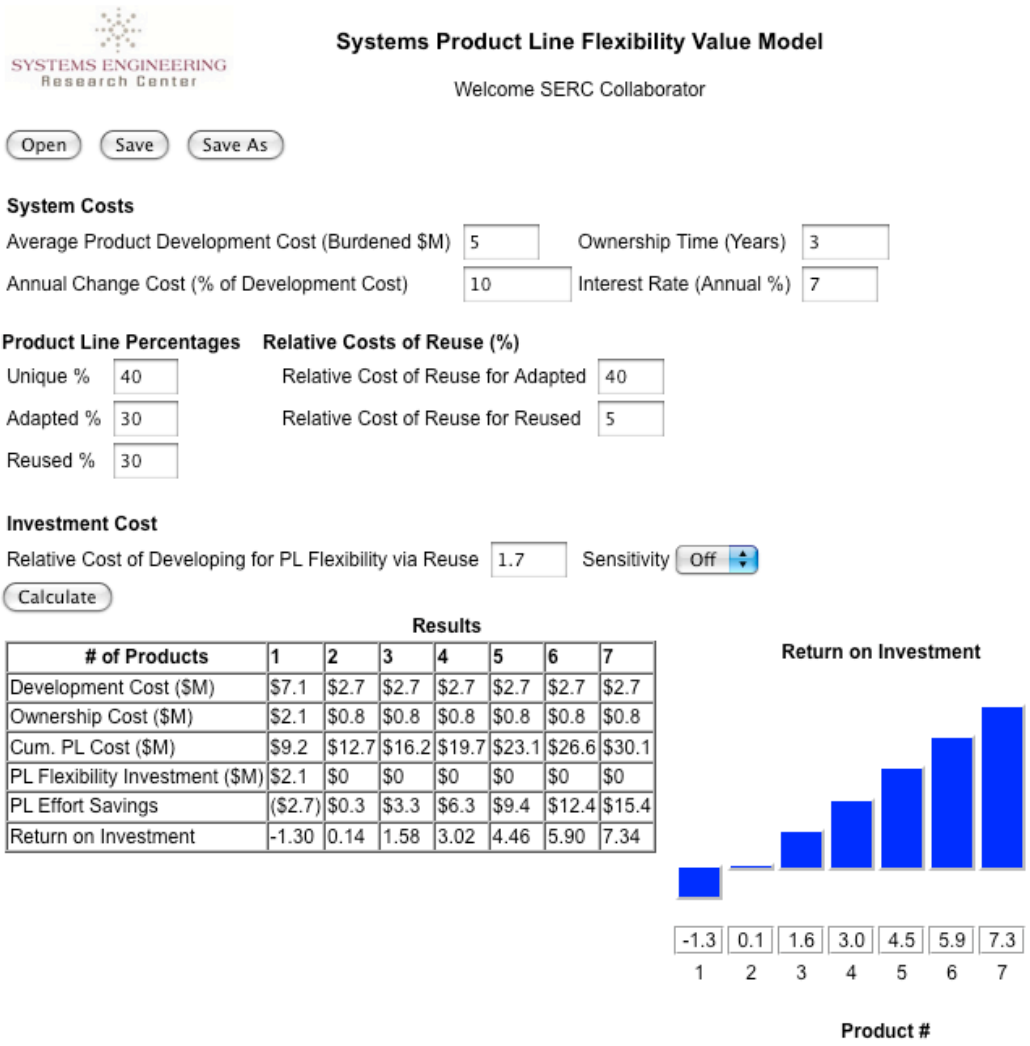


Figure 26: Product line flexibility TOC and ROI results.

However, it is desired to evaluate ranges of options and assess the sensitivity of TOC. The tools allow for a range of relative costs as shown in **Figure 27** for sensitivity runs. The results show that the model can help projects determine "how much product line investment is enough" for their particular situation. In the **Figure 27** situation, the best level of investment in developing for reuse is an added 60%.

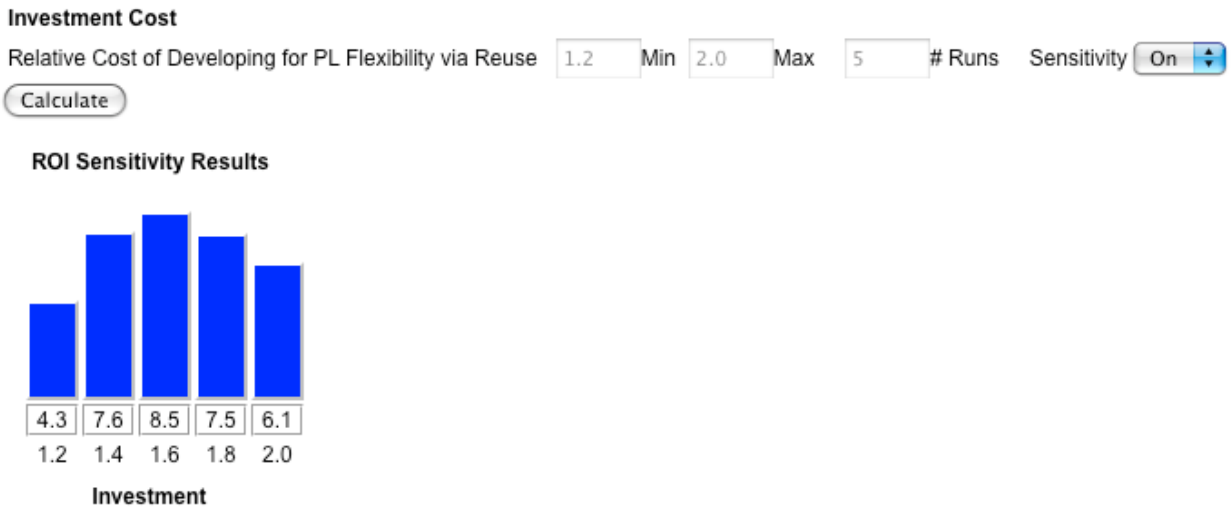


Figure 27: Example sensitivity analysis (ROI only).

Other types of sensitivity analyses can be conducted. **Figure 28** shows example results of assessing the sensitivity of TOC across a range of product ownership durations.

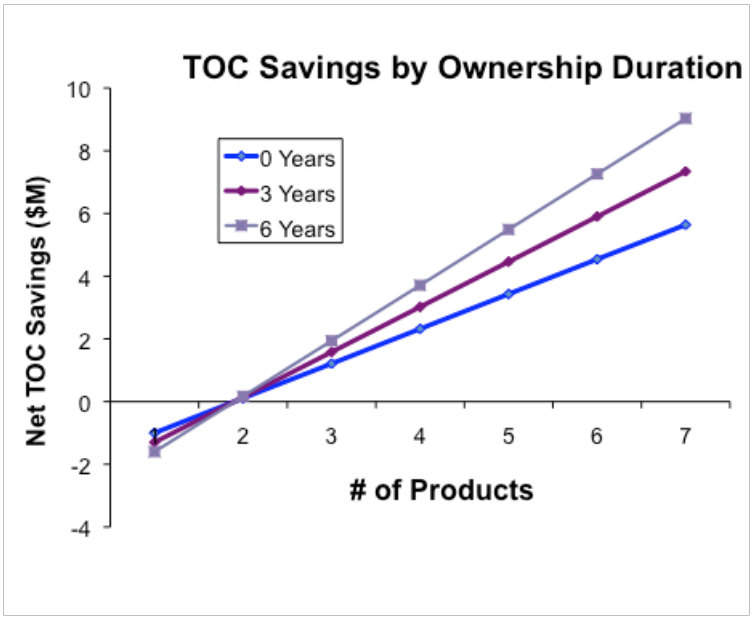


Figure 28: TOC-PL sensitivity by ownership duration results.

The TOC-PL model can also be used in an acquisition decision situation to show that if a project proposes a stovepipe single-product point solution in an area having numerous similar products, and has not done an analysis of the alternative of investing in a product line approach, the project's TOC will represent a significantly higher cost to DoD and the taxpayers.
The general model was enhanced to handle specific DoD application domains, and added initial Monte Carlo simulation capabilities. It incorporates the life cycle cost ratios for Operations and Support (O&S) for hardware O&S cost distributions were derived from [Redman et al., 2008] and software from [Koskinen 2010].

Setting the life cycle cost ratios as a function of system type in the tables impacts the general TOC Product Line model inputs for Ownership Time and Annual Change Cost. The user chooses a system type and ownership time, which invokes a calculated annual change costs for the relevant domain.

The next example illustrates a domain-specific analysis for a missile system with a demonstration of Monte Carlo simulation. The initial case study was for a general system, but in this scenario the user specifies a missile system for O&S life cycle cost defaults.
A missile product line development with three year ownership time is being evaluated. The user chooses the Missile System Type, and sets Ownership Time to 3 years. With these inputs, the pre-calculated Annual Change Cost = 12%/3 years = 4%. The results are in **Figure 29**.
Shown also are the optional Monte Carlo results from varying the relative cost of developing for flexibility. The means are listed with the ROI distribution graph. All input parameters are open to variation for more sophisticated Monte Carlo analysis in follow-on work, per the next section on proposed next steps.

Open Sa

Aircraft
Ground Vehicle
✓ Missile
Ship

System Type

System Costs

Average Product Development Cost (Burdened $M) 5 Ownership Time (Years) 3

Annual Change Cost (% of Development Cost) 4 Interest Rate (Annual %) 7

Product Line Percentages **Relative Costs of Reuse (%)**

Unique % 40 Relative Cost of Reuse for Adapted 40

Adapted % 30 Relative Cost of Reuse for Reused 5

Reused % 30

Investment Cost

Relative Cost of Developing for PL Flexibility via Reuse 1.7

Calculate Monte Carlo On

Means

# of Products	1	2	3	4	5	6	7
Development Cost ($M)	$7.1	$2.7	$2.7	$2.7	$2.7	$2.7	$2.7
Ownership Cost ($M)	$0.9	$0.3	$0.3	$0.3	$0.3	$0.3	$0.3
Cum. PL Cost ($M)	$8.0	$10.9	$13.9	$16.9	$19.9	$22.9	$25.9
PL Flexibility Investment ($M)	$2.1	$0	$0	$0	$0	$0	$0
PL Effort Savings	($2.4)	$0.3	$2.9	$5.5	$8.1	$10.7	$13.3
Return on Investment	-1.12	0.12	1.36	2.60	3.84	5.08	6.32

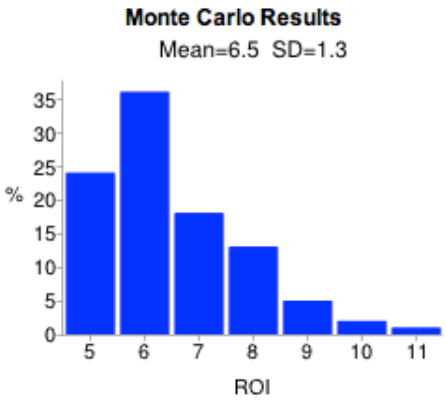


Figure 29: DoD application domain and Monte Carlo TOC-PL results.

Summary

The TOC system product line models provide strong capabilities for analyzing alternative approaches to system acquisition and the effects on TOC. They show that if total life cycle costs are considered for development and maintenance, product lines can have a considerably larger payoff, as there is a smaller base to undergo corrective, adaptive, and perfective maintenance.

There are other significant product line benefits besides life cycle cost savings, such as rapid development time and adaptability to mission changes. The models provide an easy-to-use framework for performing these broader ility and affordability analyses.

The models also demonstrate that not all attempts at product line reuse will generate large savings. A good deal of domain engineering needs to be done well to identify product line portions of the most likely to be product-specific, fully reusable, or reusable with adaptation.

Much product line architecting needs to be done well to effectively encapsulate the sources of product line variation.

Extensions can be added including the effects of varying product sizes, change rates, product line investment costs, and degrees of reuse across the products in the product line. The models could be combined with other complementary models involving real options, risk assessments, or tradeoffs among flexibility aspects such as evolvability, interoperability, portability, or reconfigurability; or between flexibility aspects and other –ilities such as security, safety, performance, reliability, and availability.

References

Program Executive Office Integrated Warfare Systems (PEO IWS), *Surface Navy Combat Systems Software Product Line Architecture*, Architecture Description Document (ADD), Version 1.0, 31 July 2009.

Department of Defense, Naval Open Architecture Enterprise Team, *Naval Open Architecture Contract Guidebook for Program Managers, version 2.0,* June 2010, available at http://www.dtic.mil/docs/citations/ADA550060

Boehm, B., C. Abts, A.W. Brown, S. Chulani, B. Clark, E. Horowitz, R. Madachy, D. Reifer, & B. Steece (2000). *Software Cost Estimation with COCOMO II*, Prentice Hall.

Boehm, B., A. W. Brown, R. Madachy, & Y. Yang, A Software Product Line Life Cycle Cost Estimation Model. *Proceedings of the 2004 International Symposium on Empirical Software Engineering*, ISESE'04, August 19-20 2004, pp. 156-164.

Q. Redman, A. T. Crepea, G. Stratton, (2008). "Weapon Design Trade Off Using Life Cycle Costs", Raytheon Corporation, URL: http://www.galorath.com/images/uploads/3_-_Quentin_Redman_-_Design_trades_using_Life_Cycle_Costs_short_version.pdf

Koskinen, Jussi , (2010). "Software Maintenance Costs", Jyväskylä: University of Jyväskylä. URL: http://users.jyu.fi/~koskinen/smcosts.htm

Boehm, B., J. Lane, and R. Madachy. 2010. "Valuing System Flexibility via Total Ownership Cost Analysis." Proceedings of the NDIA SE Conference, October 2010, San Diego, CA, USA.

Parnas, D. L. 1979. "Designing Software for Ease of Extension and Contraction." IEEE Transactions on Software Engineering SE-5(2): 128-138. Los Alamitos, California, USA: IEEE Computer Society.

