Systems 2020

Strategic Initiative

Final Technical Report SERC-2010-TR-009

August 29, 2010

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

Contributors: Jennifer Bayuk, Abhi Desmukh, Robert Graybill, JoAnn Lane, Alan Levin, Azad Madni, Mike McGrath, Art Pyster, Stas Tarchalski, Richard Turner, and Jon Wade

Contract Number: H98230-08-D-0171 , DO 002, TO 0002, RT 0020

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
29 AUG 2010		**00-00-2010 to 00-00-2010**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Systems 2020: Strategic Initiative	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) **Stevens Institute of Technology,Systems Engineering Research Center (SERC),1 Castle Point on Hudson,Hoboken,NJ,07030**	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
SERC is sponsored by the Department of Defense.

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)**	**88**	

Table of Contents

Figures

Tables

1.0 Executive Summary

The Department of Defense (DoD) increasingly faces a mix of relatively foreseeable and unforeseeable threat and opportunity profiles. This means that DoD technological superiority relies on rapid and assured development, fielding, and evolution of progressively more complex and interoperable defense systems. Meeting these challenges requires DoD to design and build an entirely new class of adaptive systems that allow the Department to operate with far greater speed and agility. Mr. Lemnios, the Director, Defense Research and Engineering (DDR&E), requested a study of systems engineering research areas that enable agile, assured, efficient, and scalable systems engineering approaches to support the development of these systems. This report addresses the four highest-potential research areas determined by the study: Model Based Engineering (MBE), Platform Based Engineering (PBE), Capability on Demand(COD), and Trusted System Design (TSD). It elaborates each research area, characterizing them in terms of current state of the art and state of the practice, and identifies the most promising research topics. It then proposes next steps to create a DoD-wide Systems 2020 initiative based on a coordinated set of high-leverage, game-changing activities comprising systems engineering research, technology maturation, and pilot-based transition into practice.

1.1 Need for Systems 2020

Systems in the design phase today may not be fully deployed for 10-20 years. Given these long gestation times, the probability is extremely low that the threat environment and technological solution will remain as envisioned by the time the system is fielded. The longer systems are in development, the longer warfighters face existing and evolving threats and the higher the likelihood of enemy-developed countermeasures. Realizing the value of interoperability via net-centric capabilities requires a degree of adaptation and continuous evolution not found in today's systems.

Existing systems engineering tools, processes, and technologies poorly support rapid design changes or capability enhancements within acceptable cost and schedule constraints. Their focus on point solutions makes ad-hoc adaptation cumbersome in theatre. To increase development efficiency and ensure flexible solutions in the field, systems engineers need powerful, agile, interoperable, and scalable tools and techniques. As stated by Mr. Lemnios, DoD's goal is to adapt and improve upon commercial marketplace approaches so that warfighters can have "innovative solutions that are transitioned into capabilities in months, not years."

The tools and approaches presented in this study have been selected as the ones best able to meet two Systems 2020 goals. First, they include the solution-acceleration goals of achieving a threefold decrease in development time to develop a fieldable first-article product; a fourfold decrease in change processing time to implement foreseeable classes of fielded systems changes; and the ability to rapidly adapt to unforeseeable threats (for example, improvised explosive devices) or opportunities (for example, the expanding uses of unmanned vehicles). Second, they include the solution-assurance goals of ensuring that the resulting systems are trusted, assured, reliable, and interoperable.

1.2 Integrated Game-Changing Solution Strategy

After studying several candidate research areas (e.g., agile methods, formal methods, requirements tools, test tools), the study initially concluded that three research and technology areas had the best prospects of producing *game-changing* approaches to achieve the Systems 2020 goals. These are:

- Model-Based Engineering (MBE): *Changing the traditional DoD requirements-delay-surprise acquisition game* .MBE applies product, process, property, environment, and mission models to ensure rapid, concurrent, and integrated development of DoD systems that can adapt to foreseeable and unforeseeable change.

- Platform Based Engineering (PBE): *Changing the traditional DoD stovepipe acquisition game.* The complement of MBE for portfolios or product lines, PBE invests in determining DoD-domain commonalities and variabilities, develops product-line architectures that package the commonalities into physical and informational platforms, and provides plug-compatible interfaces to the variable product line components.

- Capability on Demand (COD): *Changing the traditional brittle DoD point-solution acquisition game.* COD provides technology support for evolutionary acquisition strategies that combine short, stabilized build-to-specification increment developments with concurrent change anticipation, analysis, and self-adaptation. This amplifies the effects of MBE and PBE capabilities for rapid new-component generation and integration.

During the course of this study, a fourth research area was identified:

- Trusted Systems Design (TSD): *Changing the traditional slow DoD acquire-certify-patch security assurance game.* TSD includes up-front analysis and systems engineering of foreseeable threat patterns, uses MBE and PBE capabilities to build trust and assurance into DoD system architectures, and ensures that agile change adaptation fully addresses trust and assurance concerns.

A decision tree (Figure 1 in Section 2) illustrates that these four areas fully address the Systems 2020 goals.

Clearly, these technical solutions will need to be complemented by improvements in DoD acquisition policies, practices, regulations, specifications, and standards. Given the complementary USD(AT&L) Ashton Carter "value task force" initiative and resulting planned acquisition guidance, there are major opportunities for Systems 2020 initiatives to be informed by and to inform their acquisition management counterparts.

1.3 Systems 2020 End Results

Table 1 summarizes how DoD systems design and development will be transformed as a result of Systems 2020, along with complementary enabling acquisition practice improvements initiated in other parts of DoD.

Table 1. Systems 2020 Transformation of Systems Design and Development

Goal	Current Practice (with exceptions)	After S2020 (and enabling acquisition improvements)
System Design and Development Speed	Sequential single-step progression from pre-specified system requirements Much subsequent delay, rework from emergent requirements, new changes	Concurrent convergence on baseline-system requirements, design, and feasibility evidence via MBE 3X faster to fieldable first article
System Design and Development Flexibility	Individually designed stovepipe systems with much functionality overlap, incompatible elements and interfaces Serious interoperability problems, expensive change due to system-wide ripple effects of changes	Product lines for families of systems and life-cycle evolution of common physical and informational platforms via PBE 4X faster in response to foreseeable change via modularization around sources of change, confining ripple effects
System Design and Development Adaptability	Short-term focus on over-optimized, brittle point-solution designs with little built-in flexibility Much subsequent expensive rework and change-adaptation delays	Proactive prediction of emerging changes Self-adaptive and rapidly recomposable system components Much faster adaptation to unforeseeable change via COD self-adaptive systems and recomposable components
Trust, Assurance, Reliability, Interoperability	Long delays for security certification; Reactive patching of security holes; High-risk uncertified off-the-shelf (OTS) products Late discovery, slow fixing of side effects of security over-optimization on reliability, interoperability, usability, performance	Much more threat-resilient, trustworthy systems via TSD up-front security-oriented design and tradeoff analysis capabilities Pre-worked avoidance of negative side-effects of security solutions on reliability, interoperability, usability, performance

1.4 Systems 2020 Program Risks and Recommended Next Steps

Section 2.6 of the report identifies a number of potential risks to the success of Systems 2020. These include the risks of insufficient DDR&E-external stakeholder buy-in; the risks of either prematurely or belatedly adopting desired technologies; the risks of failing to identify and address emerging DoD needs and opportunities; and the risks of mismatches between matured, high-payoff Systems 2020 technologies and acquisition-practice disincentives toward their adoption.

The way forward presented in Section 3.1.1 is a flexible, pilot-driven approach that allows immediate start of a few high-potential pilot programs and evolves an overall 2020 plan based on experience, organizational relationships, commonly understood needs and solutions, and openness to changing threats, new requirements, and emergent technologies. It addresses the risks via a set of recommended next steps. These include 2-4 early pilot projects to involve strong but improvement-oriented DDR&E-external stakeholders in the baseline Systems 2020 definition process; identification of the most promising mix of research and technology capabilities via the pilots and direct interaction with DoD research and development organizations as well as technology developers; incrementally evolving the baseline program definition and strategy based on experience and the changing environment; and coordination between developing high-payoff technologies and developing and applying acquisition practices that stimulate their use. Section 3 also presents a set of potential pilot organizations for the currently-identified technology areas.

2.0 Analysis and Findings

2.1 Introduction of Research Areas

2.1.1 Decision Tree Relating Research Areas to Systems 2020 Goals

The decision tree in Figure 1 shows how the four Systems 2020 initiatives (MBE, PBE, COD, and TSD) work together to achieve the Systems 2020 goals of Develop Fast (FAST), Provide Flexibility (FLEX), Provide Adaptability (ADAPT), and Provide Trust and Assurance (TRUST), under the constraints of always having the desired level of Trust and Assurance (TRUST), Reliability (RELIA), and Interoperability (INTEROP).

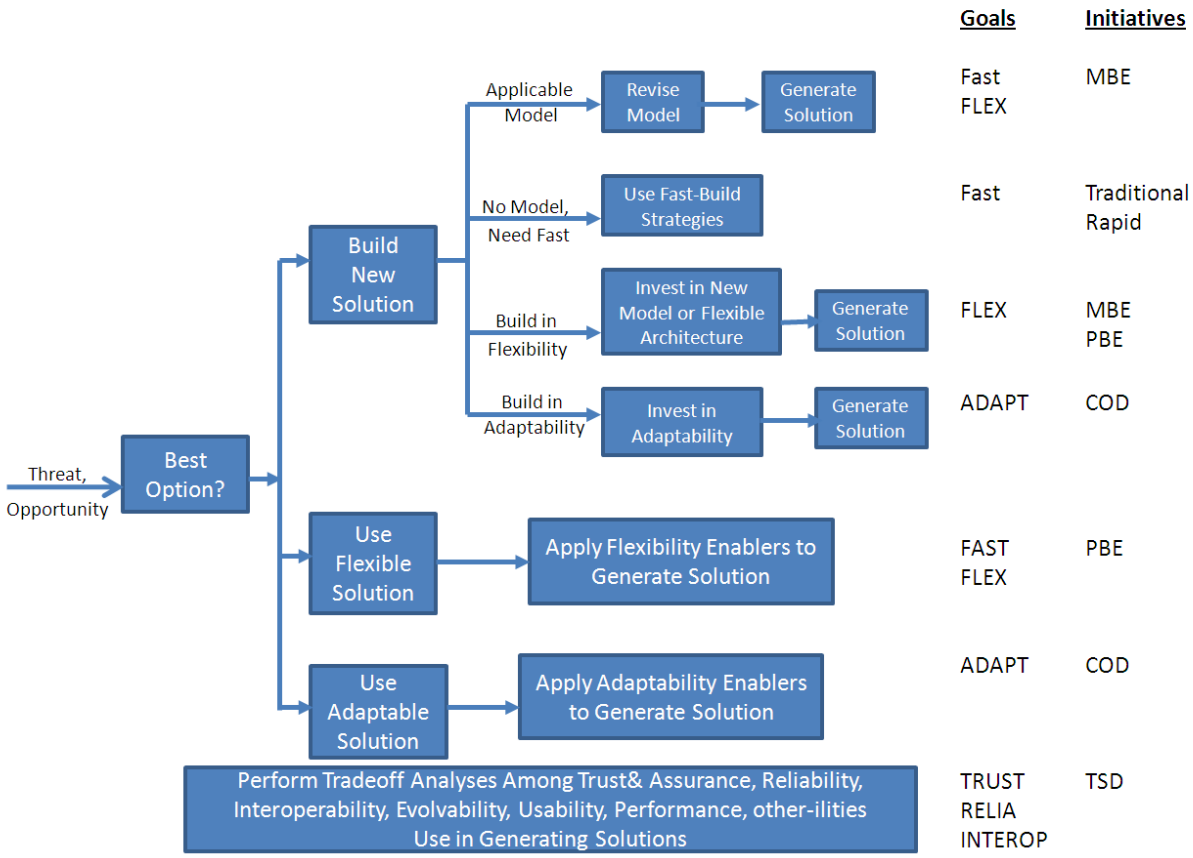

Figure 1. Systems 2020 Goals-to-Initiatives Decision Tree

For any threat or opportunity presented to DoD, there is a need to analyze the available solution options and pick the best cost/schedule-effective option for generating the solution. Starting at the top, it may be that none of the available off-the-shelf modeling, platform, or self-adaptive capabilities can generate a solution, and a new solution needs to be built. Then there are several options that may be most cost/schedule effective. Again, starting at the top, the best solution may be to extend an existing model so that it can generate the solution. The net result of this is that the goals of FAST and FLEX are addressed via the rapid solution generation and the increased flexibility of the MBE capability.

If no such models are available or applicable, and a completely unprecedented, rapidly-fielded solution is needed (e.g., defense against Improvised Explosive Devices (IEDs)), then a rapid application development (RAD) approach using best-available components may be the best option.

Next, based on assessment of the new threats and opportunities, it may be most cost/schedule-effective in the long run to not just build a new point solution, but to invest in making it model-based, platform-based, or inclusive of self-adaptive capabilities. In such cases, one gets a solution to the current problem that takes more resources, but that generates capabilities for future FAST and FLEX outcomes via MBE and/or PBE, or for future ADAPT capabilities for unforeseeable but similar threats or opportunities via COD capabilities. For very large systems or enterprises, different approaches may be applied to different areas, with the result that capabilities from one initiative area (e.g., MBE or PBE) may help in accelerating the solution generation process of another area (e.g., COD).

It may be that a previously developed or enhanced PBE solution is more cost/schedule-effective than building a new solution, in which case the PBE capability realizes the FAST and FLEX goals. Or it may be that using a previously developed or enhanced adaptive COD solution is most cost/schedule-effective, in which case the COD capability realizes the ADAPT goal.