Emery K., "Surface Navy Combat Systems Engineering Strategy", IWS OA Briefing to NPS, Monterey, CA, March 2010, available at http://www.nps.edu/Academics/Institutes/Meyer/docs/IWS%20OA%20Briefing%20to%20NPS%20March0410.pdf

K. Shuttleworth, D. Kolodgie, and G. Albertson, "A Systems Engineering Approach to Commonality across Surface Ship Combat Systems (Requirements and Architecture)", Proceedings of Engineering the Total Ship (ETS) Symposium, Falls Church, VA, July 2010

1.5.2 PARTICIPATION IN SYSTEMS OF SYSTEMS (USC)

The "participation in system of systems (SoS)" view is important in the affordability equations because, as stated in the DoD Systems Engineering Guide to Systems of Systems, "…with the advent of networks and increased efforts to link systems for information sharing across the battle space, most systems are part of virtual SoS. " This means that many capabilities are performed not by a single system, but by a set of of interconnected, interoperating systems and that the expected value of a given system depends on its interoperability with other existing systems as well as its flexibility and adaptability to interoperate with new systems that are deployed in order to support new capabilities.

Overview of System of System and Enterprise Environment and Tradespace

When viewing SoS, one finds that many of the cross-cutting capabilities of interest are software-intensive and support communications and information sharing related to military situational awareness, decision making, targeting, and platform operation. SoS may reside on platforms (or across multiple platforms) or they may be integrated in (or across) fixed command and control centers. SoS may operate at the single service level, joint service level, or in international situations, a coalition level. In addition, the operational level may change for each situation or mission.

In many cases today, it is often easier and more affordable to provide new capabilities using existing systems that interoperate with each other as an SoS or in an enterprise. The DoD acquisition process for providing new capabilities requires an analysis of alternatives that, in addition to evaluating options for a new system, often includes one or more SoS alternatives for providing the desired new capability: there may be a set of existing systems that can be modified so that they can interoperate to achieve the desired capability.

Participation in a SoS tradespace differs from the single system tradespace in that the SoS (or enterprise) system engineers focus on options within an existing set of systems (often in various stages of development and sustainment) rather than a tradespace for a new system. This change of focus presents many challenges that can often conflict with single system engineering goals for performance and affordability. According to the *DoD Systems Engineering Guide for Systems of Systems*, the SoS tradespace often can be characterized as follows:

> "*When assessing how to support SoS functions, it is important to develop a solid technical understanding of the functionalities, interrelationships, and dependencies of the constituent systems. But in an SoS it is equally important to understand the objectives, motivations, and plans of those constituent systems, since these factors play a large role in SoS SE trades. In many cases, decisions about where to implement a needed function are based on practicalities of development schedules or funding as much as on optimized technical allocations. When a needed function is aligned with the longer-term goals of a particular system's owner, it may be advantageous to select that system to host the function even if*

there are more technically favorable alternatives. Funding is more likely to be available for development and maintenance, and the program sponsor may be more motivated to adjust schedules and make alterations if the function benefits the owning organization in the long term." [ODUSD(A&T)SSE, 2008]

As mentioned earlier, many of the software-intensive SoS reside on platforms, weapons, or in command and control centers. These systems are used to share information and coordinate activities between users or platforms and decision makers. It is often the software embedded on platforms that provide much of the desired platform flexibility and adaptability. As a result, many of the SoS capability trades are related to software architectures, interoperability, reliability, flexibility, adaptability, and data compatibility with other systems.

There are also SoS tradespace implications for software resources such as hardware, power, heat, and space on the platforms, weapons, and command and control centers. Newer technologies such as multi-core hardware and "big data" techniques can be employed to expand the software capabilities on large platforms such as aircraft and tanks as well as on miniaturized platforms such as UAVs, UGVs, and satellites. There are also tradespace opportunities for networks and other mechanisms for sharing data/information across multiple platforms or geographic regions. We can build systems that can work with "big data", but if we can't access or move the "big data" in a timely manner, then the potential value may not be achieved. In addition, the data-related "affordability" trade may require bigger, faster networks and have implications for security, reliability, trustworthiness, etc. of systems that are collecting and processing the data.

Common Trades at SoS Level

Affordability trades conducted at the system level are focused on making decisions that provide for a system that meets its basic requirements, that has “good bones” that can provide for longer term flexibility and adaptability to meet future needs, and that are within cost constraints. However, current DoD affordability constraints also apply to the broader mission capabilities, asset management, and inventory management. This means that trades must also be considered at the portfolio and SoS levels. The following summarizes some key trades at the portfolio and SoS levels.

SoS performance and interoperability vs. single system performance: Single systems generally optimize designs and implementations for single system performance. Similarly, an SoS engineering team will encourage single systems to conform to standard (or compatible) protocols, data formats, and algorithms to facilitate interoperability between multiple constituents within an SoS. However, this can create performance challenges for the single system, especially in cases where the single system interoperates in more than one SoS and the multiple SoS do not have compatible protocols, data formats, or algorithms. When systems support multiple protocols vs “a few” standard protocols, it

- Enhances interoperability between systems using multiple protocols
- Provides flexibility at system interfaces
- Increases complexity of systems that must receive/transmit using multiple protocols

When systems provide "a few" standard protocols (vs. multiple protocols), it can

- Increase system performance/throughput by minimizing conversions
- Reduce development and support costs (e.g., supports affordability objectives)
- Support better reliability, security due to less complexity
- Makes VV&A easier.

Similar trades exist for a few common data formats vs. multiple data formats. For additional information on convergent protocols and common data standards, see (USAF SAB, 2005). Another SoS architecture framework that focuses on architecture and data standards is the Vehicular Integration for C4ISR/EW Interoperability (VICTORY) frameworks and standards developed by the US Army (http://victory-standards.org).

Management of legacy systems – when to retire: Even the best of systems become difficult to maintain as they age. Software systems reach a point where the technology, middleware, and embedded commercial off the shelf products become obsolete and are no longer upgradable. Hardware components are no longer available to replace failed components. More affordable options in this situation might include a) re-engineering and/or porting the legacy system to a more current platform/environment in order to provide the needed functionality in a more cost-effective way and b) retiring the system and replacing it with a commercial product or equivalent capabilities in other existing systems.

How to most affordably manage SoS capability development and evolution: In the SoSE environment, there is a taxonomy of SoS that reflects varying levels of management authority and responsibility: virtual, collaborative, acknowledged, and directed (Maier 1998; Dahmann and Baldwin 2008). Of interest to SoS and capability affordability decisions is when to engage an SoSE team to guide SoS capability development and evolution, e.g., transition from a collaborative SoS to an acknowledged SoS. SoSE cost model research (Lane, 2009) shows that there are cost-related trades to be made when deciding whether to allow the constituent systems SE teams to collaboratively decide how to best implement and upgrade SoS capabilities or whether to use a more independent SoSE team to manage the SoS capability development and evolution. The research described in (Lane, 2009) indicates that several factors determine the cost effectiveness of an SoSE team: size of SoS, scope of SoS capabilities, level of SoS changes to systems vs. single system stakeholder changes, and level of SoSE constituent system oversight (OSF). **Figure 30** shows an example cost analysis where the return on investment of an SoSE team depends upon the desired level of constituent system oversight: minimal oversight (5%) can result in a considerable savings in person months as the number of systems in the SoS increases and a high level of oversight (15%) can result in no savings and in fact, can be more expensive in terms of labor hours than a collaborative SoS as the number of systems in the SoS increases.

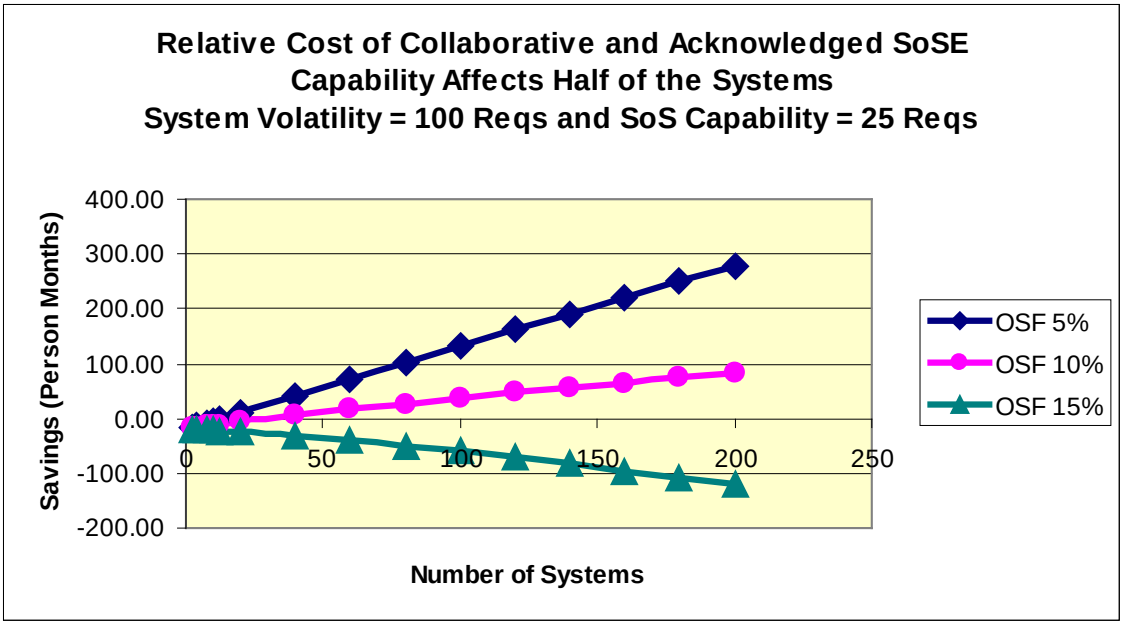


Figure 30: Example Collaborative vs. Acknowledged SoS Trade

Summary

Affordability considerations for participating in one or more SoS include:

- New systems to provide new technologies and capabilities not easily provided by existing systems
- Existing systems to upgrade existing single-systems capabilities, some of which are required to support cross-cutting SoS capabilities
- Sustainment and enhancement of existing cross-cutting SoS capabilities that the system participates in
- Development of new cross-cutting capabilities using data and services from other systems.

To achieve the maximum value from a system that participates in one or more SoS, it is important to begin with "good bones", that is, solid, flexible foundations upon which capabilities can be developed and maintained over time. In addition, foundation considerations need to include manufacturability, flexibility to support future options and take advantage of future opportunities, and key for systems in an SoS environment, interoperability to easily interact with other systems. These "good bones" can support early affordability as well as enable maximum value over the life of key systems by allowing them to effectively interoperate with each other to share data and services.

References

Dahmann, J. and K. Baldwin. 2008. "Understanding the Current State of US Defense Systems of Systems and the Implications for Systems Engineering." Paper presented at IEEE Systems Conference, 7-10 April, Montreal, Canada.

Lane, J., Impacts of System of System Management Strategies on System of System Capability Engineering Effort, PhD Dissertation, University of Southern California, 2009.

Maier, M.W. 1998. "Architecting Principles for Systems-of-Systems." *Systems Engineering.* 1(4): 267-284.

Office of the Deputy Under Secretary of Defense for Acquisition and Technology, Systems and Software Engineering. Systems Engineering Guide for Systems of Systems, Version 1.0. Washington, DC: ODUSD(A&T)SSE, 2008.

United States Air Force (USAF) Scientific Advisory Board (SAB). 2005. *Report on system-of-systems engineering for Air Force capability development*; Public Release SAB-TR-05-04.

1.6 Affordability and Better Buying Power

Affordability is a particularly important ility for iTAP, as indicated by the A in its acronym. It is also particularly challenging, given DoD's needs to cope with increasing speed and diversity of both adversary threats and technology opportunities, all within likely decreasing effective budgets. Its importance has escalated with recent initiatives by DoD leadership.

In particular, Better Buying Power is an initiative started by Dr. Ashton Carter, when he was the Under Secretary of Defense for Acquisition, Technology, and Logistics, and continued in his current position as Deputy Secretary of Defense. It seeks to define affordability for future DoD acquisition, operations and support capabilities. It is documented in 61 memoranda at http://bbp.dau.mil/references.html. Perhaps the best summaries for iTAP's purpose are the sets of definitions in the two "Implementation Directive for Better Buying Power" memoranda dated 03Nov2010 and 24Apr2013 on the DAU web site. Portions of the 2010 memorandum are cited below, with corresponding iTAP objectives developed during Phase 1.

1.6.1 The 2010 Set of Affordability Definitions and Corresponding iTAP Objectives

The 2010 memorandum states in the key part on DoD life cycle decision points ("I" refers to Dr. Carter in his then USD (AT&L) role):

… I will implement affordability-based decision making at milestone decision points for all Acquisition Category (ACAT I) programs. Specifically, I direct the following actions:

Baseline Portfolio and/or Mission Area Definitions: As a basis for affordability analysis, you will use standard budget categories to the extent possible. Representative examples include: tactical wheeled vehicles, tactical aircraft, surface combatants, and communications satellites.

The corresponding iTAP objectives will combine developing general tradespace and affordability definitions and analysis foundations addressing the full range of DoD systems and portfolios, while also developing and prototyping compatible solution capabilities for particular-system budget categories, while ensuring that the resulting capabilities can be adapted across the full range of DoD systems and portfolios.

Milestone (MS) A: You will establish an affordability target to be treated by the program manager (PM) like a Key Performance Parameter (KPP). This affordability target (initially, average unit acquisition cost and average annual operating and support cost per unit) will be the basis for pre-MS B decision making and systems engineering tradeoff analysis. This analysis should show results of capability excursions around expected design performance points to highlight elements that can be used to establish cost and schedule trade space. The affordability target should be presented in the context of an analysis of the resources that are projected to be available in the portfolio(s) or mission area(s) associated with the program being considered for the MS A decision, assuming programmed defense budgets and force structures. In order to meet this requirement, you will provide a quantitative analysis of the program's portfolio or mission area across the life cycle of all products in the portfolio or mission area, including acquisition and operating and support budget suitability to absorb the proposed new start as a content change. Specifically, if introducing a new program into a portfolio or mission area, you should indicate what specific adjustments will be made to absorb the new program.

The corresponding iTAP objective will be to develop a full-coverage cost estimation capability for next-generation systems and portfolios in a particular domain for specificity and calibration purposes, but architected to be easy to tailor to other domains (e.g., by identifying which costs need to be estimated by domain-specific parametric models, and which costs can be estimated by more general unit-cost, activity-based, learning-curve-based, or portfolio-based estimation methods). From a tradespace standpoint, it will recommend using desired and acceptable ranges of cost and other mission parameters, along with prioritization of parameters. It will also initially focus on parameters likely to be specifiable prior to Milestone A, and on estimating both cost and savings impacts on other existing systems and portfolios.

Milestone B: You will present a systems engineering tradeoff analysis showing how cost varies as the major design parameters and time to complete are traded off against each other. The analysis will pay due attention to spiral upgrades. You will recommend for my approval to establish and document, in the Acquisition Decision Memorandum (ADM) and in the program baseline, an 'Affordability Requirement' for acquisition cost and for operating and support cost. This requirement will be the functional equivalent of Key Performance Parameters (KPPs) for baseline establishment and monitoring. You will provide cost tradeoff curves or trade space around major affordability drivers (including KPPs when they are major cost drivers) to show how the program has established a cost-effective design point for these affordability drivers.

The corresponding iTAP objective will be to develop subsequent versions of the model family that include parameters more likely to be known by Milestone B, with corresponding increases in detail and accuracy.

Further important consideration for iTAP include:

1. The view is at the portfolio or mission level, not at the individual weapon system level. This enables the decision authority to make trades inside the portfolio as a way to balance affordability with mission requirements.

   The corresponding iTAP objective will be to build on the portfolio and product line cost and risk-based portfolio analysis capabilities developed for ship maintenance in the SERC RT-18 Valuing Flexibility project, to provide initial analysis capabilities that can be extended to other domains.

2. The affordability targets are defined as values for average unit acquisition cost and average annual operating and support cost per unit. Average procurement unit cost (a synonym for AUAC) is "is calculated by dividing total procurement cost by the number of articles to be procured. Total procurement cost includes flyaway, rollaway, sailaway cost (that is, recurring and nonrecurring costs associated with production of the item such as hardware/software, systems engineering (SE), engineering changes and warranties) plus the costs of procuring technical data (TD), training, support equipment, and initial spares." (https://dap.dau.mil/glossary/pages/1471.aspx)

   The corresponding iTAP objective will be to develop capabilities for estimating life cycle costs and savings in operational and support personnel, facilities, equipment, and consumables.

3. Trades at the portfolio level will be made in terms of the total funds available for that portfolio for 30 years (see newer guidance, below) and in harmony with the Total Obligation Authority for that period. TOA is the line budget authority granted by Congress. In other words, to add a new weapon, the portfolio must show what is it cutting in order to stay

even with the TOA. This new view discourages the previous practices of: accounting only for the development phases, not operations and sustainment, and "fixing" programs once they emerged from development by applying operations and sustainment funds (that is, a different color of money).

4. Point solutions are no longer permitted, as the relevant trade space must be presented, especially in terms of the affordability target.

   The corresponding iTAP objective will be to enable stronger tradespace analysis by specifying desired and acceptable ranges of system and portfolio cost and mission effectiveness parameters rather than point targets, along with prioritizing the effectiveness parameters and estimating resulting savings as well as costs.

1.6.2 UPDATES TO THE AFFORDABILITY BASELINE AND iTAP IMPLICATIONS

The 2010 baseline memorandum was later updated by, among others, the April 24, 2013, "Better Buying Power (BBP) 2.0 Guidance and Actions." (contained in http://bbp.dau.mil/doc/USD-ATL%20Memo%2024Apr13%20-%20BBP%202.0%20Implementation%20Directive.pdf) It expanded the scope of affordability, as follows:

> The initiative to provide affordability constraints that was put into practice over 2 years ago under BBP 1.0 will continue and will be enforced. Affordability analysis guidance and the process to establish affordability goals (formerly called "affordability targets") at Materiel Development Decision (MDD) and Milestone (MS) A and affordability caps (formerly called "affordability requirements") at the Pre-Engineering and Manufacturing Development (Pre-EMD) and MS B Review and beyond are included in the Department of Defense Instruction (DoDI) 5000.02 revision currently in the coordination process. Component Acquisition Executives (CAEs) are required to establish affordability goals and caps for lower Acquisition Category (ACAT) level programs as they are considered for MDD, MS A and B, or the equivalent, and beyond. Affordability constraints should be based on the anticipated available level of future budgets that will be available to procure and support the product being acquired within a relevant portfolio of products. In general, affordability constraints are the product of budget, inventory, and product life-cycle analysis within a portfolio context. They are not the product of cost analysis but a constraint on costs. Affordability constraints force prioritization of requirements, drive performance and cost trades, and ensure that unaffordable programs do not enter the acquisition process. If affordability caps are breached, costs must be reduced or else program cancelation can be expected. Constraints stem from long-term affordability planning and analysis, which is a Component leadership responsibility that should involve the Component's programming, resource planning, requirements, and acquisition communities.

Affordability analysis will examine competing Component fiscal demands for production and sustainment within a relevant portfolio of products over enough years to reveal the life-cycle cost and inventory implications of the proposed new products within the portfolio – nominally 30 to 40 years. Example portfolios include tactical aircraft for the Air Force, shipbuilding for the Navy, and ground combat vehicles for the Army. This analysis should be relatively stable and useful for multiple programs until an update is required. A program is defined to be affordable if the driving cost elements – usually production and sustainment – can be accommodated within the modernization and recapitalization plan for the portfolio. If not, then either a lower cost product or identifiable reductions in another component portfolio (trading shipbuilding for tactical aviation within the Navy, for example) must be pursued.

Affordability analysis and recommended constraints will be presented to the Milestone Decision Authority (MDA) before major acquisition decisions to demonstrate the affordability of the program.

Each MDA will then establish affordability constraints in the form of goals and caps. Affordability goals are set at the MDD or MS A to inform early requirements and design tradeoffs. Affordability caps are set at the Pre-EMD Review or MS B, for unit procurement and sustainment costs and will be considered equivalent to Key Performance Parameters (KPPs) within the Acquisition community. Implementation during the program's life cycle will require Program Manager (PM) diligence and support from Configuration Steering Boards to meet affordability constraints set by the MDA, and the PM will promptly notify the MDA if a constraint will be exceeded.

SPECIFIC ACTIONS:

The Assistant Secretary of Defense for Acquisition (ASD(A)) will provide additional details on requirements, formats, and supporting data submissions in the revised DoDI 5000.02, "Operation of the Defense Acquisition System," as well as updates to the Defense Acquisition Guidebook (DAG) and the Defense Acquisition Board (DAB) templates by June 1, 2013.

ASD(A), with support from the Service Acquisition Executives (SAEs), will define a standard list of portfolios for my approval by June 1, 2013.

Director, Acquisition Resource and Analysis (ARA), will update its program data repository, the Defense Acquisition Management Information Retrieval system, to track affordability constraints, effective immediately.

CAEs and all other MDAs who have not already done so will develop and issue similar guidance to apply affordability constraints to ACAT II–IV programs by July 1, 2013.

MDAs are responsible for enforcing affordability caps effective immediately.

The impact of this new guidance is to amplify the previous directives, offer greater specificity (for example, the length of time to consider costs), to broaden the scope to more of the acquisition life cycle, and apply them to more Acquisition Categories, namely those less than the highest (that is, II-IV).

The corresponding iTAP objectives will be to develop tradespace, capability prioritization, total ownership cost, and portfolio cost, savings, and effectiveness estimation capabilities and associated tradespace methods, processes, and tools to support such analyses. These will include the costs in future DoD systems and portfolios of investing in portfolio definition and support, of evolving multiple versions within a portfolio, of enabling systems and portfolios to evolve in support of evolving systems of systems, and of coping with increases in the speed and diversity of future threats and technology opportunities.

The parts that apply most critically to iTAP are, "In general, affordability constraints are the product of budget, inventory, and product life-cycle analysis within a portfolio context. They are not the product of cost analysis but a constraint on costs." This does not mean that cost analysis is not involved in system definition, but just the opposite. In order to converge on affordable solutions, cost analysis cannot be deferred until after system definition. It needs to be an active participant in providing cost analysis capabilities that enable realistic capability-affordability tradeoffs to be made during the system and portfolio definition process.

iTAP Methods, Processes, and Tools (MPTs)

2.1 Overall Approach (USC)

A major objective of iTAP Phase 1 has been to summarize and demonstrate the team members' iTAP capabilities to interested parties to identify potential early-adopter organizations for piloting the capabilities, and for identifying high-value areas for extending and refining the capabilities.

Two such activities were pursued at the iTAP team workshops at the INCOSE International Workshop (IW) in Jacksonville on January 28, 2013, and at the Conference on Systems Engineering Research (CSER) in Atlanta on March 19, 2013. In addition, two visits to the Army Engineer Research and Development Center (ERDC) in Vicksburg, MS, the lead organization for

the DoD Engineered Resilient Systems (ERS) key research area on January 8 and April 30, 2013. Some further exploratory engagements involved Georgia Tech demonstrations personnel, NAVSEA CREATE-Ships personnel, and Army TARDEC personnel, and USC exploratory demonstrations and discussions with USAF/SMC, NRO, and Aerospace Corp. personnel with respect to researching and incrementally developing a next-generation full-coverage space systems cost estimation model, provisionally called COSATMO.

2.1.1 MPT DEMOS AT INCOSE IW, CSER; TWO ERDC PRESENTATIONS

The January 28 iTAP workshop at INCOSE IW was held in concert with the INCOSE Affordability Working Group, and focused its discussions and demonstrations on cost and cost-effectiveness models. These have led to further discussions on refining the systems engineering cost estimation model (COSYSMO) extensions for modeling reuse and requirements volatility into a unified model that also estimates the return on investment from investing in reusable components, and on researching a next-generation cost estimation model addressing current and future trends such as model-driven development, 3D printing, cloud services, and revisions in DoD acquisition and ownership guidance.

The March 19 iTAP workshop at CSER involved several discussions and demonstrations of iTAP Capabilities. For example, GTRI provided a demonstration of the USMC Framework for Assessing Cost and Technology (FACT) tool, that showed the utility of using FACT to achieve near real-time analysis for exploring the design parameter trades that affect the overall performance, reliability, and cost of a system design. FACT provides decision support tools to the acquisition program to manage risks of cost, schedule, and performance through a rapid analysis of alternative technology and materiel using surrogate models, or equation regression representations of more complex M&S tools, as illustrated through several successful implementations discussed during the demonstration. FACT will ultimately reduce program development and life cycle costs, both of which are tenets of effective “should-cost” management, by providing a dynamic tradespace between traditional performance metrics (such as range and speed), with those -ilities that drive all lifecycle phases (such as affordability and reliability).

FACT’s data schema is based on SysML. The demonstration also showcased using FACT as a web-based SysML authoring tool that provides real-time collaboration. The goal is to offer an experience where all users have up-to-date information and can concurrently modify the model. To date all SysML authoring tools are single-user, thick-client software applications geared towards only model definition, forcing sequential model development and losing an opportunity to parallelize effort. Additionally, the completed model is not in and of itself able to execute tradespace exploration or constraint optimization. Most complex system development occurs in geographically disparate teams working on various components of the model concurrently, which introduces a process complexity in configuration management and

slows progress. Although extensions to thick-client applications and separate software applications have attempted to solve these issues, FACT poses a different solution in one integrated environment. Further information on FACT is provided in Section2.2.1.

2.1.2 Resulting interest at ERDC: CREATE-Ships, CRES-GV, MIT Epoch-Era approach

The two SERC visits to ERDC and an ERDC visit to GTRI resulted in several ERDC-suggested initiatives, including exploration of complementing the physical modeling capabilities of CREATE-Ships and CRES-GV with SERC modeling capabilities for software-intensive systems and cost estimation.

CREATE-Ships has expressed interest in affordability analysis tools to incorporate into set-based and inside-out design methods. The affordability analysis needs include the time and cost of design, production, operation and sustainment, re-purposing, and service life extension. CREATE-Ships also expressed interest in tools addressing tradeoffs of versatility (flexibility, adaptability, extensibility, re-configurability, and changeability) with other ilities. Important design factors include reserve capacity (size, power, weight and cooling), packing density, and modularity. A meeting with CREATE-Ships personnel was arranged for Phase 2 in June.

CRES-GV was only recently initiated and is still formulating their objectives and plans. Wayne State personnel are involved in the planning activities and will pursue collaborative efforts with TARDEC, as discussed below.

In addition, ERDC personnel expressed interest in the applicability of the MIT Epoch-Era approach to improving their ability to address uncertainties in their logistics supply-chain system development and evolution. MIT personnel are preparing for a followup visit to understand and suggest ways to apply the epoch-era approach to their situations.

2.1.3 TARDEC interest in WSU and GaTech; CREATE-Ships interest in GaTech FACT

The US Army TACOM Research Development and Engineering Command (TARDEC) has developed and is using a Systems Engineering framework, TARDEC's Advanced Systems Engineering Capability (ASEC). ASEC is a larger system acquisition System Engineering framework into which "ilities and affordability tradespace" MPT can be integrated to "fill in" the gaps in MPT for system capabilities, concepts and baselines. ASEC is a government-owned integrated suite of tools to help to document and navigate the system engineering process and tradespace. ASEC is growing in a planned evolution to provide enhanced functionality based on expressed user needs. ASEC provides an integrated, consistent process for program SE, and is a "backplane" for SE methods, processes and tools (MPTs) "plug-ins". ASEC has been and is being used on a portfolio of programs. ASEC has demonstrated reduced SE time and burden, and improved SE products. ASEC is under consideration by the Air Force, Marine Corps and Navy for adoption as a common, multi-service system. As we develop "ility and affordability tradespace"

MPT, we need to consider the larger framework(s) into which the MPT can/will be integrated. ASEC is a potential platform into which to integrate, apply, evaluate, and show the "value added" of SERC MPT.