In all cases, it is important not to optimize on speed, flexibility, or adaptability to the extent that the other Systems 2020 goals of Trust and Assurance (TRUST), Reliability (RELIA), and Interoperability (INTEROP) are compromised. This is also an essential element of the fourth S2020 initiative of Trusted Systems Development (TSD)

2.1.2 Effects of Strategy

As DoD begins to realize the "see first; understand first; act first; finish decisively" advantages of net-centric systems, it also begins to see the increase in complexity involved in such systems and systems of systems (OUSD(AT&L) 2008, SEI-CMU 2006). There are many views, definitions, and proxies for complexity (objective, subjective, algorithmic, informational, decision path, lines of software code, etc.). A good summary of complexity views is provided in (Alderson-Doyle 2010). For the purposes of this report, Table 2 provides a pragmatic composite characterization of complexity in terms of the challenges that it presents to DoD systems engineering (Maier 2007).

Table 2. Factors Associated with Complexity

	Simple			*Complex*
Sponsors	One, w/ $	Several, w/ $	One, w/o $	Many, w/o $
Users	Same as sponsors	Aligned with sponsor	Distinct from sponsor	Unknown
Technology	Low	Medium	High	Super-high
Feasibility	Easy	Barely		No
Control	Centralized	Distributed		Virtual
Situation-Objectives	Tame	Discoverable	Ill-structured	Wicked
Quality	Measurable	Semi-measurable		One-shot and unstable
Software Size	10 KSLOC	100 KSLOC	1000 KSLOC	10,000 KSLOC
Program Scope	< $1 Million	$10's of Millions	$100's of Millions	>$ 1 billion
Organizational Maturity	High	Inside low, outside high		First of kind
Technical Scope	Discrete product	Product + Delivery Enterprise	Products or Product-line + Delivery Enterprise	Assemblage of products and enterprises
Operational Adaptation	Stable	User Adaptive	Competitor Adaptive	Full Scope Adaptation

Another definition of complexity that is often useful distinguishes static measures such as parts count, which it defines as "complicated," from an operational measure of complexity as "the degree of difficulty in accurately predicting the behavior of a system over time." (Wade et al 2010a). Often, these are highly correlated, as noted in (DSB 2007) for the growth in thousands of source lines of code (KSLOC) of large commercial software systems such as Red Hat Linux (17,000 KSLOC in 2000; 30,000 KSLOC in 2001) and Windows NT (35,000 KSLOC in 2001; 50,000 KSLOC in 2007), "This increasing size brings with it increasing complexity."

Corresponding data on DoD system complexity vs. year, using KSLOC as a measure of complexity, is shown in Figure 2 (Lucero 2009). Some of the effects of this complexity are shown in Figure 3, also from (Lucero 2009). The red curve on the left shows the number of months of development time as a function of SLOC up to 1250 KSLOC. For Future Combat Systems (FCS), shown in Figure 2 as 17,000 KSLOC, the corresponding formula in (Boehm et al., 2004) is

Development Months = 5 * cube root (KSLOC),

and the corresponding number of development months for FCS is 5 * cube root (17,000) = 129 months, or almost 11 years.

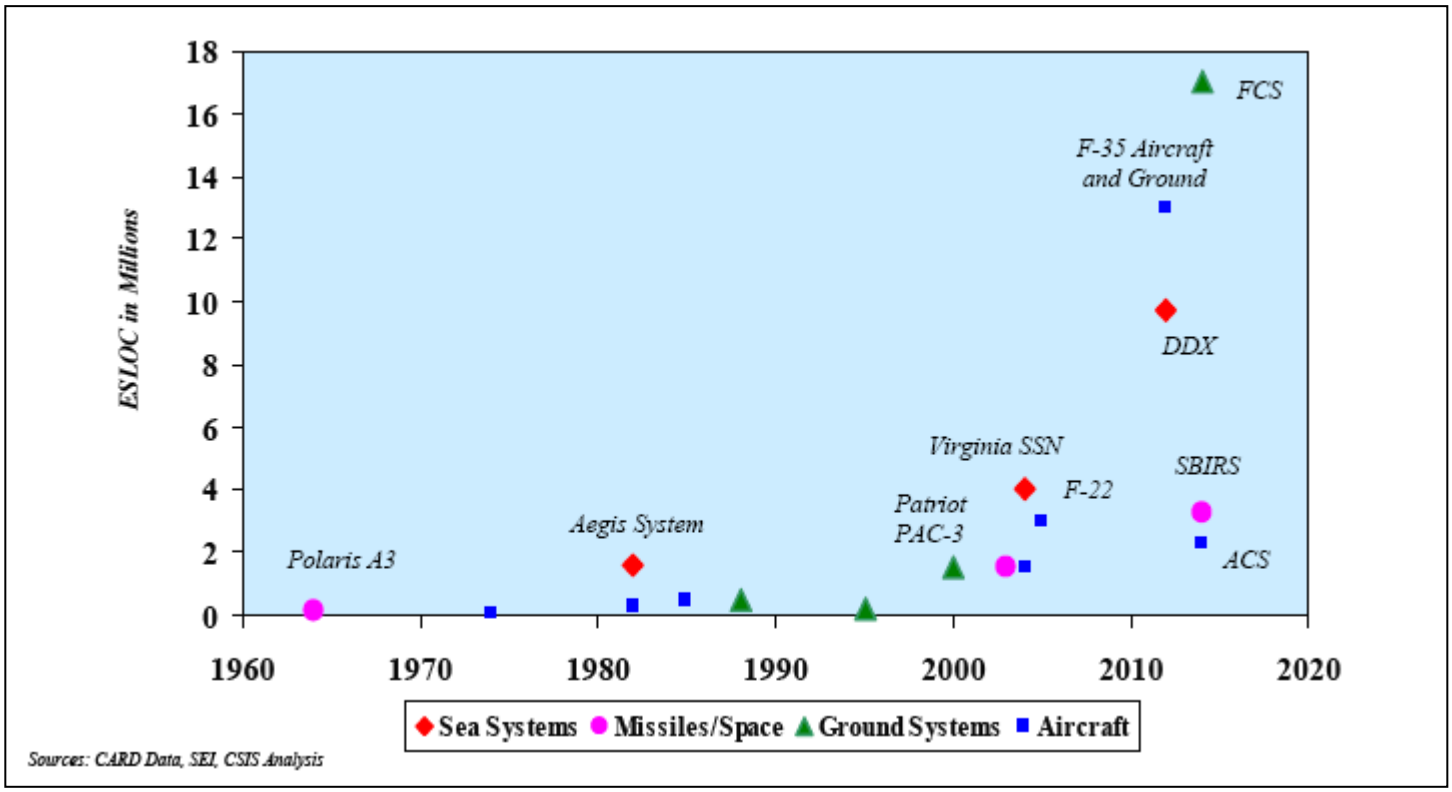

Figure 2. Software Content of Sample Major DoD Weapons Systems 1960-2020

The right hand curve in Figure 3 shows the program cancellation probability as a function of KSLOC. It indicates that the likelihood of cancellation reaches 50% even at 1250 KSLOC, indicating that the cancellation probability is over 50% for programs the size of Future Combat Systems. Currently, more and more programs are going for the benefits of net-centric operations, such as the F-35 with its Autonomic Logistics information System (ALIS), which uses net-centric technology to significantly reduce aircraft turnaround time between sorties. As a result, net-centric systems' software size increases significantly (about 14,000 KSLOC for the F-35, including its ground segment, vs. about 3,000 KSLOC for the F-22 airborne software). Given such trends, it is clear that more rapid and trouble-free methods of systems and software development are needed. For example, the F-35 ALIS' use of an Architected Agile method of development appears to be holding up well. One of the keys to reducing schedule and cancellation probability for very complex systems is to increase investment in systems engineering.

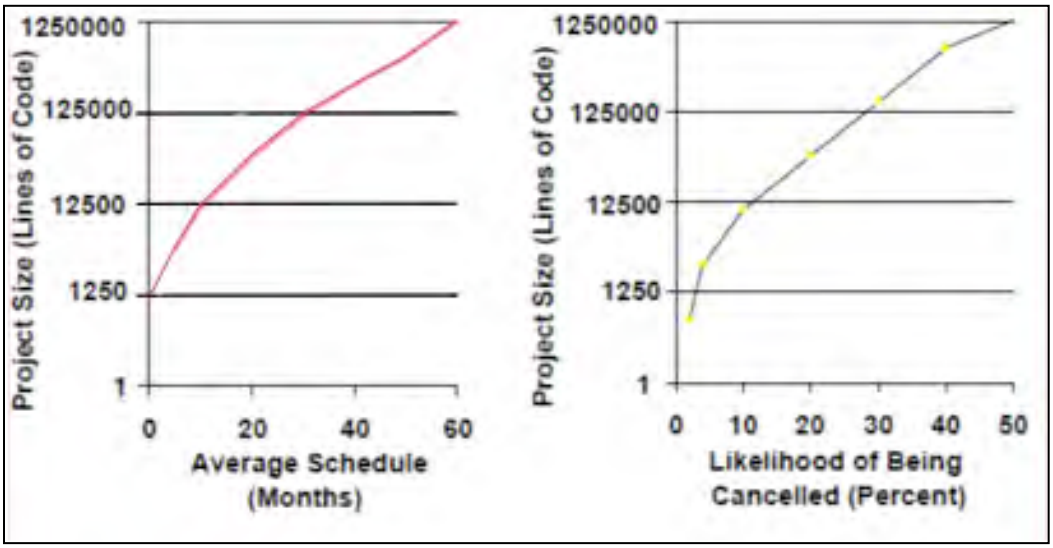

Figure 3. Development Schedule and Cancellation Probability vs Complexity in SLOC

A good example of the effects of software size as a complexity metric on project rework due to shortfalls in systems engineering (SE) is provided in Figure 4. It uses thousands of source lines of code (KSLOC) as a proxy for complexity. For software-intensive systems, quantitative data from 161 DoD-representative projects show that the return on investment (ROI) of SE investments, as measured by the degree that they have fully resolved the project's architecture and risk issues by the project's Preliminary Design Review, is very high for large, critical projects, but relatively low for small, less critical, volatile projects.

The analysis is based on the COCOMO II Architecture and Risk Resolution (RESL) factor. This factor was calibrated along with 22 others to 161 project data points (Boehm et al. 2000). It relates the amount of extra rework effort (the black dashed curves) on a project to the percent of project effort devoted to software-intensive system architecting (the red dotted curve), and to the resulting sum of these costs (the green curves). The analysis indicated that the amount of rework effort (or project delivery delay at full project staffing) 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. As shown in Figure 4, this corresponds to a very high initial ROI for reducing total costs by further investments in large-project SE (the slope of the left end of the highest green curve) which decreases to a diminishing-returns "sweet spot." This analysis was performed for the US Army Future Combat Systems program, and resulted in an additional 18 months being added to the SE schedule. Further details are provided in (Boehm-Valerdi-Honour 2008).

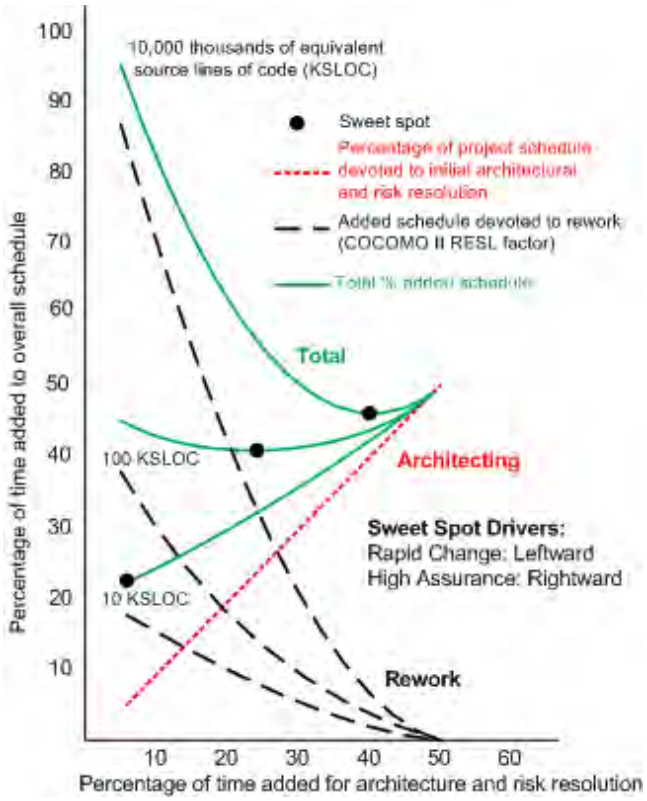

Figure 4. Effect of Size on "How Much Systems Engineering is Enough?"

Figure 5 is adapted from the classic (Blanchard-Fabrycky 1998) Systems Engineering textbook. As illustrated in this figure, major decisions and commitments concerning technologies, materials and potential sources of supplies and equipment, manufacturing processes, and maintenance approaches are often made prematurely in the early stages of a program. These often encumber the program with architectural and contractual commitments that make changes difficult and expensive to accommodate when more system-specific knowledge about preferred solutions becomes available. The bottom line result often plays out as a major overrun, not only within DoD (GAO 2008), but also in commercial projects (Johnson 2006).

This figure serves as a useful roadmap for improving the situation. Strategies such as Model Based Engineering and Platform Based Engineering defer architectural and technology commitments by investing more up-front cost into SE efforts to build up system-specific knowledge. Such knowledge about likely directions of change can be used to determine more change-adaptive architectures (e.g., (Parnas 1979, Baldwin-Clark 2000)) that reduce the steep drop off in ease of change vs. time. Investments in technology, marketplace, and threat trend analysis; autonomous environment monitors; and recommendation engines also build up system-specific knowledge and enable systems to be architected for significantly greater ease of change.

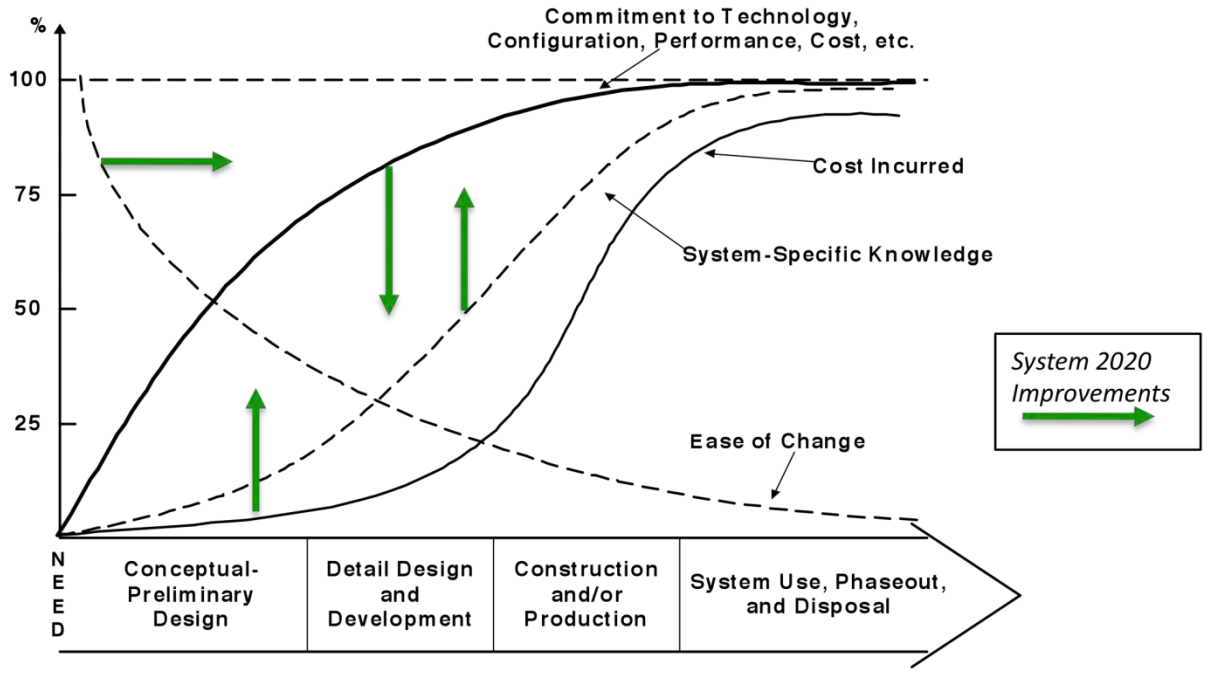

Figure 5. Systems 2020 Systems Engineering Improvements

In response to the primary Systems 2020 objectives of increasing the speed of development, flexibility for foreseeable change, and adaptability to unforeseeable change, Figure 6 provides a Rapid Application Development (RAD) Opportunity Tree for shortening the critical path from inception to fielding. It was inspired by a presentation by Motorola (Pittler 1997) on their corporate "10X" initiative to reduce their critical-path time by a factor of 10. They only achieved factors of 3 to 4, but such factors are consistent with the Systems 2020 goals. Details on the Opportunity Tree strategies are provided in Appendix B.

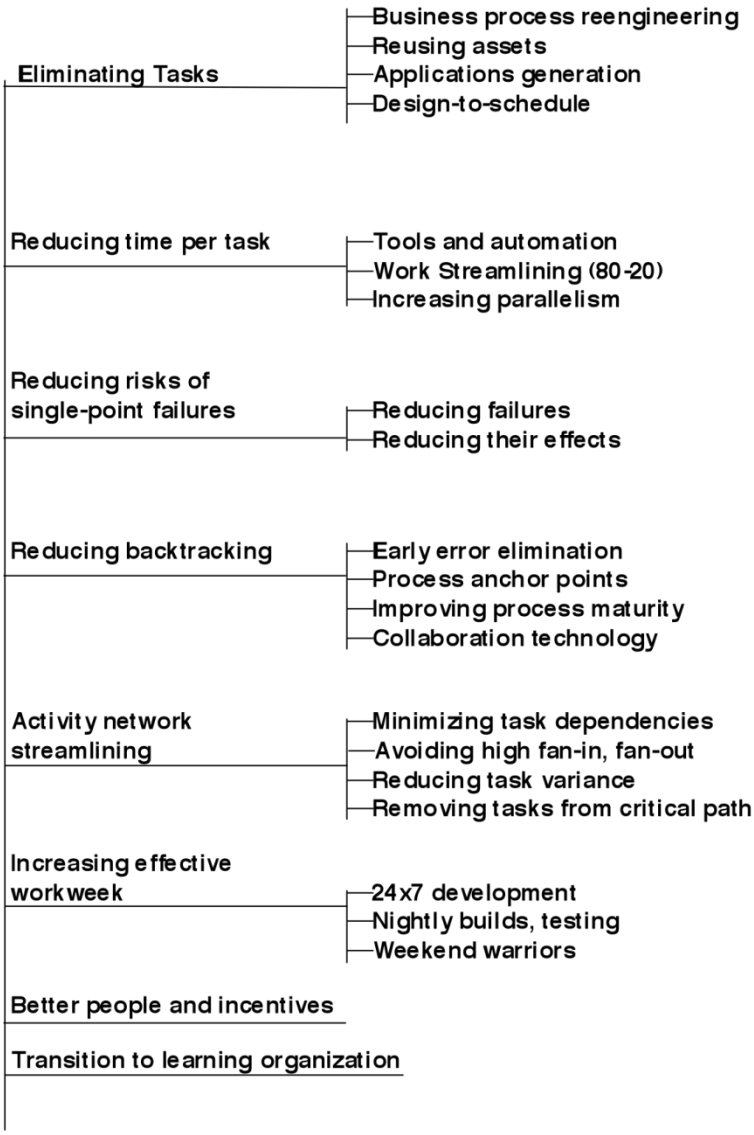

Figure 6. The RAD Opportunity Tree

A representative Platform Based Engineering investment achieving a factor-of-4 decrease in development time is shown in Figure 7. In the late 1980s, Hewlett Packard (HP) found that several of its market sectors had product lifetimes of about 2.75 years, while its waterfall process was taking 4 years for software development. HP's investment in a product line architecture and reusable components increased development time for the first two products in 1986-87, but had reduced development time to one year by 1991-92 (Lim 1998). Similar Platform Based Engineering initiatives in DoD hardware and software domains can have similar payoffs in time to first article development.

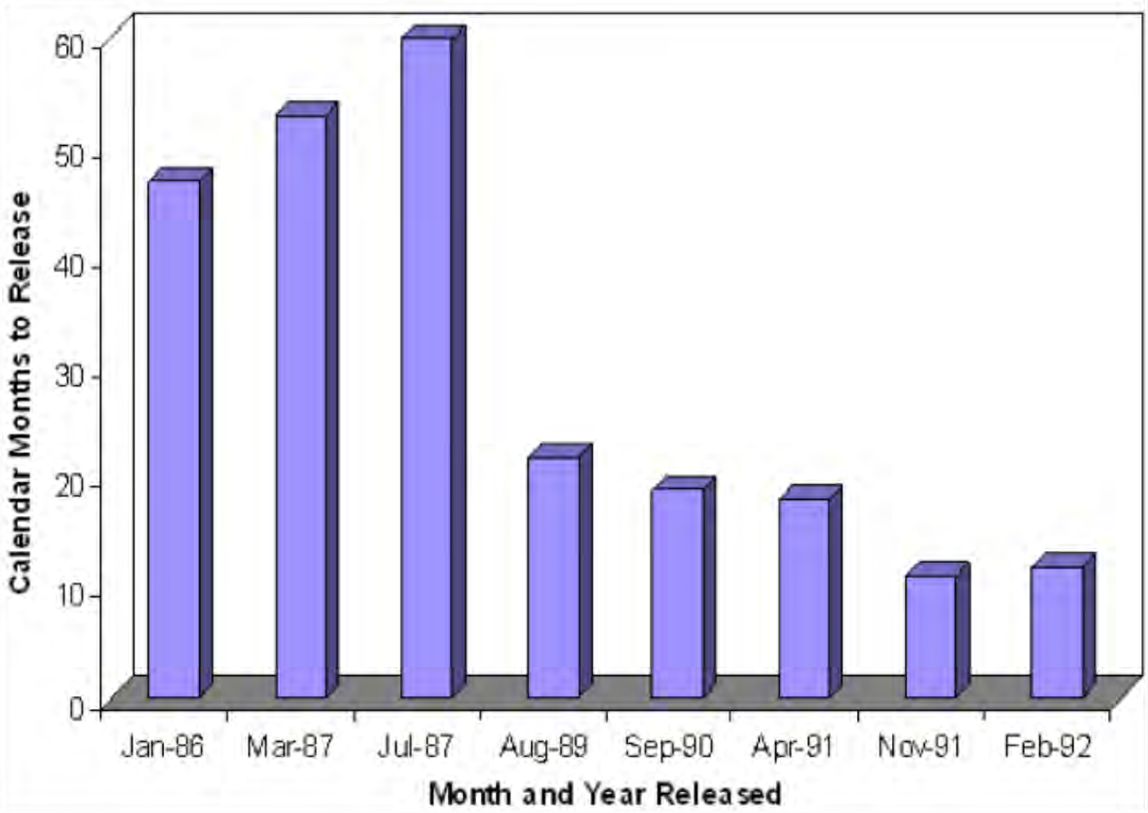

Figure 7. HP Product Line Reuse Investment and Payoff

2.1.3 Model-Based Engineering (MBE)

Traditional DoD system acquisition and development is a linear, sequential, requirements-first process over an extended number of years. Recent DoD initiatives such as DoDI 5000.02 have advocated more concurrently engineered and evolutionarily developed systems, as shown in the top part of Figure 8. However, current acquisition and development support tools (contracting mechanisms, sequential process models, phase-specific support tools, complete-requirements-first review standards) make it difficult for DoD projects to realize the benefits of concurrent engineering of requirements and solutions and of evolutionary acquisition. The length of the process means that committed-to technologies are often stretched to meet distant future performance goals, typically with belated recognition of risks in technology readiness and manufacturing readiness. The linearity of the process involves many handoffs across organizational boundaries, leading inevitably tore-creation of data, miscommunication, errors, and late consideration of production potential and life cycle attributes. Design decisions based on aging concepts of operation and technical assumptions may not remain valid years later. The result is that systems entering production too often encounter problems of low manufacturing yield, high cost, and multiple design changes – and the problems increase as system complexity increases. The opportunity to achieve first pass success – that is, to deliver a system that meets the operational need without scrap, rework and extensive design changes in production – requires shortening the initial development content and timeline; eliminating or mediating the seams in the process; and, as much as possible, deferring change traffic to later evolutionary increments of capability.

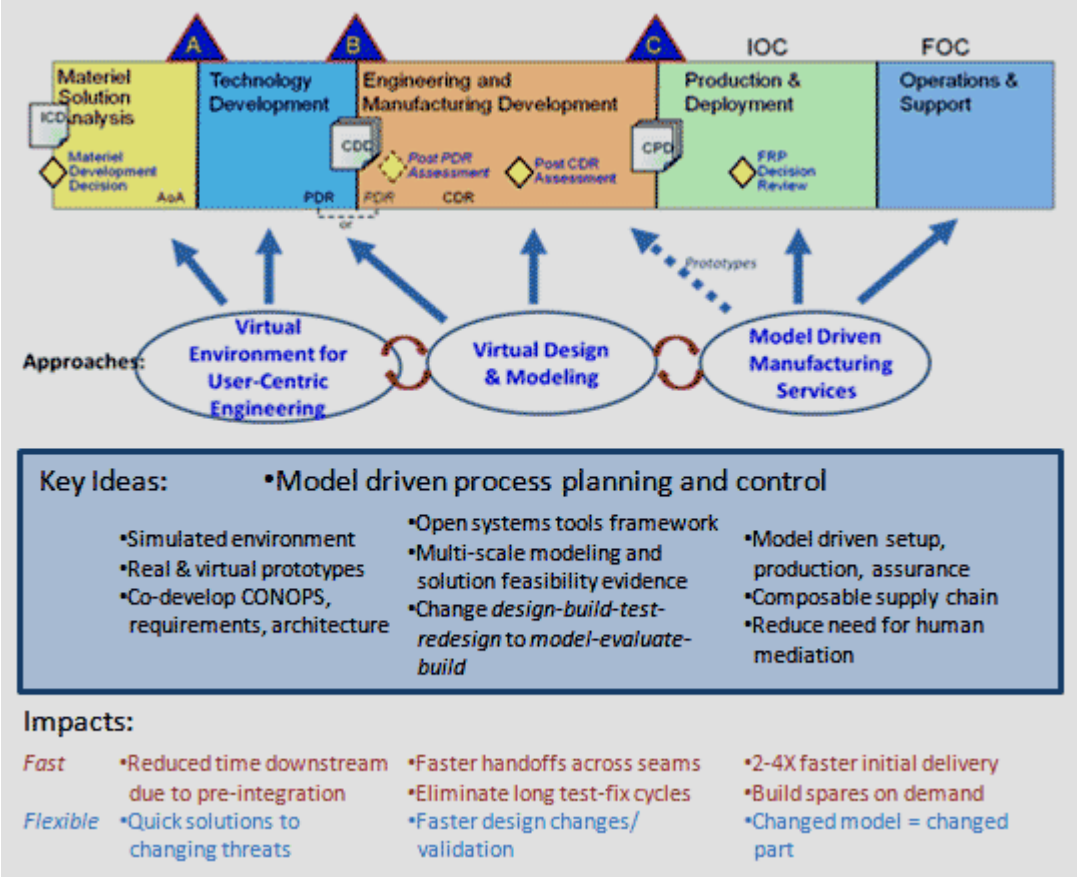

Figure 8. Model Based Engineering Impacts

The MBE solution to address this grand challenge is comprised of four big game-changer ideas: Stakeholder-Centric Concept Engineering, Virtual Design & Modeling, Model Driven Manufacturing, and Complementary Systems Engineering Process, Property, Environment and Mission Models. The lower parts of Figure 8 highlight the role and impact that these ideas would have on traditional system acquisition and development processes. Figure 9 highlights the overall technical approach and the importance of linking all four game changers to create an open-system development environment supporting cross-model change propagation and change impact analysis, and linking CONOPS with produced platforms (manufacturing). The first three game changers are shown to the right of Figure 9; the fourth game changer provides the complementary set of models that determine the appropriate process and context for integrating the first three.