At a more detailed level of analysis, there are a variety of integration/evaluation frameworks in use and/or under consideration by different elements of the TACOM and TARDEC. The PEO GCS developed the Whole System Trade Analysis Tool (WSTAT) and the Capability Portfolio Analysis Tool (CPAT). WSTAT is a combinatoric system analysis tool intended analyze among capabilities, affordability, program timeline objectives, and additional second-order effects given component and subsystem properties. WSTAT is intended to assist the Requirements Review Team in setting performance specification level to meet program cost, schedule, and performance objectives. CPAT is designed to identify the optimum courses of action (cost, schedule, and performance) for portfolio investment. Integration of these tools with ASEC is under consideration. The Threat-Oriented Survivability Optimization Model (TOSOM; developed at TARDEC) is a combinatoric analysis tool specifically for tradeoff analysis of the survivability subsystem suite against cost, weight and other burdens. It incorporated models and databases of the effectiveness of different survivability technologies against different threats. The Concept and Modeling Tool Suite (CMTS; developed for TARDEC at the University of Louisville) focuses on the architectural level of vehicle structure design. CMTS is built around function-based abstractions that represent structurally integrated assemblies (cab/crew compartments and other volumetric enclosures, frames, suspensions, closures, etc.) and transfer-process component associations (powertrains, braking systems) for full vehicle architecture concepts. It is hoped that CMTS package's architecture-level view could also help adapt existing vehicles to changing operational conditions.

TARDEC generally appears willing to consider alternative tools and structures. Concerns include the time and effort needed to configure tools to align with the Ground System Architecture Framework (GSAF), and to integrate models of ground vehicle functions and technology alternatives. Individual Program Managers and project leads make the final determination as to what tools will be used on any given program, when they will be used and what they will be used for.

Additionally, the Engineer Research and Development Center (ERDC) has shown interest in leveraging GTRI's Collaborative Web-based Tradespace Tools, such as FACT (see section 2.2.1), to support the Ships-related ERS and CREATE programs. Discussions are underway for GTRI to support the ERS program with an initial proof-of-concept tradespace tool with a focus on an application for ships. This tradespace tool would allow a group of stakeholders to examine and trade capabilities of various systems of interest, integrating modeling and simulation (M&S) within the acquisition process. This includes a collaboration with the Naval Surface Warfare Center Carderock Division (NSWCCD).

2.1.4 USAF/SMC and Aerospace interest in COSATMO full-coverage cost model

2.2 GROUND VEHICLE DOMAIN

2.2.1 FACT (GATECH)

The GTRI-developed USMC Framework for Assessing Cost and Technology (FACT) as a candidate tool for tradespace analysis. FACT is an open architecture web services based environment that enables models to be interconnected in order to provide a rapid exploration of the design tradespace in support of systems engineering analysis. FACT is government owned, model agnostic, and capable of linking disparate models and simulations of both government and commercial origin through the application of community established data interoperability standards.

FACT has been developed thus far for application to ground vehicles for the USMC. It does, however, have strong potential as a candidate tool to incorporate and integrate –iliity methods and toolsets being developed as part of the ongoing SERC effort. the USMC Framework for Assessing Cost and Technology (FACT) is an effort spearheaded by the U.S. Marine Corps System Command as a real-time, collaborative modeling, simulation and design web-service. During the ITAP RT46 Phase 1 effort, GTRI investigated relevant, existing tools and their ability to capture –ilities in a tradespace environment. The investigations were limited to those toolsets developed by SERC members involved in the ITAP Phase 1 effort. FACT was identified as a promising tool, integrating MBSE approach and methodology, that enable tradespace analysis incorporating –iliities defined as critical to support DoD and other acquisition and design processes. A FACT-like framework and methodology may incorporate extensions to the SERC team's methods as they are defined to capture -ilities tradespace of interest.

This section describes FACT in more detail, especially its utility to help achieve near real-time analysis for exploring the design parameter trades that affect the overall performance, reliability, and cost of a system design. FACT provides decision support tools to the acquisition program IPT to manage risks of cost, schedule, and performance through a rapid analysis of alternative technology and materiel using surrogate models, or equation regression representations of more complex M&S tools.

Utility in Tradespace Analysis & Design Stage Applicability

FACT emerged to answer the following fundamental acquisition questions:

- How well will the system perform?
- How reliable will it be?

- How much will it cost?, and
- When can we get it?

As such, FACT is well-suited to evaluate –iliities in concurrent design. For example, the questions above reflect reliability and affordability directly.

The FACT process achieves the capability to answer these questions concurrently rather than in a stove- piped independent fashion. FACT algorithms recognize the inter-dependence of design and maintenance and procurement philosophy on the tradespace. The options in the tradespace represent the inter-related impacts of cost, performance and reliability based on the multitude of design options available to the Program Manager. Based on the options selected, FACT calculates the procurement cost for the system and projects, the operational and support costs for the system versus a level of performance, and associated reliability metrics. Additionally, FACT is designed to support the Analysis of Alternatives (AoA) process, comparing viable options against weighted objectives.

FACT addresses the challenge of how to determine will- and should-cost and then apply it in a concurrent tradespace analysis. FACT has developed cost estimation relationships between the design components and addresses the maintenance philosophy, repair vs. replace, obsolescence analysis, disposal strategy, and usage data developed from operational scenarios to project operations and support cost information. Similarly, procurement cost data has been gathered from contracts and purchase documents.

FACT is designed primarily to provide tradespace analysis during conceptual design. Addressing the broad challenges of modeling and simulation (M&S) support for the acquisition enterprise is a huge problem space and requires some upfront choices about where increments of benefit can be obtained quickly and with the greatest return on investment. Recognizing the choices made during the DoD 5000 pre-milestone A, conceptual design of systems offers the greatest opportunity to influence the performance and cost of a system. Other stages of the system lifecycle can benefit from the FACT process, but the conceptual design phase is where both good and bad decisions have the greatest impact on cost and performance.

Initial Development to Support Ground Vehicle Design

While the Marine Corps performs systems engineering across the gamut of systems, the most immediate and largest opportunity to realize a benefit was in the domain of ground tactical vehicles. Consequently, the first applications of FACT were to the Amphibious Combat Vehicle (ACV) and the recapitalization program from the HMMWV, building heavily on prior work in support of the Marine Personnel Carrier (MPC) program. The focus on these ground vehicles does not imply the FACT framework is not equally applicable to weapon systems or Command and Control (C2) systems. Exploratory efforts are underway to examine the applicability of FACT in support of naval surface vessel programs, weapon systems, and Unmanned Aerial System (UAS) programs with joint applicability.

FACT Framework

The FACT framework focuses on interoperability and data sharing with the emphasis centered on metadata. Definition of metadata is the building block on which the FACT data sharing is founded. FACT was designed on a philosophy of open architecture to enable extensibility. To achieve this, and avoid the encumbrance of licensing fees limiting its use or tethering it to a single manufacturer over its lifetime, FACT was built using open source software and government-owned code.

FACT's approach to model based systems engineering (MBSE) is in response to the need of defense acquisition programs for decision support tools that facilitate the concurrent management of system requirements, performance, cost and reliability in the context of rapid tradespace analysis. FACT's development followed guidance from the Department of Defense, mandating that it be web-based and accessible from common computer workstations, be built entirely from open source software, and offer an open and extensible architecture [1, 2]. Although the development of FACT was originally envisioned for vehicle acquisition programs, the overall process and application is independent of vehicles and can be applied to any system-of-systems.

Recognizing the need for quick response from user queries, FACT incorporates the use of surrogate models in its design, which are parametric regression equation representations of high fidelity M&S tools developed via Designs of Experiments sampling [3]. These surrogate models are developed offline from FACT, but are easily integrated into the framework.

M&S in support of defense acquisition often suffers from stove-piped processes, creation of boutique solutions and one-off tools without broad application beyond a specific program, and lack of authoritative data management to facilitate the reuse and update of models. To address these issues, FACT approaches the problems from the perspective of creating a process based on open architecture and open standards, focusing on the portability of data rather than direct communication between disparate models of vastly different pedigrees. FACT facilitates creation of a federation of models and access to databases by creating a common language for data interchange based on the Systems Modeling Language (SysML).

SysML Backbone

The Systems Modeling Language (SysML) is gaining traction within the systems engineering community to define complex systems. SysML is a general-purpose graphical modeling language for MBSE applications that supports the specification, design, analysis and verification of a broad range of systems [4]. It is a subset and extension of the Object Management Group's Unified Modeling LanguageTM (UML), the industry standard for modeling software-intensive systems, which gives systems engineers the ability to represent system requirements, structure, behavior and properties using a formal diagram syntax. It includes nine diagram types, seven of which are directly borrowed or modified UML 2 diagrams, and two of which, requirement and

parametric, are added to support requirements engineering and performance analysis, respectively. The diagrams can be thought of as windows into the model that offer different perspectives from which to view the system. SysML is not a methodology or tool for model development and execution; it simply provides a standardized set of semantics and notation for creating graphical system models.

SysML models consist of an interconnected set of model elements, or instances of SysML metaclasses, whose relationships to one another are visualized via symbols on the various diagram types. A model element must conform to the properties, constraints and relationships allowed by its metaclass, ensuring model consistency. Beyond using SysML to define systems, many commercial SysML tools allow users to connect constraints to external models to produce a more comprehensive and executable model.

SysML was highly leveraged as the point of reference for the data schema implemented as FACT was initially developed. SysML provides a general requirement construct that offers a title and human-readable descriptive statement. Often, though not always, requirements can be represented in a quantitative manner. SysML's requirement construct is insufficiently strong to map a value property of one block to a requirement that could define threshold and objective values. As FACT was being developed, this shortcoming was identified so the data schema utilized by FACT strengthened the requirement concept. The FACT team envision that by extending the current SysML specification requirement construct, any type of SysML model execution engine could easily provide an automated means to determine how an instantiation of a system meets/exceeds/falls short of its requirements. In order to do this in FACT, each quantitative requirement is associated with a value property, which is a calculated value for a defined constraint. By enforcing this relationship, requirements move beyond just being human-readable descriptions and can provide insight into feasibility across a set of requirements.

To date all SysML authoring tools are single-user, thick-client software applications geared towards only model definition, forcing sequential model development and losing an opportunity to parallelize effort. Additionally, the completed model is not in and of itself able to execute tradespace exploration or constraint optimization. Most complex system development occurs in geographically disparate teams working on various components of the model concurrently, which introduces a process complexity in configuration management and slows progress. Additionally, the FACT team found that executing external models could be somewhat cumbersome and inefficient.

Since the FACT data schema was already based on SysML, the FACT team investigated incorporating an in-browser SysML authoring tool as a modular plug-in to the greater framework in order to address the issues described above and to provide real-time collaboration. The goal is to offer an experience where all users have up-to-date information and can concurrently modify the model.

In late 2011 the Object Management Group (OMG) officially adopted an XMI specification providing a true standard for describing a SysML model [5]. Adoption of this specification allowed any SysML authoring tool vendor to understand, at least at the model level, a SysML model developed in any other authoring tool adhering to the specification. Prior to this, the means for storing the model data varied from tool and tool (and with respect to the visualization aspect such as the diagrams, the same is still true). Because OMG has published this open source specification, new authoring tools, now have a straightforward approach for ingesting existing models or providing users the ability to move among authoring tools.

The FACT team is still refining the capability to import or export a SysML model that adheres to the XMI specification, encouraging users to try the web-based capability. However, significant work has been completed in how to understand a model described using the XMI specification. A primary overarching goal of the development was to provide an interface with the adopted SysML XMI specification such that users could chose to enter or leave the web-based authoring tool for a thick-client at any point in time using a provided interface. The development team also wanted to ensure that the initial set of authoring capabilities was sufficient to utilize the tradespace analysis capabilities offered by FACT, so that it was in some sense immediately executable at the completion of the SysML model. The objective is to seamlessly connect the system-level model to the detailed domain specific models, ultimately enabling a connected model-based approach across the product lifecycle and through its entire hierarchy.

In order to support a wide range of web browsers (specifically older though still commonly used ones such as Internet Explorer (IE) 7 and 8), certain technologies such as scalable vector graphics (SVG) could not be used directly. To provide a desktop-like experience in a web browser, the team identified a collection of Javascript libraries. The most highly leveraged libraries include JointJS, Raphaël, and jQuery [6, 7, 8]. Each of these is open source and used widely by other web applications. JointJS provides a library to build UML diagram elements as well as make each component movable on the page. The team extended the JointJS library to include necessary SysML diagram elements. Raphaël abstracts the creation of visual elements and will create either vector markup language (VML) or SVG files depending on the browser being used by the client; for example in IE7 VML is generated while if using Google Chrome an SVG is generated. JointJS utilizes Raphaël to draw the appropriate diagram elements. Finally jQuery is actually a collection of libraries that greatly simplifies how to modify the Document Object Model (DOM) (i.e. the web page) as a user interacts with it. Selecting these technologies ensured that just about any web browser in use today could be used to access the authoring tool.

For the collaboration piece, experience with FACT and other web applications led the team to select a solution which could utilize either an XMPP server or a standard asynchronous server-side polling service [9]. Both solutions allow a user to receive updates automatically and instantaneously from a server as a result of other users' interactions with a model. XMPP is the

technology utilized by most chat clients and Google Docs® to provide real- time user experiences. An important note about both of these approaches is that the primary load for handling the distribution of messages to simultaneous users remains on the server, not with the client, a critical design choice for increasing accessibility and user experience.

Since SysML is a graphically based standard, prior to release of the XMI specification the most appropriate means to store the data would have been in an SVG or similar format. Even with an XMI specification, storing an entire model as a single XML document poses challenges. Making incremental changes to either format is not straightforward, therefore the FACT team determined the best approach was to define a data schema for storing a SysML model in a database. A schema was defined using JavaScript Object Notation (JSON), which lends itself well to being stored in the NoSQL database MongoDB [10, 11]. The team is beginning the process of releasing this standard for consideration by the broader community of interest to be eventually published.

Additional detail on FACT may be found in [12, 13].