Currently, automobile, integrated circuit, and aircraft developers employ this type of MBE approach to great effect. While commercial MBE endeavors demonstrate the effectiveness of MBE tools and techniques, it is important to note that these solutions are tailored to a particular company and/or product line of systems that are, in most cases, much less complex than typical DoD systems (or systems of systems). The government's investment in networked and high-performance computing provides new opportunities to realize the concept of a model-based virtual system prototype, from CONOPS to manufacturing, through the use of networked and large scale computing resources.

It is also important to emphasize that the models are not just of the physical system, but also of the system's information and human aspects, its environment, the processes of its definition and development, and the tradeoffs among the system's levels-of-service goals such as trust and assurance, reliability, interoperability, and performance. Research challenges include integration of these model areas; integration of models across multiple domains, timescales, and competitive tool vendors; tool trust and assurance; and model verification and validation.

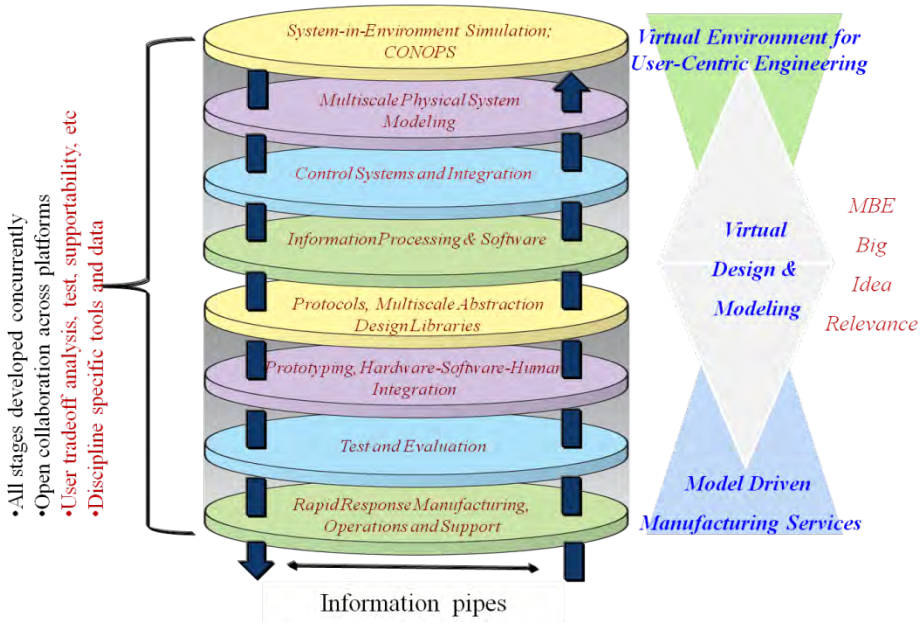

Figure 9. Concurrent Model Development and Integration Framework

2.1.4 Platform Based Engineering (PBE)

Platform Based Engineering (PBE) is a cost-effective, risk-mitigated system design and development approach that employs a common structure from which high-quality derivative products can be developed rapidly. When appropriately scoped, PBE is effective in decreasing development cost and lead times while increasing product quality (see Figure 7). PBE is distinguished by two unique characteristics. First, the platforms are *reusable, configurable, and extensible* system implementation infrastructures (platforms). These can include *physical platforms* such as aircraft and spacecraft configurations designed to support a range of payloads or missions, as well as *information platforms* that provide a layered set of services for network management, operating system, data management, and mission support. These platforms are leveraged to simplify and accelerate the development of families of systems or product lines for a particular problem domain. Second, the platforms encompass *domain-specific components and services* that reflect the *commonalities* of systems in the domain (that can be configured as reusable physical or informational components), and *variabilities* across the domain (that need to be individually developed to achieve a domain product line), along with interface conventions that ensure that they can plug-and-play with the domain infrastructure and common components.

2.1.5 Capability on Demand (COD)

In the context of Systems 2020, Capability on Demand (COD) addresses the challenge of rapidly adapting fielded systems to unforeseeable new threats and opportunities. Keys to rapid adaptability include capabilities to rapidly detect and analyze new threat and opportunity trends and patterns, and capabilities to rapidly reconfigure the system to counter the threat or to capitalize on the opportunity.

Some significant advances have been made in the areas of purely autonomous threat and opportunity analysis and reconfiguration capabilities, such as the use of agent-based systems and incentive-driven complex adaptive systems to mediate multidimensional network qualities of service. However, such autonomous capabilities have challenges in terms of verifying and validating multi-agent collaboration and self-modifying-system failure diagnosis (consider stock market trading as an example). On the other hand, purely human adaptability has complementary shortfalls in processing masses of data and human error modes.

This leads to classes of rapid human-computer collaboration to draw upon needed and available resources that are both "hard" (autonomy and infrastructure based) and "soft" (human, organizational, process, and collaborative). These can use the human and computer strengths to diagnose unforeseeable threats and opportunities, and to articulate, execute, and monitor solutions. Other keys to success involve rapid-MBE component generation, PBE component adaptation, the use of service-oriented components that can be rapidly reconfigured, and the provision of user-programmable capabilities to empower warfighters with reconfiguration capabilities.

A particularly high-leverage opportunity area in this regard involves composable DoD components. For example, embedded sensor and computing components that can be composed to generate, filter, and analyze data to meet emerging needs for information, along with DoD counterparts to commercial capabilities that support creating improvised combinations of application-library components.

2.1.6 Trusted System Design (TSD)

Trustworthy systems are those accompanied by evidence that they will perform satisfactorily. TSD is a discipline composed of methods and support tools for architecture, design, and design-driven implementation of trustworthy systems from untrustworthy components or subsystems, or for assurance of the trustworthiness of systems of unknown provenance. TSD requires trust assurance criteria and associated measurement techniques for trust verification. TSD models and metrics allow systems to operate in the context of a current and accurate attribution of trust. Trust criteria are applied to the system of interest or to its communication or interaction with other systems that employ the same or similar techniques. The techniques should be applicable to a wide variety of system models and architectures while avoiding pitfalls due to composability and transitivity.

Trusted system design encompasses hardware, software, and operational trust verification. Methods, tools, and procedures support criticality analysis of system components, identification of critical program information identification, security against persistent and adaptive threats, countermeasures for counterfeiting, supply chain design and validation, forensic investigation, and damage assessment. Hardware-specific TSD includes solutions for FPGA reverse engineering, security of data at rest, and

component integrity verification, traceability, and testing. Software-specific TSD includes solutions for software design assurance practices, simplex-enabled configuration monitoring, models for mutable threat surfaces, and malware quarantine and reporting.

A key to trusted system design is the portion of Model Based Engineering that supports the analysis of tradeoffs between trust and assurance and other system objectives such as performance (avoiding too much security system overhead), reliability (avoiding single points of failure), usability (proliferation of keys and passwords), and interoperability (over-constraining information sharing).

2.2 Model Based Engineering (MBE) Analysis

2.2.1 Introduction

The Systems 2020 challenges are to achieve a threefold decrease in time to develop a fieldable first-article product, a fourfold time reduction to implement foreseeable classes of systems changes, and the ability to maintain a tactical edge when presented with unforeseeable threats or opportunities. The first two of these goals are fundamentally challenges in latency reduction, while reducing cost and improving quality. The reduction of latency can be achieved by the following as evidenced by the success at Motorola (Pittler 1997):

1) as much in parallel as possible while keeping all available resources optimally loaded ,
2) by reducing the amount of work that needs to be done and increasing productivity, and
3) avoiding rework, particularly late in the life-cycle.

These goals are addressed by the four research areas under MBE:

- Virtual environment for stakeholder-centric concept engineering
- Virtual design and modeling
- Model-driven manufacturing services
- Complementary Systems Engineering Process, Property, Environment and Mission Models

2.2.2 Virtual Environment for Stakeholder-Centric Concept Engineering (VESCCE)

2.2.2.1 Opportunity

Concept Engineering (Cloutier et al. 2010, Carlini 2009) requires the creation of an infrastructure and processes necessary to provide an efficient interactive environment where multiple stakeholders can collectively develop a concept of operations model that can be used throughout the lifecycle. The approach should include a broad range of tools to enable conceptualization at multiple levels of refinement, ranging from brainstorming to creating and evaluating high-level system behavioral models. Moreover, it should be done in a manner that facilitates input from a set of diverse stakeholders and limits the cognitive demands placed on them. Done correctly, the models will also be of value in supporting continuous validation throughout the lifecycle. In addition, they can be used to facilitate operator training and decrease the time from deployment to system effectiveness. A graphical

environment is needed to quickly communicate a concept of operations, conceptual and behavioral models for new missions, business processes, and feature sets to realize a shared mental model and understanding of the mission, and potential solutions across a set of diverse stakeholders. This graphical environment has the potential to become an agile mechanism for eliciting stakeholder expectations. It is likely that this environment will have to be tailored for selected application domains, along with cross-domain bridging capabilities for operational, mission, and semantic consistency.

2.2.2.2 Approach

The vision for the Virtual Environment for Stakeholder Centric Concept Engineering (VESCCE)is an interactive, collaborative, multimedia environment for multiple stakeholders to quickly construct CONOPS and other high-level abstract models of the system under development. The environment supports creation, validation, prioritization, and resolution of conflicting requirements; managing changing and emerging requirements; tradeoff analyses; and, creation of a mutually understandable description of the desired system concept. A number of conceptual views of the systems, with increasing levels of fidelity, guide the user towards better solutions with respect to time, effort, cost and risk. Rapid conceptual prototyping uses both real and virtual prototypes in both real and simulated environments. Bridging the communication gap between the DoD requirements and engineering organizations is done through visual models and/or simulations. It is challenging work as the technology projections are fuzzy, the future operational requirements may be questionable, and the high-level concepts do not contain a great detail of engineering analysis. Handoff of requirements to drive detailed design takes place through the iterative transfer of models to the engineering environment, rather than ambiguous text-based documents. Providing this shared and validated model of the system may be the most effective means by which to generate derived system requirements. Rather than creating a complex set of requirements that together defines what the system should do, this approach provides the means to define and validate the operator expectation for the system and its performance, and then generate an evolving set of resultant requirements.

Attributes of the envisioned environment (Wade et al. 2010b) include the following:

Low-Overhead Communication: VESCCE should provide low-overhead communication between all of the stakeholders who interact in the creation of the conceptual model of the system. The interactive capabilities provide instant feedback to the users and between the users. This feedback conveys information in the most effective way by tailoring communication to the needs of each of the users.

Architectural Design Support: Effective architectural design is dependent on an intimate understanding and vision for the conceptual operation, behavior and desired characteristics of the system. VESCCE provides an executable model that can be used interactively by the architects and stakeholders in providing this shared vision and understanding of the system.

Risk/Opportunity Management: Much of the leverage in a system lifecycle happens during the conceptual phase. Mistakes in this phase are often irrecoverable or amended only at great cost later in the system lifecycle. VESCCE provides the means to validate the value proposition of the system early,

thus reducing the risk of discovering problems downstream and providing a rapid prototyping conceptual environment for opportunity discovery at the onset.

Verification & Validation: Understanding the conceptual operation of a system early is critical for both verification and validation. The architectural and/or behavioral model should be incrementally developed and interfaced with VESCCE to validate the solution. VESCCE then provides the capability for continuous validation of the system concept, while providing the fundamental understanding that is essential for effective verification throughout the lifecycle.

Legacy Integration: VESCCE provides the capability to allow users of legacy systems to validate new systems concepts. This provides a bridge for end users between the system they currently have and the systems they will have in the future. VESCCE makes use of a library of composable elements, and thus can take advantage of legacy models for future developments, ensuring their consistency. High-level models of the current system can be integrated into VESCCE and provide validation at the desired level of granularity. If they do not exist, VESCCE will allow for the creation of "boundary components" to enable simulated interfaces to the legacy systems.

Human Aware/Self-Adaptive: VESCCE provides an interface that is tailored to the specific user of the system, using the lexicon of the user instead of "engineering speak." Thus, the user's interactions provide a means by which to gauge if the system will conceptually provide humans with the desired behavior. The development of the VESCCE models is a human-based activity that is meant to capture the appropriate elements of human behavior.

Complexity Handling Capabilities: VESCCE is, by its nature, a conceptual modeling system that enables the abstraction of complex systems behavior to the appropriate level under consideration. As the VESCCE models are refined, increasing levels of fidelity may be added while interfacing to existing systems throughout the lifecycle. Thus, these models can be used to provide information at the desired level of detail, while providing the conceptual view of how the system is actually supposed to work. This connection is essential to handling complex systems.

Cycle Time Reduction: By providing an efficient environment for the construction of conceptual models, VESCCE can both reduce the amount of time spent on these activities and, more importantly, greatly increase its effectiveness. This upfront work will dramatically reduce the efforts downstream by focusing those efforts on the necessary, value-creating work. In addition, the ability to do continuous validation reduces the risk of rework that can have a huge impact on schedule and quality of the delivered system.

2.2.2.3 Impact

The benefits of VESCCE are to improve the ability to collaboratively and interactively create a model of the desired system behavior that can be used throughout the system life cycle resulting in:

- Accelerated **time to fielding** through:
  - ability to rapidly evaluate alternative concepts
  - increased capabilities in effective, rapid distributed decision making

- improved communication with all stakeholders of the value proposition and intended operation of the system
- avoidance of lengthy redesign cycles associated with poorly defined requirements
- automated or semi-automated generation of associated artifacts such as test cases, simulator-stimulators, prototypes, or system components

- Increased **quality** through improved ability to:
  - analyze more alternatives and choose the most satisfactory initial system definition upfront
  - support continuous system verification and validation throughout the lifecycle
  - train deployment, service, and support personnel earlier in lifecycle

- Increased **flexibility** through ability to:
  - rapidly evaluate changing threats and explore the solution space
  - develop CONOPS to use existing/modified systems in creative new ways
  - modify component and system definitions at the model level and generate new capabilities

2.2.2.4 State of Practice and State of the Art

The weakest link in systems engineering is often the link between what the warfighters need and what the development team *thinks* they need, together with a shared understanding of the operational environment and associated constraints and dependencies. While conceptualization seems less formal than engineering, a disciplined approach is necessary to ensure that the stated needs of the customer are transformed into a product or service that meets the actual customer and mission needs. Currently, the conceptualization phase of a project is either an ad hoc event with pens and napkins resulting in presentation slides, or it is a sterile, and laborious document-driven process in which a large, unwieldy text-based document is created – destined to become shelf-ware (Cloutier 2009). In either case, system developers and users alike often have inconsistent understandings of what the system is actually supposed to do and how it creates value. State-of-the-art tools and processes are emerging that can provide an efficient interactive environment so that multiple stakeholders can create a shared mental model during the brainstorming process through the development of a concept of operations model which can be used throughout the lifecycle. One example is the DARPA Real World toolkit for rapid user-developed simulations. Another example can be found in efforts by EADS to develop a synthetic environment to design simulate evaluate and demonstrate Network Centric Operations concepts and systems.

2.2.2.5 Gaps

Additional tool development is needed for the following areas:

- A more effective cognitive concept development environment
- An environment for rapid evaluation of alternative concepts
- A vehicle to validate the concept throughout the development lifecycle
- A vehicle to perform trade analysis for upgrade options in subsequent versions of products

- Scalable capabilities for complex systems of systems.

Research in this area will apply tools and processes necessary to provide an efficient interactive environment so that multiple stakeholders can create a shared mental model during the brainstorming process through the development of a concept of operations. The full set of tools would include graphical interface tools to allow the interactive, collaborative development of models representing a concept of operations and system behavior, that can be simulated and tested with a portfolio of canned scenarios, with the resultant behavior being viewable from a variety of perspectives that are tailored to each user's needs.

In order to prioritize capabilities, it would be convenient if all the system's success critical stakeholders had readily expressible and compatible value propositions. *Readily expressible* is often unachievable because the specifics of stakeholders' value propositions tend to be emergent through experience rather than obtainable through surveys. In such cases, synthetic-experience techniques such as prototypes, scenarios, and stories can accelerate elicitation of priorities. Readily compatible stakeholder value propositions can be achievable in situations of long-term stakeholder mutual understanding and trust.

2.2.3 Virtual Design and Modeling

2.2.3.1 Opportunity

Virtual Design and Modeling has its roots in the modeling and simulation field. According to some experts in systems engineering, in recent years these two areas have been diverging, creating two distinct perspectives. Therefore, a common definition of these two modalities was developed in order to encourage convergence to one understanding. The development of models and simulations, mathematical, physics-based, or simply abstract representations for systems, are excellent for both predictive behavior and trade-space analysis by designers. Developers have become highly dependent on modeling systems under development, making systems modeling an integral part of systems engineering. As systems increase in complexity and technologies advance rapidly, modeling itself and the interactions of models becomes more important. The four areas of research interest under Virtual Design and Modeling are 1) multi-scale modeling and process simulation, 2) hardware and software multi-scale library data bases, 3) test driven analysis, validation and trust, and 4) open systems engineering environment. By making these innovative ideas the core of the Virtual Design and Modeling effort, real-time assessment of system changes, enabling robust evaluation of different approaches, facilitating fast development of products and processes, as well as design flexibility are all now possible. The result is a major paradigm shift from a linear sequential development process to an evidence-driven concurrent process. This continuous involvement and visualization capability enables users to gain increasing confidence of the process and the final product. Even greater benefits are achieved when Virtual Design and Modeling is coupled with VESCCE on the front end and Model Driven Manufacturing on the back end, thus creating a complete environment from CONOPS to manufacturing.

2.2.3.2 Approach

The approach is to develop an open systems engineering environment that allows the system to be built virtually before it is built physically, with full community participation and early consideration of downstream issues. The goal is to create an iterative collaborative environment with a rich suite of discipline tailored tools, extensive use of multi-scale physical and virtual prototypes, model-based design and test of the system and its manufacturing processes, and an integrated living MBE database. The multi-scale temporal models that populate this framework over the system life cycle will enable analysis of most system trades, from CONOPS to design to manufacturing, and greatly reduce the cost and time for future upgrades. Such an environment will enable acquisition managers, system engineers, design specialists and manufacturing engineers to get the product and process design pointed in the right direction, and able to accommodate changes quickly.

2.2.3.3 Impact

In regards to the System 2020 objectives of fast, flexible and adaptive, MBE is most relevant in reducing the time to successful acquisition and fielding of systems (fast), and also in enabling flexibility and adaptability where changes can be accommodated via model-based synthesis. A number of existence proofs already exist such as the "Rapid Synthesis of HLA-Based Heterogeneous Simulation: A Model Based Integration Approach" by Vanderbilt University, RDECOM MRAP Expedient Armor Program (MEAP), and Boeing commercial aircraft development. The RDECOM adaptive example is based on a new theatre threat that resulted in a requirement that the MRAP get added protection. Modeling and simulation (M&S) was used to assess possible alternatives in regards to survivability, mobility, and dynamic ride. This was achieved in two weeks versus many months if M&S was not used. An example at the subsystem level that provides additional evidence is that Boeing Commercial Aircraft has reduced wing design time from 18 months to less than 6 months using comprehensive Computational Fluid Design (CFD) M&S tools. A large portion of the time reduction was due to reducing the number of wind tunnel tests required. As another example of improvements possible with virtual design, consider automotive systems. Twenty years ago, the time to manufacture a vehicle after launching initial design concept was 50 months. It is now around 20 months, even though vehicle complexity has grown by 2-3 orders of magnitude. In the absence of virtualization, the process would take much longer, be able to consider many fewer alternatives, and be much slower in responding to change.

2.2.3.4 State of Practice and State of the Art

Much of today's development relies on stove-piped, specification-driven requirements to prototype to production and sustainment processes. The linearity of this process involves many hand offs across organizational/discipline seams, inevitably with re-creation of data, miscommunication and errors, and late consideration of producibility and life cycle attributes, coupled with the inability of lower level design decisions and analyses to influence higher level designs and architectures. Existing state of the practice tools are in a category called Product Life-cycle Management (PLM) and within that there are four major categories: Mechanical CAE/CAD/CAM (MCAD), Electrical CAE/CAD/CAM (ECAD), Software design, development and management (formerly called CASE tools) and Engineering Data Management (formerly called EDM tools). Dassault Systems, SIEMENS, and Parametric Technology Corporation provide families of mechanical design and analysis tools (CATIA, PLM software products, CoCreate

family) through the MCAD market (automobile, aerospace, shipbuilding), , Cadence Design Systems, and Mentor Graphics address electrical and electronic design, analysis and manufacturing with ECAD design and analysis tools (electronics market), while Rational and Wind River address software design, development and management of enterprise scale and embedded systems. Telelogic and Rational (acquired by IBM) provide enterprise architecture and systems engineering, and modeling technologies, but these tools do not provide sufficiently robust capabilities for effective system level design trades. Each of the tool vendors provide their own data and process management tools using proprietary data formats, thus inhibiting the fluid flow of information across the life cycle and different disciplines. The majority of the basic model-based tools for simulation of a single discipline or domain already exist. Multi-domain modeling simulation tools, libraries and processes are just starting to emerge in research. Today's tools support partial traceability of requirements but do not enable designs to be fully evaluated and iterated, to support accurate change impact analyses across large systems, or to enable system models to be run against performance envelopes prior to procurement. All of the major aerospace and automobile companies have proprietary capabilities for creating advanced virtual systems.

2.2.3.5 Gaps

The goal is to develop an open systems engineering environment that allows a system to be built virtually before it is built physically, with full community participation and early consideration of downstream issues. This iterative collaborative environment requires a rich suite of discipline-tailored tools, multi-scale models and libraries with extensive use of multi-scale physical and virtual prototypes, and implementation of a system model-based design process. Today's tools offer uneven capabilities, do not follow service-oriented composability conventions, and some critical capabilities are completely missing. Tools must be integrated, scaled, and enhanced in order to support a new MBE design paradigm. In order for tool vendors to see the business value of investing in such tools, and industry MBE-related trade and standard organizations to endorse this approach, successful early pilots are needed to mature the MBE process. Ultimately in order to be successful, model-based evidence of solution feasibility, including trust, must be treated as a first class citizen throughout the MBE process.

2.2.4 Model-Driven Manufacturing Services

2.2.4.1 Opportunity

The overall MBE shift from *build-test-redesign* to *model-evaluate-build* creates an opportunity to reduce time and increase flexibility in the physical product realization process. In building a physical prototype or production item, time on tool is a small fraction of the total time from order to receipt. Most of the manufacturing time is consumed in human-mediated activities of sourcing materials and parts, process planning and scheduling, shop floor integration, supply chain integration, queuing time, setup times, and final assembly and quality control. The approach of Model Driven Manufacturing Services targets these latter components of manufacturing time by capitalizing on two classes of manufacturing enablers. The first is model-driven process planning and control of physical manufacturing resources (machine tools, assembly processes, etc.) to reduce human delays and process errors. The second is service-oriented manufacturing, where manufacturing resources and transaction processes are virtualized and exposed as services that are composable across the networked manufacturing enterprise. Building on Service

Oriented Architecture (SOA) concepts from the information technology (IT) world, trusted manufacturing services can be linked through standard interfaces and managed with service level agreements across a supply chain. The advantages for DoD prime contractors are to reduce the time required for sourcing parts, integrating shop floor and supply chain operations, and eliminating bottlenecks. There are also advantages in the flexibility to accommodate changes in product design or manufacturing capabilities. Advances in the flexibility of physical manufacturing tools (such as 3-D printing), and advances in supply chain management (such as agent based scheduling) are viewed as improved services that can be substituted for less efficient services as they become available.

2.2.4.2 Approach

Model driven manufacturing process planning and control starts during the design phase, with design rules, design for manufacturing (DFM) advisors, process visualizations, and physics based simulations that ensure the design is able to be manufactured and that new manufacturing processes are executable. This vision is often called Virtual Manufacturing (NRC 2004). The resulting product and process models can drive not only simulation models but also machine tool controllers. For unit processes (single manufacturing steps) this is straightforward, but for more complex series of steps, process planning is needed. Both machine tool characteristics and human factors aspects of fabrication and assembly are part of this process planning. A goal in this area is auto-generation of process plans to produce the desired product using a given set of manufacturing resources. Validation is accomplished by building the product in simulation before releasing the design for physical fabrication. The resulting process plans and product data should be in a form that can be compiled into machine tool and shop floor control instructions. An analogy from desktop publishing might be the "print preview" function that validates the format on a virtual printer followed by the "print" function that drives a physical printing device.

Service-Oriented Manufacturing encompasses both the architecture and business practices of future manufacturing enterprises, extending from global networks to the factory floor. A key idea in this concept is the virtualization of manufacturing processes and resources so that they become accessible via a Service-Oriented Architecture (SOA) from anywhere within the manufacturing enterprise. SOAs have been developed to decouple business services from the underlying *plumbing* of information technology. Rather than large, highly integrated software applications that are hard to change, SOA systems compose end-to-end solutions by linking modules called services. The infrastructure to support the linking of modules includes the following:

- Registries to make available services discoverable
- Standards to make services interface properly
- Service level agreements (contracts) to ensure performance
- Design patterns (templates) to promote interoperability and reuse

These concepts have counterparts in the manufacturing domain today, but they lack the formalisms of SOA that allow linkage of modules at a high level of abstraction and with a high degree of automation. In particular, people are needed to mediate manufacturing services today much more than in IT. Manual effort adds time, complexity, and transaction costs to the manufacturing process, but is

necessary to work across “seams” between design and manufacturing, and among elements of the supply chain. Human designers must mediate the complex interactions between functional design requirements, parts design and selection, and manufacturing processes. Purchasing agents must discover what components and manufacturing services are available, and at what price, in a constantly changing market. Human production planners must orchestrate a dynamic process to reserve the right resources and get the right sequence of manufacturing and assembly steps to come together at the right points in time and space. Human quality control planners must ensure a service meets specifications and is delivered on time. And, unlike IT, the transaction costs and setup charges become a dominant cost element in small-lot manufacturing. The Service-Oriented Manufacturing vision is to establish a SOA-based infrastructure and design patterns that will put manufacturing industries on par with information services in terms of rapid innovation and agility.

2.2.4.3 Impact

The Model-Driven Manufacturing Services approach offers high impact on the Systems-2020 objectives of fast and flexible, starting during design and test phases, and increasing during production and life cycle support.

Model Driven Process Planning and Control: The primary impact during the design phase is the development of designs that can be manufactured and validated process plans that avoid redundant effort and redesign cycles downstream. There is long experience in the microelectronics domain with design rules (Conway-Mead 1979) that decouple chip design from manufacturing, and with semiconductor process simulation for new generations of semiconductor technologies. The result has been to reduce manufacturing development time from multiple years to a predictable 18-month cycle for each technology node, until recently making Moore's law possible (Sangiovanni-Vincentelli 2009). Developing similar capabilities for 3-D structural and electro-mechanical system design is expected to yield similar payoffs. The Intelligent Manufacturing Technology Initiative (IMTI) reports "a ten-fold reduction in the response time from requirements to complete manufacturing plan and a factor five times reduction in the cost and time of managing change are being documented” (IMTI 2010) by companies using early versions of manufacturing process simulation. Beyond initial delivery, the ability to build a part on demand for sustainment or to change a product configuration by changing the models that drive manufacturing will contribute to the Systems-2020 flexibility objective.

Service Oriented Manufacturing: For defense systems, typically 70% of the manufacturing cost and time is associated with items from lower tier suppliers. Making product and process data available as a service can shorten the time for sub-tier suppliers to respond to both initial orders and changes. For example, the DoD ManTech program demonstrated a substantial increase in competition and decrease in response time by providing process data in addition to product data. The ManTech demonstration on a M2 machine gun part showed that an enhanced digital technical data package enabled suppliers to reduce the time required to interpret data and generate process plans, resulting in multiple competitive bids where there had previously been no bidders, and reducing time to delivery by 59%.Moving beyond data as a service at the DoD and prime contractor level to a full set of manufacturing services can have even larger benefits. The Electronics Services Manufacturing sector has a number of commercial firms

that provide design, manufacturing, and order fulfillment services. In 2005 DoD used such a service-oriented firm to produce an IED jammer called Warlock Blue. The result was that 8,000 units were delivered in 54 days, compared to an estimate of 240 days from a traditional defense supplier. A change to the Warlock Blue order requiring addition of an antenna was handled with no delay in delivery, showing that service orientation offers flexibility as well as speed.