1. Assistant Secretary of Defense (Networks and Information Integration) and DoD Chief Information Officer, DoD Inform
   Accreditation Process (DIACAP), Department of Defense Instruction 8
   2007.
2. Assistant Secretary of Defense (Networks and Information Integration) and Chief Information Officer, Clarifying Gu
   (OSS), Department of Defense Memorandum, 2009.
3. Forrester, A., Sóbester, A. and Keane, A. (2008). Engineering Design via Surrogate Modeling: A Practical Guide, West Sussex: John Wiley & Sons, Inc.
4. http://www.omgsysml.org/, Copyright © 1997-2012 Object Management Group, Inc.
5. http://www.omg.org/spec/XMI, Copyright © 1997-2012 Object Management Group, Inc.
6. http://www.jointjs.com, Copyright © 2009 David Durman.
7. http://raphaeljs.com, Copyright © 2008 Dmitry Baranovskiy.
8. http://jquery.com/, Copyright © 2012 jQuery Foundation.
9. http://xmpp.org, XMPP Standards Foundation.
10. http://json.org, Introducing JSON.
11. http://www.mongodb.org, Copyright © 2012 10gen, Inc.
12. Browne, D., Kempf, R., Hansen, A., O'Neal, M., and Yates, W. "Enabling Systems Modeling Language Authoring in a Collaborative Web-based Decision Support Tool", Conference on Systems Engineering Research (CSER), Atlanta, Georgia, 2013.

13. Ender, T., Browne, D., Yates, W., and O'Neal, M. "FACT: An M&S Framework for Systems Engineering", Interservice/Industry Training, Simulation, and Education Conference (I/ITSEC), Orlando, Florida, 2012.

2.2.2 ARMY GROUND VEHICLES (WSU)

2.2.2.1 Ground Vehicle ilities and Affordability Tradespace Needs

As with acquisition programs in different domains, acquisition programs have different tradespace needs in different stages of the acquisition. Different information is available and different types of decisions are made leading up to the Material System Analysis, Technology Development, and Engineering and Manufacturing Development stage. Decisions made early in the acquisition process tend to have disproportionally large effects on total cost of ownership. By some estimates, 85% of the life cycle costs are determined by decisions made prior to entry into Engineering and Manufacturing Development. In recent ground vehicle acquisition programs, steps to keep the tradespace open longer have been formalized in the acquisition strategy, emphasizing continuing to make capability/affordability trades even during Engineering and Manufacturing Development. Review of these considerations elucidates the needs for different types of ility and affordability tradespace MPTs. The following description is couched in the terms of the formal acquisition process. While only Major Defense Acquisition Programs (MDAPs), e.g., Acquisition Category I and II programs, go through the formal acquisition process, smaller programs including science and technology demonstrators, go through the same steps, but with less formality.

Prior to initiating Materiel Systems Analysis, the incipient program develops a set of capabilities (documented in the Initial Capabilities Document – the ICD), must present a convincing argument that the set of capabilities are feasible within the timeframe of the need and affordable at an acceptable level of risk for a positive Materiel Development Decision (MDD). This is prior to formulation of a system concept. The tradespace is defined in terms of core capabilities, deferred capabilities, affordability and risk. The GAO has identified mismatch between the capabilities, resources and risk at this stage as being a major contributor to later cost and schedule overruns, and performance shortfalls. Ility and affordability tradespace tools at this stage are needed to address capabilities, affordability and uncertainty at "rough order of magnitude" prior to developing solution concepts.

The steps of the subsequent Material Solution Analysis (MSA) phase are to develop a set of alternative concepts, conduct an Analysis of Alternatives (AoA), then select and/or synthesize a system concept (iterating as needed). The ICD is refined based on the findings and tradeoffs of the AoA. The GAO has identified failure to conduct a "robust AoA" with a diverse set of

alternative system concepts as a major source of "poor acquisition outcomes." In current practice, the alternative concepts are developed by the Program Manager's Office. Ility, affordability and uncertainty tradespace tools to partition and sample the tradespace could assist in developing a diverse set of alternative concepts. The GVX project will include a truncated AoA.

Following MSA and a successful Milestone A review, the acquisition process enters the Technology Development (TD) phase. The TD phase begins with defining the concept in the in draft system requirements, and functional and allocated product baseline documents. Recent practice has been to articulate three tiers of requirements: tier 1 non-tradeable, tier 2 tradeable, and tier 3 deferrable. Individual requirements may have threshold and objective levels of capability. Recent ground vehicle acquisition programs using this approach include JLTV, GCV and AMPV. The concept definition documents are input to competitive prototyping of the system and/or key system elements are employed to reduce technical risk, validate designs and cost estimates, evaluate manufacturing processes, and refine requirements. This is part of the Technology Development phase. The competitive prototyping results are used by the Government to produce a refined and detailed system concept: a Capability Development Document (CDD), RAM strategy, finalized system specification documents for competitive procurement. Ility can affordability tradespace tools can potentially be useful to the Government in developing the draft requirements, the contractors in making tradeoff decisions in their prototype designs, and in developing the final requirements. Important tradeoffs made at this stage include (1) capability tradeoffs to meet affordability goals, and (2) capability tradeoffs to limit the risk of cost and schedule overrun.

After Technology Development phase and a successful Milestone B review, the program enters the Engineering and Manufacturing Development (EMD) phase. Capability tradeoffs have always been entertained as a risk mitigation technique. Their tiered and tradable requirements structure formalizes the approach. Ility and affordability tradespace tools to assist in exploring how reducing capabilities expands the design tradespace, and how design decisions limit the capability and affordability tradespace.

2.2.2.2 Army Ground Vehicle Design and Development Principles

There are several important design and development principles for Army ground vehicle: Families of Vehicles, versatility, continuous modernization, and architecture-based design. These closely-principles are important for long-lived systems. Army ground vehicles typically remain in the inventory for 30 to 60 years.

Current and historical practice and intent is to base a family of vehicles, or product line, on a common platform. The M113, HMMWV, Stryker, and Bradley are excellent examples where one initial vehicle spawned a large number of mission variants. Recent acquisition initiatives

such as the JLTV, GCV, and AMPV all have explicit requirements for to support multiple variants and mission equipment packages. In some cases, the function puts such extreme demands on the platform that there are only limited opportunities for economical conversion to other purposes. The 155mm Howitzer (Paladin) and the Abrams Main Battle Tank are examples of platforms with limited opportunity for re-purposing. The benefits of platform-based product lines include improved affordability (commonality of parts, production facilities, training, maintenance equipment, etc.), ensured interoperability, and shared reliability growth upgrades.

Versatility is the ability for the vehicle to be adapted to different conditions, threats, and mission needs, and the ability to be extended by integrating more capable subsystems. Versatility applies both to the design, i.e., the design can be modified, and to physical instances of the vehicle. Adapting and extending the vehicle can take place in the field (e.g., replacing tires with mattocks, adding applique armor, etc.), or at depot as part of recapitalization.

Continuous modernization includes reliability growth (replacing components or subsystems as reliability problems become reveled through use), adding capabilities to meet evolving conditions and needs (System Enhancement Programs), and occasional block upgrades to restore the design margin (reserve capacity) for further growth and/or to incorporate substantial changes (e.g., a higher-capacity engine, a larger weapon, switching from analog to digital electronics).

Architecture-based design is an approach that enables the related capabilities of platform based Families of Vehicles, versatile systems and continuous modernization. Architecture-based design is a knowledge-based design approach. It is an expression of deep content knowledge to define generic architectures. The generic architecture includes a detailed generic product work breakdown structure (with options or branches that may or may not be part of any given realization), the network-interface structure among the subsystems, a catalog of technology options, and a collection of models and guidelines for sizing, design and evaluation of components, subsystems and interfaces.

2.2.2.3 Ground Vehicle Ility Priorities and Interactions

This section summarizes the major ility categories for ground vehicles. The definitions and explanations address how the terms are used in the ground vehicle domain. Manned and unmanned ground vehicle ilities are addressed separately. Unmanned ground vehicle ilities are described in terms of differences from manned ground vehicle ilties.

2.2.2.3.1 Manned Ground Vehicle Ilities

Affordability. Affordability includes the average unit production cost, operation and sustainment cost per mile (including the fully-burdened cost of fuel, cost of spare parts, maintenance and logistics support costs), and the development program average unit cost. Fuel economy, logistics reliability (mean time between failure), maintenance time, repair time, spares and logistics footprint are all contributors to operation and sustainment costs.

Force Protection. Force protection is also known as occupant survivability. It refers to the ability of the vehicle to protect occupants – crew and troops – from hostile attack. Force protection subsystems include intrinsic armor, add-on armor, energetic armor, crush layers, spall liners, fire suppression, seating, shaping (sloped glacis, "vee" shaped hull), active protection systems, mobility to escape or avoid attack, situation awareness (sensors), jammers, obscurants, and self-protection (counter-fire, dazzlers, etc.). Force protection is achieved by these systems working in combination. All of these systems add weight, especially armor. Weight itself reduces the acceleration from a blast. But weight reduces mobility. Mobility is a key element of force protection – to escape an attack, or limit the opportunity for an attack.

Survivability. Survivability refers to the ability of the system to function following a hostile attack. Survivability formerly included force protection, but force protection has recently been broken out a separate category. Since system functions, especially mobility and self-protection, contribute to force protection, system survivability is positively related to force protection.

Usability. Usability refers to human factors, safety, and training. It includes ingress and egress time, noise and vibration, shock and pitching, air quality and cooling, vision system quality, training time to operate and maintain the systems, "pinching" hazards (e.g., hatches), traction surfaces, and adequate workspace. The scope of usability includes both the crew and troops being transported.

Versatility. Versatility is a term from the Army Equipment Modernization Plan. In the plan it is defined as consisting of adaptability and extensibility. Other concepts under the heading of versatility include flexibility and changeability. Adaptability refers to the ability to add or replace mission equipment to perform different functions. Extensibility refers to the ability to increase the level of performance by replacing components with more capable components, or to integrated additional components to enhance capability. Versatility include the ability to integrate "kits" in the field such as "B armor", fording kits, etc. Versatility includes the ability to replace Line Replaceable Units with upgrades in the field, and to upgrade major subsystems at depot (e.g., engine, transmission, suspension). Versatility includes the ability to re-purpose the vehicle – i.e., to change its mission - replacing major components. Versatility includes both modifying existing vehicles as well as changing at the design and production stage. Versatility enables a basic system to become the seed for a product-line family of vehicles with common components and maintenance. Factors that contribute to versatility are reserve capacity (design margin) in size, weight, power and cooling (SWaP-C), modularity, standard interfaces, and common components.

Mobility. Mobility refers to ability of the vehicle to maneuver under its own power. Key mobility attributes include range per tank of fuel, acceleration, maximum speed, dash performance, turning radius, side slope stability, rubble-crossing, "bulldozing" capability, slope climb, soft-soil traction, handling, gap crossing, step climb, ability to negotiate constrained urban spaces, and fording. Key system attributes include ground pressure, horsepower per ton, torque per ton, ground clearance, length to width ratio, center of gravity height to vehicle width ratio. Factors that affect size and weight affect mobility. For amphibious systems, additional mobility parameters include maximum speed on water, range on water, maximum safe sea state, time to come up to maximum speed, surf zone safety, on water stability and handling.

Capacity. Capacity refers to the ability of the vehicle to carry troops and cargo. It includes interior volume, as well as free exterior surface areas where cargo can be attached without interfering with system functions. It also refers to the ability to mount and carry external equipment. Capacity requires the power and suspension to carry additional weight.

Interoperability. Interoperability refers to the ability of the vehicle to operate as a part of the combined arms team with other systems in tactical operations. It includes the ability maneuver and survive with the other vehicles, on the portion of the battlefield, in the missions the vehicle is intended for. It also includes the C4ISR/networking with the other vehicles – communications, data formats, networked applications, common data, etc.

Operational Reliability, Availability and Maintainability (RAM). Operational RAM refers the probability that the system can complete a mission without failure, the fraction of systems that are available, and the time to restore a vehicle to operational availability. Operational RAM differs from logistical RAM. Logistical reliability is the mean time and/or miles between a failure. Redundant systems increase operational reliability while decreasing logistical reliability.

Security. Security refers to the ability of the crew to detect potential threats, to prevent unauthorized access to the vehicle, its systems, or to interfere with its functions in situations other than combat.

Transportability. Ground vehicles must be transported to theater, and transported within theater. They are transported on-board ship, within fixed wing aircraft, slung under rotary wing aircraft, carried on flatbed trucks and railcars, and, in some cases, under their own power. Transportable constraints are the "cube" (height, width, length) and weight. Cargo holds, bridge, tunnel, and road widths constrain size. Lift and stability characteristics constrain weight. In some cases, transportability is facilitated by "kitting" the vehicle – some components, e.g., add-on armor, are installed after transportation to theater. The transportability evaluation for a vehicle includes which transports can carry the vehicle, the cube and weight of spare and other equipment, and the time to off-load and prepare the

vehicle for operation after transport. Common measures of transportability include the time and number of sorties to transport a fully equipped battalion.

2.2.2.3.2 Unmanned Ground Vehicle (UGV) Ilities

Affordability. Affordability is a concern for all systems.

Force Protection is not a system ility for UGVs. The function of a UGV is to enable troops to accomplish effects remotely, out of harm's way, and thus to provide force protection. But it is not a system ility.

Survivability has not been a significant concern for UGVs. While not considered a consumable, they are low-cost compared to manned vehicles. Damage and loss of function does not directly put troops at risk. The increase in cost and degradation in performance from survivability systems mitigates against their inclusion on UGVs.

Usability is a major consideration for UGVs. Remote control with direct overwatch is the preferred mode of control, but this limits the range and exposes the operator to greater risk. Teleoperation viewing through the on-board camera but without direct overwatch is more stressful, with limited situational awareness and navigation awareness. At the present time, autonomous navigation methods are not trusted and have not demonstrated reliable operation or Technical Readiness Level 8 or 9. One operator controlling multiple UGVs in formations or team operations is a desired capability that is also not available on fielded systems.