Based on these examples, an impact of 2-4 times reduction in initial production time is a reasonable goal for Model-Driven Manufacturing Services. An outcome that a change in a model usually results in a near-instantaneous change in the manufacturing of a part is the ultimate goal for flexibility.

2.2.4.4 State of Practice and State of the Art

Model driven process planning and control: The current state of practice is mature in the semiconductor domain, and embryonic in most other domains. Physics based process models are within the state of the art for selected processes, such as welding or metal casting, but their use in practice is ad hoc and spotty. Product Data Management systems handle configuration data, bills of materials and CAD data (product models), but generally do not include manufacturing process data. Queuing models and simulations of process flows on the shop floor are widely available, but are generally not integrated with shop floor control systems. Visualization of manufacturing processes and associated ergonomics has been demonstrated but is not widely applied. Auto-generation of process plans from product models has been demonstrated in well-bounded environments, but the state of practice is predominantly manual process planning.

Service-Oriented Manufacturing (SOM): SOA and cloud computing have made substantial inroads in the IT sector, but not yet in the manufacturing sector of the economy. Commercial manufacturing services are available on the web, but are human mediated and lack a SOA infrastructure. Offshore supplies offer many available manufacturing services, and there is no mechanism for certifying trusted services. The state of the art of SOM reflected in peer-reviewed journals is dominated by research from Asia, and to a lesser extent, from Europe. The U.S. experience with manufacturing services is most mature in the electronics sector. In microelectronics, early success in DARPA's MOSIS program paved the way for "fab-less" design companies who today have highly automated connections with semiconductor fabrication companies. At higher levels of assembly, companies such as Dell computer routinely use third-party service providers to build products and provide other services. Large companies use Manufacturing Enterprise Resource Planning (ERP) systems that are highly integrated and difficult to maintain and change; the state of practice has not yet migrated to modular services. Engineering data systems are dominated by the large CAD vendors, who are just beginning to offer tools as a service via cloud computing. Data interchange among CAD and CAM tools is difficult and prone to translation errors and ambiguities in design intent. Product data standards (such as STEP) exist, but have not been updated to semantic web standards needed for services to understand product and process information they receive. DoD legacy systems have huge archives of product data in the form of 2-D drawings, but newer systems have 3-D product data that can drive automated manufacturing with less human mediation.

In sum, the state of the art provides a few examples showing the feasibility of Model-Driven Manufacturing Services, but the state of practice is largely trapped in a traditional product oriented build to print paradigm.

2.2.4.5 Gaps

Model driven process planning and control: The notable gaps requiring applied research are as follows:

- Multi-scale manufacturing process modeling
- Standards for product and process model interoperability and reuse
- Semantic structures that allow process planners to understand design intent
- SOA and cloud computing libraries of product and process models as services

Basic research is required to understand the limits of modularity in structural and electromechanical systems, which differ in fundamental ways from the decoupled product and process architectures of VLSI (Whitney 1996).

Service-Oriented Manufacturing: The hard problems in applying SOA to the manufacturing domain arise from its differences with the IT domain. Advances are needed in the following areas:

- Semantic structures (Yang 2007) for domain-specific, machine-interpretable product and process information that can drive manufacturing services without the need for human mediation of ambiguities
- Smart registries of services that can operate on this information, and mechanisms for discovery and coordination with information that is distributed across the manufacturing community
- Design rules, reasoning engines and resource allocation approaches that can match product and process needs with available services and constraints
- Dynamic scheduling tools and automated brokers that can orchestrate supply chain and dynamic production planning solutions faster and better than humans
- Open and scalable interface standards and service level agreements that extend SOA to handle manufacturing information, and wrappers to make legacy systems compliant with the interfaces.
- Mechanisms to assure trusted services

These advances will require a multidisciplinary attack to integrate the best ideas from innovators in planning technology, knowledge representation, 3-D graphics and spatial reasoning, agent-based systems, business-to-business transactions, and standards. Implementation of the resulting technologies will require new engineering, manufacturing, and business practices. Demonstrations will need to show pragmatic business cases in addition to technical feasibility.

2.2.5 Complementary SysE Process, Property, Environment and Mission Models

The Model-Driven Manufacturing Services capabilities in Section 2.2.4 provide integration support for manufacturing product and process models. Similar integration is needed for the earlier systems engineering phases. The VESCCE environment and the Virtual Design and Modeling initiatives in

Sections 2.2.2 and 2.2.3 provide strong support for reasoning about the system product as it goes from high-level virtual system models through mid-level component designs to realized and assembled components. However, unless such product models are complemented more explicitly with systems engineering process, property, environment, and mission models, they can run afoul of serious risks, delays, and rework instead of achieving the desired 3X and 4X system development and modification speed goals.

The (Al Said, 2003) analysis of model clashes among a project's product, process, property, and success models on 35 risk-tracked projects found that only 30% of the model clashes and 24% of the risks were caused by product-product model clashes. 16% of the model clashes and 20% of the risks came from product-property model clashes (e.g., too many product features to develop within the promised schedule); 12% of the risks came from product-success model clashes (e.g., programmer-friendly vs. user-friendly user interfaces).

The virtual environment capabilities in Section 2.2.2, 2.2.3, and 2.2.4 provide strong support for distributed concurrent engineering. However, if there are not processes for synchronizing and stabilizing the concurrency, the project will founder. A key process capability for synchronizing and stabilizing the project elements and performers is the use of evidence-based (vs. schedule-based or event-based) management commitment reviews. This is a mature commercial practice, used within the AT&T family of companies since 1988 (Maranzano 2005). Top-level guidance for such evidence-based reviews is provided in such sources as the CMMI (Chrissis et al. 2007) and the Services' Probability of Program Success mechanisms (e.g., USAF 2007), and the beginnings of detailed guidance for evidence-based reviews is provided in (Boehm-Lane 2010). These can be added to traditional DoD reviews such as System Requirements Reviews and Preliminary Design Reviews, but the infrastructure for their general use has not been developed. Such infrastructure would include integrating the other classes of models (process, property, environment, and mission models) into a Model Integration Framework such as the one shown in Figure 10 (Wade et al. 2010b).

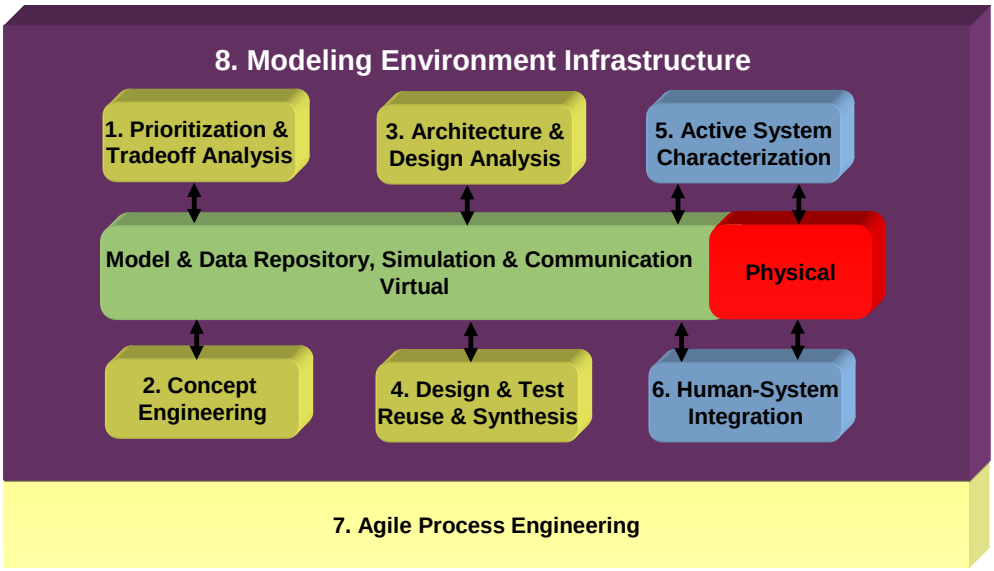

Figure 10. Model Integration Framework

In this figure, an MBE solution is comprised of an integrated, yet modular framework composed of capabilities in eight critical areas that are necessary to achieve the Systems 2020 objectives. The Prioritization & Tradeoff Analysis module provides the capability to input the particular factors relating to the relative value and priority of high-level capabilities of the system under development. The Concept Engineering module provides an interactive, collaborative, multimedia environment to multiple stake holders, along with a library of concept modules, and Reuse and Synthesis capabilities to quickly construct concepts of operation and other high-level abstract models of the system under development. Together these two modules provide critical capabilities in the areas of decision making and provide the front end to continuous verification and validation. The Concept Engineering research thrust noted in this report is roughly comparable to these two areas in this framework.

The Architecture and Design Analysis module provides the system architect and human operator with the ability to develop and reason about an architecture and design which supports the conceptual view while providing a resilient but not over-optimized solution based on the Prioritization & Tradeoff Analysis models described earlier. Design & Test, Reuse and Synthesis provides the means, by leveraging existing assets and utilizing computational capabilities, to rapidly translate high level abstractions into lower level ones. These capabilities can be used across the entire range of design and test abstractions from concept to implementation. The Modeling Environment Infrastructure provides the plumbing that supports the other tools. Together these three capabilities greatly communication and greatly improve reuse and synthesis. The Virtual Design and Modeling research thrust in this report is roughly comparable to these three areas with the addition of the Prioritization & Tradeoff capabilities noted above. The Architectural Patterns area in PBE is also closely related to the Architecture and Design Analysis, and Design & Test Reuse and Synthesis capability.

Active System Characterization has the role of providing feedback between the virtual and physical system domains. This module constantly monitors the actively deployed system and feeds back this

information into the model and data repository ensuring that that this information is up to date in near real time. Human-System Integration, true to its name, is integrated throughout the system lifecycle activities to ensure that the human considerations are accounted for and modeled with the end goal of optimizing the entire system, not just the technical and human subsystems and components. Agile Process Engineering provides the processes and governance to enable productive parallel development in each of the aforementioned areas. The Complementary Systems Engineering Process, Property, Environment and Mission Models in this report are roughly comparable to these areas of capability. The COD research thrusts are heavily dependent on the capabilities in the Active Systems Characterization and Human-System Integration modules.

2.2.6 MBE Thought Leaders

Table 3. MBE Thought Leaders

Thought Leader	Institution	Specialty
Frank Belz	Aerospace Corp.	Aerospace Systems Modeling and Analysis
Ray Carnes, David Sharp	Boeing	Systems of Systems Modeling
Don Firesmith	CMU-SEI	Architecture Modeling Process
Doug Schmidt	CMU-SEI	Real Time Distributed Systems Modeling
Mary Shaw	CMU-CS	Architectural Style Modeling
Arnob Ray	Fraunhofer	Model Verification Tools
Hassan Gomaa	George Mason U.	Real Time Systems Modeling
Russell Peak, Chris Paredis, Leon McGinnis	Georgia Tech	Model Based Simulation and Analysis
Doug Bodner, William Rouse	Georgia Tech	Rqts., Architecture Life Cycle Modeling
Grady Booch, Walker Royce	IBM	Model Based Languages and Tools
Ivar Jacobson	IJ International	Object, Aspect-Oriented Modeling
Peter Shames	JPL	Space Systems Modeling
Sanford Friedenthal	LMCO	Model Based Systems Engineering
Steven Corns	Missouri S&T	Concurrent Design and Development Tools
Daniel Jackson	MIT	Formal Modeling and Analysis
M. Auguston, C. Whitcomb	NPS	System Behavioral Models
Colin Neill, Tim Simpson	Penn State	Formal Modeling, Tradeoff Analysis
Daniel DeLaurentis, S. Landry	Purdue	Systems of Systems, State-Based Modeling
Rob Cloutier, Jon Wade	Stevens	Model Based Concept and Design Engineering
Michael zur Muehlen	Stevens	Business Systems Modeling and Analysis
Mike McGrath	Stevens (ANSER)	Integrated System Modeling and Manufacturing
Wayne Wymore	U. Arizona	Early MBE Theory and Application
Philippe Kruchten	U. British Columbia	Architecture and Process Modeling
Richard Taylor	UC Irvine	Distributed and Interactive Systems Modeling
Lori Clarke, Leon Osterweil	U. Massachusetts	Formal Process Modeling and Analysis
Robert Graybill	USC-ISI	Integrated Microelectronics Modeling& Fabrication
Neno Medvidovic	USC	Mobile Distributed Systems Modeling
Berokh Khoshnevis	USC	Model Based Structures Fabrication
Bruce Lewis	US Army AMCOM	Modeling Languages: AADL
Janos Sztipanovits	Vanderbilt	Real Time Distributed Systems Modeling
K.Y. Yim	Wayne State	Design Model Knowledge Management

2.3 Platform Based Engineering (PBE) Analysis

2.3.1 Introduction

The potential PBE game-changing research thrusts identified are:

- Adaptive product line architectures
- Agility platforms

- Architectural patterns

The first is concerned with cost-effective, rapid system development and tailoring in risk-mitigated fashion. The second is concerned with the ability to cope with uncertainties in the operational environment by "absorbing" or adapting to changes in resilient fashion (Madni-Jackson 2008). The third is concerned with accelerating development and adaptation in risk-mitigated fashion. The key distinguishing feature of *adaptive product line architectures* is the ability to rapidly insert and "test-drive" new capabilities and evolve systems as part of a product line. Included in this thrust are: *architectural patterns*, to facilitate automated reasoning about platforms to realize adaptive system behavior and seamless legacy integration; *principled first platform development approach* to overcome "the first generation dilemma;" *tradeoffs* between design for change and costs, between reuse and flexibility, and between near-term performance and system robustness; and *specification of common components* that can accelerate development and support evolution. The key distinguishing feature of *agility platforms* is the ability to conduct tradeoffs between *product line architecture and platform agility*. Included in this thrust are: *composable functionality and flexible interfaces,* to create net centric mission capability packages; *architectural patterns* to facilitate dynamic composition of capabilities and services; and explicit adaptation *mechanisms, to enable and support platform adaptability* in the operational environment (Madni 2008a, Salasin-Madni 2007).