Versatility for UGVs is a concern at the platform level (modular appendages) and at the aggregate level of robotic ground systems. The Robotic Systems Joint Program Office concept is for a family of platforms (of different sizes and hence mobility) and a family of scalable payloads. Payloads can be software-only or combinations of software and hardware.

Mobility concerns for UGVs has some different nuances from manned systems. Range and endurance are major concerns for battery-powered systems. Obstacle detection has low operational reliability under teleoperation due to limited perception. Obstacle crossing also has low operational reliability.

Capacity. Capacity refers to the ability of UGVs to carry alternate appendages and payloads. It includes exterior surface areas where appendages can be attached without interfering with system functions. For UGVs, payloads include software that requires processing capacity.

Interoperability concerns for UGVs include ability to maneuver with manned vehicles (unless they are small and transported on manned vehicles), common operator control systems.

Operational Reliability, Availability and Maintainability (RAM) concerns for UGVs include degradation of battery capacity, recharging time, as well as operational failures. Unexpected loss of power due to degraded batteries and/or insufficient charge are concerns.

Security issues for UGVs are amplified to include physical security – kidnapping – and cyber-security - jamming, intercepting video feedback, and seizing control remotely.

Transportability is a concern for UGVs – whether they are man-packable, man-portable, or transported by truck. Transportability concerns include the ability to load and unload under their own power.

2.2.2.4 Ility and Affordability Tradespace Analysis Needs

Ground vehicles are the embodiment of tradeoffs. Reserve capacity (design margin) increases initial cost, size and weight, but can lower the life cycle cost and extend the operational lifespan. Intrinsic survivability – armor and shaping – reduce mobility. Mobility and intrinsic survivability both contribute to force protection. Capacity, mobility and intrinsic survivability all contribute to increased size and weight, which decrease transportability. Improved transportability improves force protection by putting more force in place faster.

2.3 SHIP DOMAIN (NPS, WSU)

The NAVSEA Acquisition Guide (2010) endorses evolutionary acquisition to deliver initial capability with the explicit intent of delivering improved or updated capability in the future, providing the warfighter with an initial capability that may be less than the full requirement as a trade-off for earlier delivery, agility, affordability, and risk reduction. Tradespace MPT are needed to balance earlier delivery, agility, affordability, and risk reduction. Performance requirements, although identified early on, change over the course of time. Development, with tradeoffs made in the areas of design, logistic supportability, and affordability. Modifications to the operational requirements may be required, and are encouraged, once it has been determined that the initial requirements were too optimistic or that the cost
to achieve the required levels will be prohibitive. The concept of balancing performance with that of cost and schedule requirements is called CAIV (Cost as an Independent Variable).

The Naval Sea Systems Command (NAVSEA), in support of the Secretary of the Navy (SECNAV) and Chief of Naval Operations (CNO), is specifically aimed at bringing the 313-ship Navy from concept to reality. With the defense budget already strained, Navy recapitalization dollars are split between ships and aviation, and acquisition programs over cost and schedule. When he assumed command of NAVSEA, Vice Admiral Kevin M. McCoy challenged the command's headquarters and field activities, including Carderock Division, to focus on three NAVSEA

Strategic Business Plan goals: (1) Sustaining today's fleet efficiently and effectively, (2) Building an affordable future fleet, and (3) Enabling the warfighters and workforce. NAVSEA initiatives addressing sustainment are summarized in the 2009 volume 5 of SEAFRAME ("Sustaining the Fleet). 2010 volume 6 of SEAFRAME summarizes initiatives toward "Building an Affordable Fleet" including specific tradespace needs and approaches, and use of physics based simulation and analysis tools.

NAVSEA has initiated and executed several projects related to improving the development of effective and affordable long-lived ships. The Ship-to-Shore Connector project demonstrated set based design to refine the procedures and examine their potential benefits with regard to cost control and ship flexibility. Premature retirement of ships in the U.S. fleet in recent decades has elevated design flexibility as one of the most important of design attributes. A ship with a planned 30-year service life must be able to be refitted (or reconfigured) to embrace evolving or new systems without exorbitant cost. When the cost becomes too high to upgrade, maintain or refit, the ship is often retired prematurely. Demands for a more agile force are also driving needs for increased design flexibility.

2.4 SPACE VEHICLE DOMAIN (MIT)

Operating systems in space, with its associated "high-ground" perspective, enables capabilities otherwise not available or possible using terrestrial systems. Space provides an opportunity for platforms from which systems can affect lives on a global scale through enhancements to navigation and timing, communications, and remote imagery. Sending systems to space also allows humans to conduct remote missions not possible from Earth, such as near and deep space science observation and in situ exploration. Space-based capabilities impact not only government agencies, but also civilian consumers as well, with such systems as GPS, ground imaging, satellite radio, and TV service. While enabling the use of such technologies, the development and operation of space systems comes with a high cost, especially due to difficulty in accessing the space segments of these systems after launch. Manufacturing precision, experimental technology, international cooperation, vehicle launch use, on-orbit operations, regulation compliance, unforgiving environments, and many other factors generate high risk and high costs for most space systems. In response to these challenges, space systems tend to be very complex and expensive, and often are designed to operate for long periods of time. The impact of changes in the system environments, and the ability of a system to effectively respond to these changes, could mean the difference between success and failure (Beesemyer, Ross, and Rhodes 2012).

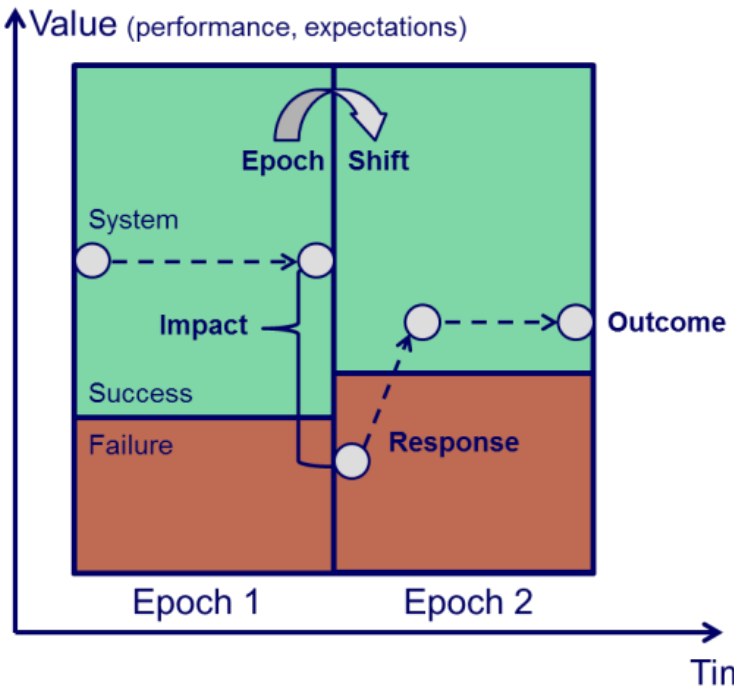


Figure 31. Epoch shift - Impact - Response - Outcome Construct (Beesemyer, Ross, Rhodes 2012)

Using a construct similar to the Epoch-Era Analysis representation presented in an earlier section, system case examples can be discussed using the Epoch shift—Impact—Response—Outcome construct as seen in Figure 31. The figure describes how a system may be operating at an acceptable level of performance in Epoch 1 and then experience an *epoch shift*. After experiencing this imposed shift in system, context, or needs, the system may display some degradation in performance, known as the *impact*, possibly bringing performance below acceptability levels. The system then, in order to recover to an acceptable performance level, may initiate a *response*, which then results in some *outcome* for the system. This construct is proposed and tested in Beesemyer, Ross, and Rhodes (2012) to determine the feasibility of using such a construct for collecting and comparing historical space systems for insights regarding the "success" of system responses relative to epoch shifts. The ultimate intention is to provide prescriptive advice about potentially good "responses" (i.e. embodied "ilities") for new systems as a function of possible epoch shifts that the system may face over its lifetime. Figure 32 below provides a simple space-based illustration of Epoch Shift-Impact-Response-Outcome.

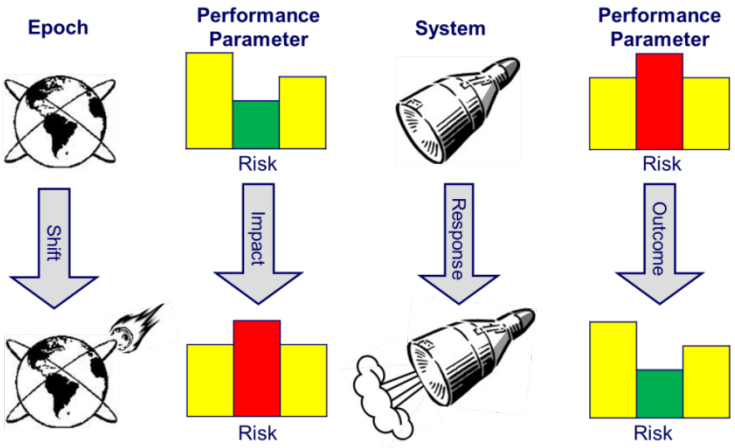


Figure 32. Simplified Epoch shift--Impact--Response--Outcome Example (Beesemyer, Ross, Rhodes 2012)

One of the key observations in the research was that a number of different ility term labels apply to system responses and are a function of whether a "system parameter" is changing or not changing, or if an "outcome parameter" is changing or not changing. For example, if one desires a system parameter (e.g. types of satellites in a constellation) to remain fixed, a "no change", but desires that the expressed functionalities of the constellation to change (i.e. the outcome parameter to change), then one desires a "versatile" constellation. This simplified taxonomy is illustrated in Figure 32, and is related to the semantic basis descried in "change-oriented view" earlier in this report.

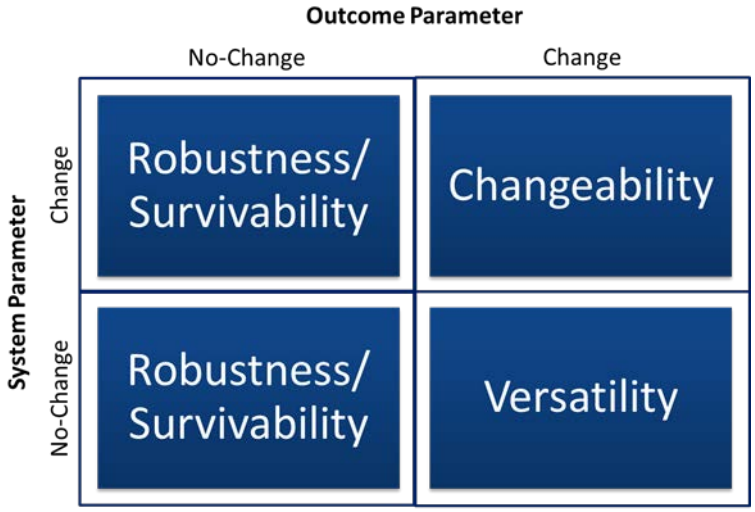


Figure 33. Typology of System Parameter vs. Outcome Parameter Change or No-Change to Achieved Desired Quality in Outcome Parameter (Beesemyer 2012)

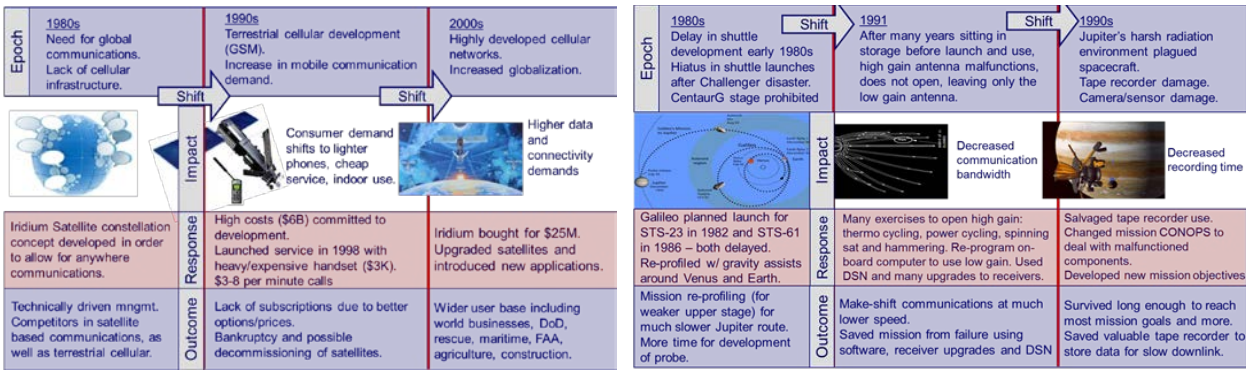


Figure 34. Iridium (left) and Galileo (right) Epoch shift—Impact—Response—Outcome Snapshot (Beesemyer, Ross, Rhodes 2012)

An illustration of Epoch Shift-Impact-Response-Outcome for the Iridium and Galileo cases considered in Beesemyer, Ross, and Rhodes (2012) are shown in Figure 34. In the former case, Iridium failed to respond to shifts in demand and competition and ultimately went bankrupt, before emerging as "successful" after writing off their losses. Galileo faced environmental challenges but was able to successfully respond through software and operations flexibility. A summary of these two cases, as well as Globalstar and Teledesic are shown in Table 5 below.

Table 5. Epoch shift—Impact—Response—Outcome Summaries (Beesemyer, Ross, Rhodes 2012)

System	Shift	Impact	Response	Outcome
Iridium	Cellular development	Low subscription	None	Failure
Iridium	Increased communications	More data demand	Bankruptcy/ Target niche markets	Success
Globalstar	Cellular development	Low subscription	Cheaper architecture	Failure
Globalstar	Increased communications	More data demand	Bankruptcy/ Target niche markets	Success
Teledesic	Terrestrial data development	Decreased demand	Scaled down	Failure/ Success
Teledesic	Iridium/ Globalstar bankruptcies	Decreased investment	Cease work	Failure/ Success
Galileo	High gain antenna failure	Decreased bandwidth	Tech/ Ops re-work	Success
Galileo	Component damage	Decreased performance	Ops/ objective re-work	Success

The Table 5 summary shows how Iridium and Globalstar failed as initial systems, but ultimately succeeded in system deployment through the use of bankruptcy. Teledesic both succeeded and failed in both epoch shifts since the system was never developed, but large amounts of downside losses were avoided. Success or failure depends on the criteria being used (providing

value as a system or providing profits or minimizing losses). Finally, Galileo shows an example of success in response to both epoch shifts.