The major benefits of PBE are substantial savings in development time and cost across a product line or family of platform-based systems. Commercial organizations have realized speedup factors of 4 to 10 in development of new systems (developing fast) and adaptation time for fielded systems (flexibility), with concurrent improvements in assurance, reliability, and interoperability.

2.3.1.1 Approach

Our overall PBE vision for Systems 2020 combines adaptive product line architectures, agility platforms, and architectural patterns to create a new class of adaptive systems with great speed and agility. This combination of capabilities is not only complementary but synergistic. Adaptive Product Line Architectures (APLAs) enable: a) reduction of product development cycle time and cost by starting with and capitalizing on a "partial solution" provided by the platform; and b) extension of the useful life of the platform by building in mechanisms for product line adaptation and evolution. Agility Platforms (APs) potentially enable fast and cost-effective adaptation to change (e.g., exploit an emerging technology, adapt to changes in regulation, policy, or threat characteristics). Architectural Patterns support both APLAs and APs by accelerating their development through reuse of patterns and in so doing reduce downstream implementation risks. Figure 11 presents a conceptual representation of the synergy between these three game-changing concepts, each of which requires technological advances. These concepts and required advances are discussed next.

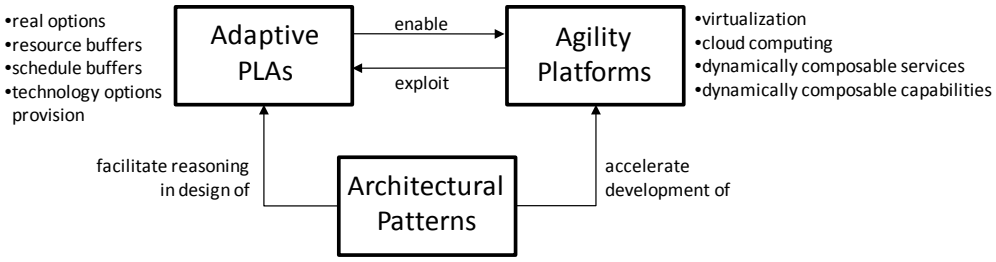

Figure 11. Vision for Next Generation PBE for S 2020

2.3.2 Adaptive Product Line Architectures

The U.S. Military needs fast, flexible and adaptable capabilities (and tools that enable the development of these capabilities) to succeed in unprecedented asymmetric conflicts during the conduct of irregular and hybrid warfare. To this end, adaptive product line architectures go beyond product line architectures (PLAs) in that they address the critical DoD need for *long-lived systems* that need to adapt and evolve (Madni 2008a,b). As such, adaptive PLAs are a game-changer for advancing the state-of-the-art in PBE for long-lived systems that need to *adapt to* and *evolve* in the operational environment.

The business case for Adaptive PLAs becomes quite apparent when one recognizes that the benefit of PLAs may come at the expense of agility and product differentiation. Adaptive PLAs are intended to overcome this drawback of PLA. The basic idea of adaptive PLAs is to anticipate likely changes in product requirements and either build in *"shock absorbers"* (e.g., as in critical chain planning and scheduling) or build in *adaptation mechanisms* as part of the delivered system. As importantly, adaptive PLAs make accommodation for technology substitution by carefully implementing real options at specific points in the product line life cycle. The latter essentially extends the life cycle of the product line.

In essence, a product line is a set of systems that share a common, managed set of features that satisfy the needs of a particular market segment or mission, and that are developed from a common set of core assets in a prescribed way. Product line methodologies consist of a robust set of practices that require initial investment in domain architectures and generalized components, but generally result in subsequent faster development, lower risk and lower costs; and often increased performance. In the commercial world, PLAs are used as frameworks for developing customer-specific applications in a particular domain. For example, all TV sets and mobile phones share some core set of capabilities, yet each model may have unique features. The PLA approach focuses on developing application families, rather than individual applications. From an economic perspective, the PLA investment pays off after a few systems (as few as 4 to 6 systems), but the savings accrued in the long run are more than worth the initial investment. Not surprisingly, companies such as HP (see Figure 7, Section 2.1.2 for a factor-of-4 speedup example), Nokia, and Motorola have embraced this approach, along with most automotive and electronics companies.

Product lines have the ability to impact key DoD objectives – *fast*, *flexible*, and *adaptable* systems – for the aforementioned reasons. Moreover, product lines can result in highly dependable, interoperable and reliable systems because components that are frequently reused are more likely to be robust.

Second, specifications of how to correctly use product line components are a fundamental element of a PLA.

A product line approach becomes a preferred approach when the intent is to build a family of systems. The key metrics by which this approach can be compared to others is in terms of cost, time to delivery, elimination of duplicate work, and quality.

Product line thinking becomes appropriate for the military when one comes to the realization that very few military systems are truly unique. It is important to realize that military platforms can be physical, informational, or a combination of the two. The latter poses a unique set of challenges. These include: creating a formal approach to PLA evaluation, including when to extend, evolve, or replace a PLA; developing a methodology based on product characteristics to include or exclude a product from a PLA. Beyond that, general methods, processes and tools are needed for domain engineering and re-engineering of legacy systems to become product line-compatible. Perhaps the main game that needs to change in DoD is the one identified for software in *megaprogramming* (Boehm-Scherlis, 1992) of having empowered product line managers, who have the resources to invest in product line capabilities and the clout to get individual developers to use them. The benefits of PLAs for PBE are:

- **reduced time to deployment**: Cummins, Inc. reports that systems that used to take a year to complete now can be turned out in about a week (Clements-Northrop 2003)
- **reduced cost; for example,** products in the National Reconnaissance Office's Control Channel Toolkit product line cost approximately 10% of what they otherwise would have
- **increased productivity**: for example Cummins estimates that they are turning out 14x number of products than before, while using two-thirds the software resources
- **superior quality**: each system is the beneficiary of defect elimination in its predecessors; higher developer and customer confidence; the more complex the system, the greater the benefit of having pre-solved performance, security, and availability problems
- **simplified training**: users conversant with working with one member of a product line can competently work with others
- **reduced logistics tail**: fungibility of components reduces the number of spares
- **increased competition:** product lines present an inherently horizontal market, whereby industries can flourish across application pillars. Reduced barriers to entry allow more vendors to compete.
- **superior leverage of human capital:** platform savings are, to a large extent, due to design reuse. This allows human capital to focus further up the value chain. This is a significant benefit as the US seeks to retain competitiveness both militarily and economically (Madni 2010a).

Given the lean years ahead for DoD, the Adaptive PLA thrust can be a potential game-changer. Adaptive PLAs go beyond PLAs in that they employ resource buffers and schedule buffers to absorb "shocks" and exploit real options to opportunistically exploit technology breakthrough. As importantly, adaptive PLAs are responsive to DDR&E imperatives: a) accelerate delivery of technical capabilities to win the current

fight; b) prepare for an uncertain future; and c) reduce the cost, acquisition time and risk of major defense acquisition programs. They achieve (a) through reuse of platform and core assets. They achieve (b) through capabilities to extend and evolve, enabling preparation for an uncertain future. They achieve (c) through reusable core assets, standard interfaces, and extensible/evolvable capabilities to handle eventualities and exploit technology breakthroughs.

The current state-of-the-practice is widespread use of product line practices throughout the software industry. A number of small, medium-sized, and large organizations have embraced product line approach in lieu of the traditional, isolated, product-by-product approach to software development. In the commercial sector, the markets covered include medical equipment, aircraft avionics, automotive products, and financial analysis systems. Organizations in these sectors report substantial gains in productivity, quality, and customer satisfaction. None of the so-called technical challenges to product line adoption are showstoppers. The biggest challenge is a management and organization challenge – the focus on short-term goals that pass up the opportunities to invest in product line assets and to reap the benefits as in the Figure 7 HP example.

The state-of-the-art is best exemplified by PLA uses in DoD. DoD organizations that have adopted the product line approach include: Navy's PEO Integrated Warfare System; National Reconnaissance Office; Naval Undersea Warfare Center (NUWC), Army's Technical Applications Program Office (TAPO); Army's Live Training Transformation effort; Navy's PEO for Submarine Warfare Federated Tactical System family of systems. One particular DoD project, OneSAF Product Line Architecture (PLA), bears some elaboration. The OneSAF Product Line Architecture provides tangible evidence in the real world to back up the projected impact of PLAs. OneSAF is the U.S. Army's next generation entity level simulation. As part of the acquisition and development of OneSAF, the U.S. Government used a task order acquisition plan. Under this plan, the government developed an initial Product Line Architecture Framework (PLAF) that was used to inform and guide the respective bidders for OneSAF Architecture and Integration contract that was let in 2001. The successful architecture and integration contractor is now responsible for the evolution and further development of the architecture.

The most spectacular platform in history is the Internet. The Internet is an example of a layered family of platforms made compatible by a series of protocols governing the transmission of bits, packets, messages, and levels of application services among nodes in hierarchies of networks. Similar layered families have become attractive ways of providing DoD net-centric services.

The deficiencies and gaps are mostly methodology, scope, and technology exploitation related (Salasin and Madni, 2007). By methodology, we mean, for example, developing formal criteria to include/exclude a product from a product line. By scope, we mean, for example, extending the concept of "services" and "products" from the information space to the hybrid (i.e., informational and physical) space. By technology exploitation, we mean incorporating the ability to opportunistically exploit technology breakthroughs. The desired state and actual state are defined as follows:

> **Desired State** – ability to apply, evolve, and refactor Product Line Architectures to systems comprising hardware, software, and networked informational and physical system (e.g., ISR,

weapons) and opportunistically exploit technology breakthroughs to extend the useful life of the product line and maintain product differentiation.

Actual State – ability to apply and extend Product Line Architectures to software and software-intensive systems, and relatively simple vehicle platforms (e.g., C-130 aircraft) re-outfitted for different missions.

The investment opportunity is in extending PLAs from purely informational or purely physical platforms to hybrid systems (i.e., systems spanning informational and physical systems and assets) and making the product line architecture adaptive.

2.3.3 Agility Platforms

Recent technological advances (e.g., virtualization, cloud computing, architectural patterns, product line architectures) have provided an unprecedented opportunity to create an Agility Platform (Madni 2008a,b). We envision an Agility Platform that combines virtualization with cloud computing and is enabled by architectural patterns. *Virtualization* is the ability to run multiple operating systems on a single physical system and share the underlying hardware resources. Virtualization allows complete transparency of computing resources and locations. These characteristics make it a key building block for an Agility Platform. *Cloud computing* is the provisioning of services in a timely (near-instant) manner on demand, to allow rapid increase or reduction of resources, the key to platform scalability. These characteristics make cloud computing a key building block of an Agility Platform. In the commercial world, virtualization is at technology readiness level (TRL) 8 and cloud computing around TRL 7. Architectural patterns have shown great promise in software development and enterprise integration. They need to be scaled up to address system issues. Product line architectures have shown great promise in the commercial world and are being seriously examined for Defense system applications. The combination of these technologies in an Agility Platform is a true game-changer for Platform-based Engineering. The Agility Platform, so defined, subsumes hardware, software, and networked systems, which is DoD's new expanded definition of the term "platform."

Virtualized platforms typically apply to IT systems, such as cloud computing. They also apply to systems-oriented architectures for heterogeneous databases. In the latter case, the disparate databases are mapped into a virtual database that can be exploited, indexed, and manipulated in a manner that given the appearance of single database with whatever virtual format the user, software, or human user desires. We contend that virtualized platforms can involve weapons, logistics, or any capability/service required by warfighters. Specifically, virtualization can be achieved in an even deeper and broader fashion than achieved today in that *physical* systems can be virtualized in the sense of providing services through net-centric systems. For example, in intelligence, surveillance and reconnaissance (ISR) one can extend systems-oriented architectures to include *asset management on demand,* whereby if the requested data is not available in data archive, it is immediately requested from the integrated ISR architecture. Likewise, if it is stored but not exploited, the system can launch an exploitation routine. To users, the only difference amongst these options will be the latency that they experience. In this section, we address the Agility Platform from the virtualization and cloud computing perspectives.

Virtualization is based on the Virtual Server Concept. Virtual servers encapsulate the server software away from the hardware. The encapsulation includes the operating system, the applications, and the storage for the server. One or more hosts can service a virtual server, and one host may house more than one virtual server. Virtual servers are still referred to by their functionality (i.e., database server, email server). In a properly architected environment, virtual servers remain unaffected upon the loss of a host. In fact, hosts can be removed and introduced almost at will to accommodate maintenance. Virtual servers scale easily, although there are ultimate limits. For example, if the resources supporting a virtual server are heavily taxed, then resources allocated to that virtual server can be adjusted. Server templates can be created in a virtual environment and then used to create multiple, identical virtual servers. As importantly, virtual servers themselves can be migrated from one host to another seamlessly. The advantages of the virtual server concept are: resource pooling, high redundancy, high availability, rapid deployability of new servers, ease of deployment, reconfigurability while services are running, and physical resource optimization by being able to do more with less. The drawbacks of the virtual server concept are: they are somewhat more difficult to conceptualize, and they are slightly more expensive in that the organization needs to purchase hardware, OS, applications, and the abstraction layer. Virtualization is now a well-established technology.

Cloud computing is one of the more promising technologies for Platform-Based Engineering. While there is no one single definition of cloud computing, for the purpose of this report, we use the definition from the National Institute of Standards and Technology (NIST):

> *"Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction"*(NIST 2010).

There are several reasons why cloud computing is gaining in popularity. Most noteworthy are the significant savings in capital investments that come about by switching to a cloud. Secondly, it is flexible, and scalable, whereby, the customer can request additional storage or processing capacity, and they are instantly available. Lastly, cloud-based services are modeled as "pay-as-you-go", hence, there are no upfront costs. Amazon and Google are the leading two providers of cloud computing services.