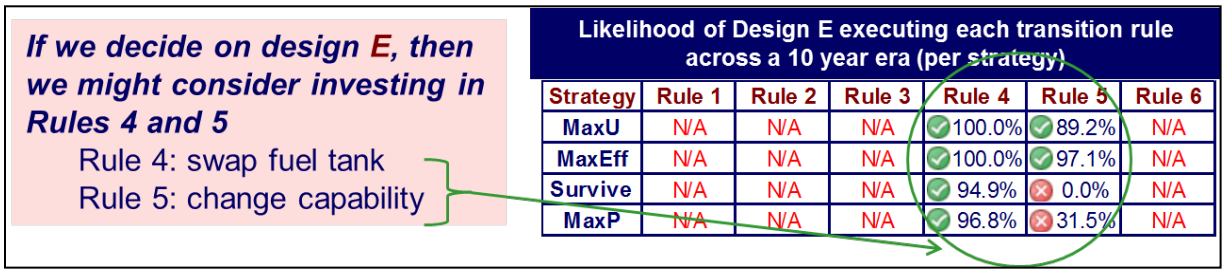
When discussing the success of some of the above described programs, the presence of ilities has been referenced as serving a pivotal role (e.g. flexibility or evolvability). Nilchiani (2005) and Saleh, Hastings, and Newman (2003) attribute much of Galileo's success to its "flexibility" in design. Similarly, de Neufville and Scholtes (2011) would label the Teledesic response and outcome as flexibility, scaling the architecture during system development with increased contextual knowledge. These cases show how varying responses to epoch shifts may lead to different outcomes in value sustainment. If the presence of ilities is attributed as a key successful quality in historical systems, they may be useful concepts for designing systems that are capable of value-sustainment in the future. Additional cases may be examined using the Epoch Shift – Impact – Response – Outcome, potentially revealing patterns of successful responses to epoch shifts. Examples of other space programs could include DirecTV, GPS, SBR, Mars Polar Lander, Milstar, TerreStar, ORBCOMM, Hubble Space Telescope, and the ISS, among others.

In addition to considering ilities as system responses to epoch shifts in a qualitative, descriptive manner illustrated using the Epoch Shift-Impact-Response-Outcome construct, one can also apply Epoch-Era Analysis and change-related ilities concepts in a quantitative manner. An example of this is the recent Valuation Approach for Strategic Changeability (VASC), which operationalizes Epoch-Era Analysis for the purpose of valuating changeability in tradespace studies.

VASC codifies an approach to value changeability by taking into account both the magnitude and counting value of having the ability to change a system. The magnitude value refers to the degree to which a destination (changed) system is better than its origin (unchanged) state, while the counting value refers to the benefit of having multiple possible change paths available. Magnitude value can only be evaluated when one knows the use context for the changed system, while the counting value is only beneficial in that it increases the likelihood that a high magnitude destination state may be accessible when desired. In addition to these two types of value experienced by having changeability, VASC uses the concept of "strategy" which specifies a logic for when and how changeability would be leveraged (i.e. the contingent objective function used to decide when to execute a change option, such as "execute a change option that will increase the bandwidth of my system if the demand increases above some threshold").

The five step procedure of VASC (described in Fitzgerald, Ross, and Rhodes 2012), is applied to a Space Tug satellite system to demonstrate its ability to quantify the costs and benefits of including various degrees of changeability into a system, allowing for tradeoffs with more classic metrics such as performance and cost. Various Epoch-Era Analysis screening metrics are used to identify designs of interest in order to perform more detailed time-dependent analysis

of how and when change options are executed over the Space Tug lifetime. This allows for both short run and long run analysis of the value of having changeability in the system. An example of changeability tradeoffs for the long run is illustrated in Figure 35 below, where each "rule" represents different change options, such as "swap fuel tank."



Likelihood of Design E executing each transition rule across a 10 year era (per strategy)

Strategy	Rule 1	Rule 2	Rule 3	Rule 4	Rule 5	Rule 6
MaxU	N/A	N/A	N/A	100.0%	89.2%	N/A
MaxEff	N/A	N/A	N/A	100.0%	97.1%	N/A
Survive	N/A	N/A	N/A	94.9%	0.0%	N/A
MaxP	N/A	N/A	N/A	96.8%	31.5%	N/A

Figure 35. Space Tug Design E Rule Usage by Strategy across a 10 Year Era (Fitzgerald, Ross, Rhodes, 2012)

Additional examples of VASC applied to space systems, including a neutral atmospheric density sampling mission (X-TOS) and a space-borne radar constellation (Satellite Radar System) can be found in Fitzgerald (2012). One of the emergent benefits of the proposed metric set and VASC approach is the ability to directly compare change-enabled value sustainment with robustness-enabled value sustainment. For example, the "effective fuzzy Normalized Pareto Trace" metric evaluates how cost-utility efficient a system alternative is across a large number of potential uncertainties, while taking into account execution of change options. If change options result in "better" destination states within a given context, then those options are more valuable than options that do not. In this way, the "effective" insensitivity of a system to uncertainties, whether by withstanding imposed changes (i.e. "robustness") or by changing (i.e. adaptability or flexibility), can be evaluated. Table 6 below shows summary results of the X-TOS system Era analysis of seven designs of interest, deemed to be most value sustaining. Designs 2535 and 7156 do not execute any change options and are therefore classically "robust," while the other designs execute various change options, and are therefore "flexible" or "adaptable", depending on the nature of their executed change options. (For this particular example, the designs with shortest 'time cost' are those executing adaptable-type change options and can therefore change more quickly.)

Table 6: Statistical Summary of X-TOS Era Modeling (N=1000), (Fitzgerald, Ross, and Rhodes 2011)

Design #	903	1687	1909	2471	2535	3030	7156
Success %	75.3	75.8	75.0	75.2	73.6	75.6	78.0
Avg $ cost	0	2.49K	6.35M	0	0	3.15M	0
Avg time cost	13 min	17 min	4.27 mo.	14 min	0	4.97 mo.	0
Avg # transitions	0.65	0.85	2.00	0.65	0	1.90	0

While much of the preceding research described in this section has been applied to space systems, ongoing research is applying these MPTs to other domains, such as transportation

systems (Nickel 2010), maritime security system of system (Ricci, Ross, Rhodes, and Fitzgerald 2013), and non-transport ships (Gaspar, Erikstad, and Ross 2012).

References

Beesemyer, J.C., “Empirically Characterizing Evolvability and Changeability in Engineering Systems,” Master of Science Thesis, Aeronautics and Astronautics, MIT, Cambridge, MA, June 2012.

Beesemyer, J.C., Ross, A.M., and Rhodes, D.H., "Case Studies of Historical Epoch Shifts: Impacts on Space Systems and their Responses," AIAA Space 2012, Pasadena, CA, September 2012.

de Neufville, R. and Scholtes, S., Flexibility in Engineering Design, MIT Press: Cambridge, MA, 2011.

Fitzgerald, M.E., Managing Uncertainty in Systems with a Valuation Approach for Strategic Changeability, Master of Science Thesis, Aeronautics and Astronautics, MIT, June 2012.

Fitzgerald, M.E., Ross, A.M., and Rhodes, D.H., "A Method Using Epoch-Era Analysis to Identify Valuable Changeability in System Design," 9th Conference on Systems Engineering Research, Los Angeles, CA, April 2011.

Fitzgerald, M.E., Ross, A.M., and Rhodes, D.H., "Assessing Uncertain Benefits: a Valuation Approach for Strategic Changeability (VASC)," INCOSE International Symposium 2012, Rome, Italy, July 2012.

Gaspar, H., Erikstad, S.O., and Ross, A.M. (2012), "Handling Temporal Complexity in the Design of Non-Transport Ships Using Epoch-Era Analysis," *Transactions RINA, Vol. 154, Part A3, International Journal of Maritime Engineering*, Jul-Sep 2012, pp. A109-A120.

Nickel, J. (2010), *Using Multi-Attribute Tradespace Exploration for the Architecting and Design of Transportation Systems*, Master of Science Thesis, Engineering Systems Division, MIT, February 2010.

Nilchiani, R., "Measuring the Value of Space Systems Flexibility: A Comprehensive Six-element Framework," Ph.D. Dissertation, Aeronautics and Astronautics, MIT, Cambridge, MA, June 2005.

Ricci, N., Ross, A.M., Rhodes, D.H., and Fitzgerald, M.E. (2013), "Considering Alternative Strategies for Value Sustainment in Systems-of-Systems," 7th Annual IEEE Systems Conference, Orlando, FL, April 2013.

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, December 2003, pp. 927-944.

2.5 COSATMO (USC)

Recent interactions among the SERC, the Air Force Space and Missile Systems Center (SMC), the National Reconnaissance Office (NRO), and the Aerospace Corporation, including the SMC’s co-sponsorship of SERC RT-6, “Software Intensive Systems Data Quality and Estimation

Research in Support of Future Defense Cost Analysis," have led to the exploration of a more general collaborative effort in the space system cost estimation area. In particular, the exploration has focused on prospects for the development of a next-generation, full coverage (cyber-physical-human; flight-ground-launch capabilities; definition-development-operations-support life cycle total ownership cost coverage) cost model for satellite systems called COSATMO, developed in a way that much of it could be used to develop similar models for next-generation ground, sea, and airborne systems cost estimation.

Current tradespace and affordability analysis capabilities in the space domain and elsewhere in DoD are generally focused on either physical or cyber/software systems, with limited ability to address impacts of one on the other, and limited coverage of human and economic concerns, even for current DoD systems. For future DoD systems, Chapter 7 of the SERC RT-6 draft Air Force Cost Analysis Agency "Software Cost Estimation Metrics Manual" identifies general future trends for which current software estimation capabilities will be inadequate, and identifies approaches for addressing them . These trends include:

1. Rapid change, emergent requirements, and evolutionary development;
2. Net-centric systems of systems;
3. Model-Driven and non-developmental item (NDI)-intensive systems
4. Ultrahigh software system assurance;
5. Legacy maintenance and brownfield development; and
6. Agile and lean systems engineering and development.

In addition, DoD satellite systems will encounter increased levels of threat that will require further investments in system security and physical self-defense, the increasing attractiveness of nanosensor-driven smart systems and 3D printing capabilities, and changes in social networking capabilities and workforce skills.

These trends will also challenge future physical and human system-element cost estimation, and the approaches in Chapter 7 will provide a starting point for addressing them and their interactions with each other. This systems approach to the integrated cost estimation of integrated cyber-physical-human systems and systems of systems is what makes COSATMO the core of our general approach to the tradespace and affordability analysis of DoD systems of the future.

The justification for starting with total satellite systems is that this community understands and is willing to support the definition and development of such a capability. Initial discussions have identified an overall incremental research and development strategy prioritized on strength of need and availability of potential starting points for initial models and calibration data. Two well-attended technical interchange meetings have begun to identify need priorities and current sources of estimation capabilities and calibration data. A proposal for funded development of initial capabilities has been submitted as an iTAP Phase 2 option.

PHASE 2 AND 3 PLANS

A summary of the overall plans is provided next, followed by short summaries of Phase 2 plans of each iTAP organization

Table 7 shows the focus, deliverables, and investment in iTAP through 2016. The timeline beyond 2016 has not yet been established. The iTAP program has two primary initial foci, and a third educational focus once a critical mass of iTAP capabilities have been built up.

The first initial focus is on researching and developing the foundations of ilities Tradespace and Affordability (IT&A) analysis via a framework of IT&A views that aid in organizing and applying IT&A analysis to address the systems engineering of cyber-physical-human systems. The views include DoD stakeholder value-based ility definitions, relationships, and priorities; means-ends views for achieving individual ilities; architectural strategies for achieving individual ilities and their second-order impacts on other ilities; process strategies for incrementally addressing uncertainties in ility tradespace situations, and for concurrently balancing a cyber-physical-human system's ility aspects; domain-specific ility tradespace views; and system of systems views, including challenges in scalability and in reconciling the incompatible assumptions of component-system domain-specific architectures.

The second initial focus is on extending and integrating existing IT&A MPTs to better support DoD cyber-physical-human system ility analysis. This will include developing more service-oriented and interoperable versions of current SERC ility MPTs; developing approaches for better integrating MPTs primarily focused on physical, cyber, or human system IT&A analysis; efforts to modify and compose existing SERC ility IT&A MPTs to better interoperate with each other and with counterpart MPTs in the ERS community and elsewhere; and efforts to apply the MPTs to the IT&A analysis of increasingly challenging DoD systems. In the affordability area, a particularly promising prospect is a collaborative SERC-Aerospace Corporation-USAF/SMC-NRO effort to develop an integrated lifecycle cyber-physical-human system cost model for satellite systems, including the flight, ground, and launch systems, which could be subsequently extended to other DoD domains.

Table 7. iTAP Project Timeline

Year	Focus	Key Deliverables	Core Invest-ment	Co-Investment
Pre-2014	Research and develop basic iTAP concepts and framework. Explore early MPT applications and interoperability, including with ERS counterparts.	Basic iTAP concepts and framework. Results of early MPT applications and interoperability improvements. Prototype integrated lifecycle cyber-physical-human system cost model. Multi-view IT&A analysis guidance papers.	$284K	$250K from DoD Services to tailor and support early iTAP MPT applications, integrate with ERS counterparts.
2014	Elaborate iTAP concepts and framework in key areas e.g., systems of systems. MPT extensions; broader and deeper applications and interoperability. iTAP new-idea explorations.	Elaborated iTAP concepts and framework. Results of broader and deeper MPT applications. Integrated lifecycle cyber-physical-human system cost model domain-specific IOC. Multi-view IT&A Analysis Guidebook v 0.5; associated papers.	$900K	$500K from DoD Services to tailor and support broader and deeper iTAP MPT applications, integrate with ERS counterparts.
2015	Mature iTAP concepts, framework, Guidebook. Increasingly scalable and interoperable MPTs. Extensions of iTAP new-idea explorations. Guidebook-based outreach and educational initiatives	Mature iTAP concepts and framework. Results of increasingly scalable and interoperable MPT applications. Extended domain-specific lifecycle cyber-physical-human system cost model; prototypes in other domains. Multi-view IT&A Analysis Guidebook v 1.0; Guidebook-oriented courseware, early usage at AFIT, NPS, DAU, SERC universities	$900K	$750K from DoD Services, other agencies to tailor and support scalable and interoperable iTAP MPT applications, integrate with ERS counterparts, and to develop iTAP educational technology.

Year	Focus	Key Deliverables	Core Invest-ment	Co-Investment
2016	Integration of new-idea explorations into iTAP concepts, framework, Guidebook. Increasingly scalable and interoperable MPTs. Further iTAP new-idea explorations. Guidebook-based outreach and educational initiatives	New-idea-extended iTAP concepts and framework. Results of increasingly scalable and interoperable MPT applications. Extended multi-domain lifecycle cyber-physical-human system cost model; Multi-view IT&A Analysis Guidebook v 1.1; Guidebook-oriented courseware, broad usage at AFIT, NPS, DAU, SERC , and other universities	$720K	$1M from DoD Services, other agencies to tailor and support scalable and interoperable iTAP MPT applications, integrate with ERS counterparts, and to develop iTAP educational technology.

AIR FORCE INSTITUTE OF TECHNOLOGY - PHASE 2 PLANS

AFIT has engaged with an AFMC training organization in need of iTAP capabilities to structure and plan a project involving next-generation training capabilities. Details are being worked out.

GEORGIA INSTITUTE OF TECHNOLOGY - PHASE 2 AND 3 PLANS

Goals & Objectives

Phase 2 Goal: Define roadmap to integrate advanced SE methods developed by SERC members into GTRI's collaborative web-based systems engineering framework. The expected to-be end state includes:

- Identification of relevant methods for implementation
- A test case that proves methods end-to-end for specific application or test case
- Prototype software tool to demonstrate for a specific pilot user/early adopter

Phase 3 Goal: Develop software implementation suitable for distribution and preliminary evaluation by target customers. The expected to-be end state includes:

- Closing gaps and implementation of critical feature requests identified in Phase 2
- Fully functional tool suitable for distribution to wider audience for trial testing and new feature identification

- Distributed to multiple users in other domains beyond the pilot phase

Domains of Cyber-Physical Systems

JCIDS is concerned with operational systems producing effects on their environment, or for their stakeholders. Levels of performance will vary with respect to that environment, and may lead to untenable situations. Failure is measured by the perspective of the stakeholder within that operational environment, and no one size fits all definitions exist for these –ilities. Therefore –ility prioritiy will vary by DoD operational system life cycle roles, primary type of system (physical, informational, or both), by system of interest level (component, subsystem, system, system of systems), and by primary type of operational platform. There is therefore a need to capture dynamic and changing priorities; the Web-based SE tools framework should be cross-compatible across domains by design.

JCIDS stresses capability, therefore a full CPS includes air, sea, land, and space. These Heterogeneous SoS implies the "Human controller or mixed initiative"/semi-autonomous class of autonomous systems. Any decision support framework must therefore handle characterizations of these various levels of control.

iTAP Shortfalls Addressed

The iTAP team is composed of universities with varying expertise in both basic and applied research. The goal of the GTRI effort will be transitioning the academic/theoretical characterization of SoS to application toolset. The DoD definition of SoS Architectures will be implemented as gaps/needs/technologies change. Autonomy is effectively trust between manned and unmanned systems, and is built through experience; this may be captured as an evolving characteristic of the SoS architecture as the operational application evolves.

Proposed Approach (Summary)

The proposed approach involves integrating advanced SE methods into GTRI's web-based Framework; those methods be developed through any of the SERC partners. A collaborative environment (deployable across wide geographies) would provide a familiar interface suitable for both technical and non-technical users, and integrated MBSE at the SoS level will define the problem in a solution agnostic manner.

A definition of "hierarchy" will provide a mapping of Needs to Capability areas to domain to attributes (-ilities), with a definition of "problem" down to how one would characterize the solution. The toolset will then be used to guide decision transition solutions over time, and increase system flexibility over time to adapt to changes in system robustness and adaptability.

Risks

Possible risks exist in bridging the gap between theoretical methods and applications. Potentially unforeseen deficiencies in methods may limit their immediate applicability without further refinement. Availability of data necessary to populate algorithms may be problematic. It is also difficult to estimate computational requirements of methods a priori for full scale test cases. Discrepancy between understanding levels of autonomy in the definition of CPS may confound proper scoping.

MASSACHUSETTS INSTITUTE OF TECHNOLOGY - PHASE 2 AND 3 PLANS

Phase 2 Plans.

In Phase 2, MIT intends to contribute to two main tasks: research and development of iTAP scientific foundations, and iTAP MPT piloting, extending, and transitioning. In the first task, MIT will contribute in support of efforts which may include ility definitions, frameworks, strategies, and principles, leveraging MIT past performance in development of such constructs. This may include extension of research into metrics for change-related ilities, such as flexibility, adaptability, and evolvability, as well as approaches for exploring tradespaces of ilities incorporating Epoch-Era Analysis-based techniques. Decision-making considerations on how affordability can be traded against change-related ilities and performance will be investigated, along with methods for real-time interrogation of tradespaces. The latter will lay the foundation for further work in interactive tradespace exploration for multi-stakeholder negotiation. Investigation of existing decision practices and metrics for multi-stakeholder tradespace exploration and decision making, with demonstration of preliminary MPTs related to how ilities can be incorporated into multi-stakeholder decision making tradespaces.

In the second task, MIT will contribute to efforts of the RT-46 research team to pilot, extend, and transition iTAP MPTs, given selected pilots (for example Army CRES-GV or CREATE-SHIPs). These efforts may include support for application of ilities metric calculations, tradespace formulations, and expanded tradespace analysis across multiple scenarios using Epoch-Era Analysis.

Phase 3 Plans

In Phase 3, MIT will continue the two path approach of contribution to “foundations” as well as application cases to help with “MPT deployment.” For the foundation work, we intend to develop methods for near real-time interrogation of tradespaces for multiple ilities within a multi-stakeholder negotiation context. As interpretation and valuation of tradeoffs among performance, affordability, and other ilities requires feedback from critical stakeholders, this phase will leverage interactive tradespace exploration techniques in order to rapidly identify

the fundamental ilities tradeoffs within a system design and development effort. In particular, where ilities, including affordability, as well as performance, will be traded, MPTs that utilize tradespaces must allow for data representation, interaction, and analysis that enhances effective decision making. Methods and metrics developed in this phase will be validated using experiments within a laboratory environment, as well as through case applications with a DoD-class system. Prototype tools may be developed in order to streamline the tradespace exploration analysis approaches developed in this phase. MIT will continue to collaborate with other RT-46 team members to seek opportunities for synergies and cross-university foundations contributions as well as case application of MPTs.

NAVAL POSTGRADUATE SCHOOL - PHASE 2 AND 3 PLANS

The NPS Phase 2 activities are to improve and pilot several existing ITA analysis toolsets based on the results of Phase 1. The initial focus for iTAP MPT extensions and applications is in the Ships and Ground vehicles domains. Additional Phase 2 activities may extend the results of the Phase 1 framework efforts or related new ideas, and continue integration and community-building activities with ERS and others.

The desired goals of NPS Phase 2 Task 2 research are:

- Experiment with tailoring existing or new tradespace and affordability MPTs for use by an early adopter organization
- Train early adopters in its use, monitor their pilot usage, and determine areas of strengths and needed improvements, especially in the MPTs' ilities
- Extend the MPTs to address the top-priority needed improvements
- Work with early adopters to help transition the improved MPTs into their use
- Identify and pursue further improvements for the early adopters or for more general usage
- Partial completion of the steps is acceptable for complex and highly desired capabilities.

For Phase 2 Task 3 we will participate in planning for Phase 3, based on progress on Tasks 1 and 2, and on emerging needs, opportunities and priorities.

At the beginning of Phase 2 we are assessing to pilot MPTs in the Navy Ship domain with the CREATE-SHIPS program. By the end of Phase 2, the plan is to have created and piloted useful ITA capabilities, along with establishing the initial foundations for overall ITA analysis and an ITA knowledge base.

Phase 2 will thus culminate our initial capabilities to be ready for major applications and extensions in Phase 3 and beyond.

Phase 3 will be as increasingly application domain focused. MPTs will be instantiated in multiple domains based on Phase 2 pilot experiences. Specific domains with interested sponsors span ships, air vehicles, C4I and satellites .

The CREATE-SHIPS application with NAVSEA will logically be followed with CREATE-AV at NAVAIR Pax River, where NPS has a substantial embedded presence. The other planned applications are for C4I at SPAWAR and space satellites on the COSATMO joint task.

These experiences on ships, air vehicles, C4I and satellites will also support cross domain mapping and analysis of MPTs. We can find leverage points for commonality of MPTs across DoD segments and help identify global vs. domain-localized application fits.

These prime target opportunities will help build and learn overall ITA and T&A analysis methods for the larger DoD user community.

UNIVERSITY OF SOUTHERN CALIFORNIA - PHASE 2 AND 3 PLANS

In Phase 2, USC will complete the value-oriented means-ends ility hierarchy and set of definitions, and coordinate them with the MIT change-oriented ility hierarchy. USC will collaborate with U. Virginia in completing the strategy-based ility strategies and conflicts cross-impact matrix, and in defining a web-based tool for use in system architecture planning and definition activities. USC will work with MIT and U. Virginia to develop a draft unified set of iTAP foundations, including characterization of the relationships among the various views.

USC will also provide cost modeling capabilities for the various CREATE-Ships, CRES-GV, and other SERC efforts to provide ility tradespace and affordability analysis capabilities to ERS and other DoD projects. Contingent on funding, USC will collaborate with NPS to prototype initial COSATMO capabilities, based on need priorities and available existing capabilities and calibration data.

In Phase 3, USC will continue to elaborate and integrate the various iTAP Foundations capabilities; support their application to ERS and other DoD projects; develop an Initial Operational Capability version of COSATMO, again contingent on funding; and lead the development of an 0.5 version of a multi-view ilities Tradespace and Analysis Guidebook.

UNIVERSITY OF VIRGINIA – PHASE 2 AND 3 PLANS

Phase 2 will develop the initial thrusts beyond early prototype stage, and will also include collaborative work with the USC team on formalizing a broader space of ilities and tradeoffs than UVa has addressed so far. Phase 3 will involve the substantial development of methods, processes, notations, and tools with high potential to be transitioned into assessment and use in practical settings.

WAYNE STATE UNIVERSITY - PHASE 2 AND 3 PLANS

Phase 2 Plans

Phase 2 plans are to conduct research into the iTAP capabilities and priorities with respect to the ground domain in general, integration with SE systems, and application to example "case study" applications. Phase 2 plans also include coordination and synchronization of MPT between the ground and sea domains.

In Phase 2 we will initiate application, evolution, and evaluation of iTAP MPT in the context of one or more ground vehicle systems. There are a number of potential opportunities, with different benefits and limitations as case study applications:

CRES-GV is a High Performance Computing project related to use of physics based simulation for ground vehicle design. This project is still formulating it technical objectives and approach.

DARPA has engaged TARDEC in the "Ground Vehicle – Experimental" (GVX) project. The intent is to think "out of the box" as to how mechanized infantry protected mobility is provided, considering novel capabilites and tradeoffs.

DARPA has also engaged TARDEC in the "Fast Adaptable Ground Vehicle" (FANG) project. The FANG mission and requirements are modeled on the Marine Corps Amphibious Combat Vehicle (ACV), a pre-Materiel Development Decision program. The DARPA FANG project is independent of the ACV. It is explicitly concerned with development methods that reduce the development and adaptation time, including the number, time and cost of engineering change proposals in modern, highly computerized vehicles. Modeling engineering change order costs and impact of improved development methods is a particular interest.

There are several major ground vehicle acquisition programs in the formal acquisition process: JLTV, PIM, GCV and AMPV. These programs are not good candidates for direct involvement because of procurement and competition sensitivities. The Marine Corps ACM and MPC, while not yet approved by the decision authority, have similar sensitivities. However they might make good models for hypothetical case studies (e.g., the DARPA FANG project).

TARDEC's Advanced Systems Engineering Capability (ASEC) is in use on a variety of programs. It could provide a framework to transition SERC MPT into actual use on major defense

acquisitions. In Phase 2 we will investigate needs, opportunities and limitations on iTAP MPT with regard to downstream integration with ASEC. ASEC is being or is scheduled to be used on the aforementioned projects.

Other case study opportunities include retrospectives of historical programs such as the Abrams tank, the HMMWV, the M113 or Bradley, or even the integration of the FBCB2 computer system across the fleet. Ability to obtain cost and architecture data may be a limitation.

Phase 3 Plans

In Phase 3, we will continue application case studies and MPT refinement, focusing on the ground and sea domains.

In Phase 3 we will adapt/tailor iTAP MPT for architecture-based design methods. Architecture-based design combine a generic architecture, model-based design, and set-based design for progressive articulation and refinement of the system concept. Architecture-based design is the foundation of ground vehicle development.

In Phase 3 we propose to develop iTAP methods for early systems engineering, specifically pre-Materiel Development Decision (MDD), and after MDD, developing concepts for the Analysis of Alternatives (pre-Milestone A). Pre-MDD iTAP will focus analyzing the feasibility of capabilities versus costs, tradeoffs of deferred capabilities and risk. We will develop methods to partition the tradespace to help generate a diverse set of alternatives for the Analysis of Alternatives. These MPT will be developed to be domain-agnostic, but will be tested and demonstrated with ground vehicle applications.