On the other hand, there is reluctance on the part of some organizations to embrace cloud computing. One of the reasons for lack of adoption by firms, concerns the privacy of data, which in a cloud is handled by third party firms. Perhaps as a consequence of this, firms and organizations have started developing their own private clouds. For example, the Army IT organization has stopped buying new servers, and is consolidating its existing infrastructure into data centers and clouds, virtualizing its servers, and is using applications developed for clouds by the Defense Information Systems Agency. Also, the Federal Government is recommending all its agencies to switch to a cloud-based platform for their information technology needs.

Cloud computing is complementary and synergistic with virtualization in that the organization does not have to own the hardware but can "rent" as needed from a cloud. It is important to realize that business

agility is more than scalability. It is elasticity. It is the ability to rapidly scale up and scale back. Rather than relying on a server cluster, Eli Lilly rapidly changed its healthcare business by using an Elastic Compute Cloud (EC2). With this technology, the organization can give back resources to the cloud when done. With cloud computing, various providers let the organization create virtual servers, and set up an account, perhaps just with a credit card. The organization can create virtual servers ("virtualization"). This process consists of: choosing the OS and software each virtual server "instance" will have; running the software on a larger server farm located elsewhere; instantiating more virtual servers on demand; and shutting down instances as needed in a matter of minutes. The providers then send the organization a bill for what the organization used. To summarize, cloud computing reduces costs, is scalable, and concerns for privacy can be addressed by developing private clouds. Virtualization is key to transparency. Together they provide an unprecedented opportunity to create an Agility Platform.

It is important to realize that the concept of Agility Platforms (Madni 2010b) and the attributes of virtualization evinced in cloud computing and systems-oriented architectures can also be applied to physical systems and services subject to certain constraints. Consider the context of tactical weapons usage in which a warfighter, in a direct action situation, is in need of fire support. In this case, the "cloud" is the entire aggregate of all weapons on all platforms available to the warfighter. The cloud may comprise aircraft carrying weapons on hardpoints or internal bomb bays, snipers, ERGM or other artillery launchers from Navy vessels, Army units, and long range missiles. In much the same way that an IT cloud has rules to determine which processors are available, a physical cloud can have rules that govern which "points" in the cloud are candidates for servicing a given request. As with traditional cloud computing, a key aspect of the solution is knowing/determining the extent to which resource allocation is going to be done in aggregate fashion, or in isolation. An example of the latter Isa memory-less greedy algorithm and variants. Note that physical virtualization requires net-centric connectivity of users and points of service. Loosely speaking, physical virtualization can be viewed as the dual of IT virtualization.

Cloud computing helps in achieving at least two objectives: Fast and Flexible. Clouds have high availability, and thereby, should be the first choice for mobile interactive applications. Secondly, using clouds can reduce parallel batch processing times (Ambrust 2009). Thirdly, clouds enable customers to use almost unlimited computational power; hence, it can be used for symbolic mathematics, which is very computing intensive. The projected benefits of the Agility Platform (i.e., virtualization-enabled cloud computing) within the PBE rubric are presented in Table 3.

Table 4. Agility Platform Benefits

Transparent	Total transparency (resources, locations, …)
Fast, Flexible	Rapid, cost-effective scale up and scale down
Low Cost	"Pay for use" lowers cost
Adaptable	Seamless migration from one host to another; readily extensible, cost-effective platform

There is ample evidence in the real world to substantiate these claims. Many companies offer virtualization, including VMWare, SUN (now Oracle), and Microsoft. Cloud computing is being exploited by Forbes.com. In fact, Forbes is hosting its website in Amazon's Elastic Compute Cloud (EC2) and "paying for use" only, while letting Amazon worry about the hardware.

The current state of the practice is wide adoption of virtualization and selective adoption of cloud computing in the real world (e.g., healthcare, finance). Research is not strictly restricted to academia, in that commercial companies such as Google, Microsoft and Yahoo Research have made significant strides in this arena. Broadly, speaking, research in cloud computing addresses four key improvement areas: availability and privacy of data, storage and organization of memory, operating systems and virtualization, and developing improved middleware for clouds. And finally, cloud computing is gaining acceptance in diverse industrial sectors due to its speed and scalability. For example, The New York *Times,* Activision, ESPN, and NASDAQ all use cloud computing at different levels. Government organizations such as the Army, and NASA (InfoWeek 2010)are also users of cloud computing for certain specific tasks.

Despite the advances in virtualization and cloud computing, an Agility Platform does not exist. Combining virtualization with cloud computing to create an agility platform requires advances in how best to optimize and integrate these two promising technologies. In addition, there are several key advances needed in cloud computing. These include: enhancing trust of cloud applications; cloud self-defense; delivering user-centric personal clouds; and reducing the power consumption of clouds. This is where a real opportunity for investment lies for commercial organizations, although they tend to address the non-defense market first.

2.3.4 Architectural Patterns

Architectural patterns are solutions to specific software and hardware development problems within a particular domain that can be applied suitably (with modifications) in many contexts. Architectural patterns are central to numerous aspects of software development, including security, usability, ad adaptability, and provide an important foundation for architectural styles, reference architectures, middleware platforms, and application frameworks. Current state-of-the-art research in architectural patterns involves automatic detection of patterns in models and code, and automatic advice delivery pertaining to the use of a particular pattern.

Architecture patterns are most useful and effective when they align organizations and system components. For example, the benefits of architecture patterns are enhanced through standardized

protocols, interfaces, languages, and models. Architecture patterns, if leveraged appropriately, provide the opportunity to uncover new ways to build platforms that are highly flexible, versatile, and adaptable. For example, PBE can potentially benefit from data mining tools that uncover latent patterns in architecture and find opportunities to reuse components and subsystems.

Although organizations are key to the successful use of architecture patterns, this report does not further explore how pattern analysis can be used for cultural and organizational change.

New, revolutionary computing models based on social, biological, and economic systems (e.g., crystal growth, auction markets) have shown enormous promise in the laboratory setting. These computing models have the potential to achieve unprecedented levels of scalability and dependability beyond models such as object-oriented, aspect-oriented, and component-oriented techniques. Systems that leverage revolutionary computing models are built using architectural patterns that mimic or emulate processes and structures found in nature and society. An example is the tile architectural style (architectural pattern), which is based on the tile assembly model of crystal growth.

To be effectively utilized by non-scientists for practical applications, platforms based on revolutionary computing models must be constructed so that engineers can specify, design, and implement systems using high-level, intuitive languages and assemble systems from reusable components and services.

The architectural pattern approach can be expected to produce systems faster, since less design time is needed. Secondarily, it will enhance flexibility, since there will be more capability in each of the subsystems by virtue of module reduction. The evidence that exists today in the real world to back up the projected impact (metrics) is limited to very confined problem sets, such as software. However, this is not a limitation of architectural patterns; rather, it means that patterns have not been scaled up to apply to systems.

Research in academia on architectural patterns, centers around: detection of anti-patterns, leveraging biologically, chemically and socially-inspired patterns, and using crystal growth and auction markets for creating highly resilient and autonomous systems (Madni-Jackson 2008). The theory behind pattern analysis in architectures is much like recommender systems, such as those used in Amazon.

Patterns for systems architecting are in the nascent stages. They are being discussed in the systems architecture community as a promising and important building block to systems architecting. For example, in TOGAF, patterns are viewed as a way of putting building blocks into context by, for example, describing a reusable solution to a problem (Version 9 2009). Building blocks are what one uses, patterns tell one how to use them, when, why, and what trade-offs need to be made. Pragmatically speaking, patterns can potentially help the system architect in identifying combinations of architectural building blocks that have yielded effective solutions in the past, and that can serve as the basis for effective solutions in the future.

Patterns for systems architecting can borrow and extend many of the concepts and terminology from software architecting, as shown in Figure 12.

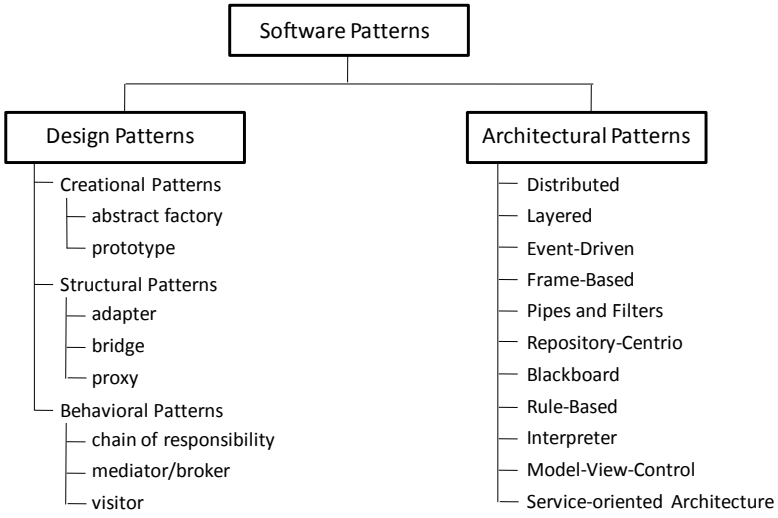

Figure 12. Exemplar Software Patterns [adapted from (Booch 2007)]

Software architectural patterns can either directly apply as system architectural patterns or can apply after re-interpretation of the pattern. An example of the former is model-view-control. An example of the latter is service-oriented architecture. SoA can be reinterpreted at the systems level as capabilities-oriented architecture where the "capabilities" can be delivered through hardware (i.e., physical) platforms or software.

Within the software community, a key challenge is discovering and codifying patterns for ultra-large-scale (ULS) systems and SoS. From a systems perspective, the current challenge is the transferability of patterns observed in software to the systems domain. In this regard, the patterns presented in Figure 10 should serve as a convenient starting point.

2.3.5 Overall PBE Summaries

2.3.5.1 Impact

From an overarching perspective, the major benefits of PBE are substantial savings in development time and cost across a product line or family of platform-based systems. Commercial organizations have realized speedup factors of 4 to 10 in development of new systems (developing fast) and adaptation time for fielded systems (flexibility), with concurrent improvements in assurance, reliability, and interoperability. In particular, adaptive product line architectures, agility platforms, and architectural patterns have the potential to be "game-changers" for the DoD. Adaptive product line architectures, while exploiting and exhibiting the well-known benefits of PLAs, will be able to extend the useful life of the product line by either "absorbing" changes or "adapting to" changes in the development environment. Agility Platforms will enable rapid and cost-effective adaptation to changes in the operational and regulatory environment. Architectural patterns will accelerate the response to change of the product line during development, and the adaptation rate of the deployed platform in the operational environment.

2.3.5.2 State-of-the-Practice and State-of-the-Art

The state-of-the-practice of PBE is best exemplified by the automobile platform. The bottom layer or the automobile "platform" is relatively fixed. The top layer varies to create different models of the automobile. The key advantage of this approach is that it is usually less expensive to design a new car model based on an existing platform than it is to design the model from scratch. The relative ease of making changes is the result of the flexibility afforded by platform-based design. Automobile platforms are examples of physical, energy-centered systems. Such platforms have only two layers, unlike information platforms that tend to have three or more layers (Moses 2010).

In the same vein, there is some evidence that the joint Strike Fighter (JSF) program started out with a platform focus with a high degree of commonality between the carrier (U.S. Navy), vertical-take-off (USMC), and long-range (USAF) versions of the same aircraft. Gradually the amount of commonality between the variants eroded because the variants were more different than originally envisioned based on prototype tests, and/or because common standards and policies were not enforced.

Lockheed Martin implemented a successful platform strategy for its family of military transport aircraft. The series of Block Upgrades (major aircraft and avionics revisions), software and avionics (major subsystems) were expected to become common for Lockheed's three platforms, the C-130, C-5, and C-27. The common subsystems were to form the basis for a new airframe platform eventually replacing the C-130. The new airframe platform was expected to be similar to today's aircraft in that it would be reconfigurable to fill a wide variety of different roles. (Today's C-130 fills roles as diverse as Search and Rescue to Gunship applications, all leveraging a common airframe. A variety of software and weapons suites can be added to the airframe with cost impact ranging from low-end to high-end.)

Today PBE is beginning to be used in industries other than traditional product engineering. PBE approaches and platform-based product family design are now being adopted in "non-traditional" sectors such as telecommunications, food and drug industries, and service systems (e.g., entertainment, tourism, banking). Initially, this expansion occurred by companies marketing existing product and service derivatives to fill current and readily exploited niches, but future product development is expected to be conducted using product platform strategies. For example, Cingular Wireless has implemented a platform approach on pre-paid and their "Take Charge" cellular plans.

2.3.5.3 Critical Areas of Research

Designing a product platform and corresponding family of products is a difficult proposition in that it embodies all the challenges of product design with the added complexity of having to coordinate the design of multiple products in an effort to increase commonality across the set of products without compromising their distinctiveness and competitive advantage. Due to these difficulties, product family designs and PBE have been primarily conducted in ad hoc fashion. Research is needed to make PBE more rigorous from both methodological and technology exploitation perspectives. Some of the specific technical challenges that need to be overcome especially for DoD platforms are presented in Table 4.

Table 5. Technology Gaps

Model-based approaches for product family design	Total transparency (resources, locations, …)
Enabling architectural approaches	To enable o systematic tradeoffs analysis o sensitivity analysis o error tracking o statistical analysis and propagation of uncertainties o requirements and specifications cascading that afford simultaneously subcontractor flexibility while enforcing accountability
Rigorous cost-benefit analysis	Based on mechanistic product models and mission effectiveness models
Quantitative modeling	Commonality, product performance, market demand, revenues, and manufacturing costs

To appreciate these technology gaps, we need to put PBE into perspective for DoD/military applications. To begin with, the typical lifetime of a platform, especially a military platform, spans multiple product generations. Thus, one of the key challenges is to either develop the ability to "predict the future" – a daunting undertaking – or design platforms in a manner that accounts for both expected and unexpected changes down the line. The key questions that need to be answered in this regard include:

- Where to introduce flexibility in the platform?
- When to design a flexible platform and when to opt for a new platform?
- How to value the cost of flexibility?
- How to prepare for the introduction of new applications/modules, new product lines, and radically new technologies?
- What criteria to apply to determine when to initiate platform redesign before it loses competitive standing?
- How to define what goes into a platform for both traditional products and non-traditional applications?
- How diverse are the set of variants that can be derived from a common platform?
- How to exploit product line architectures without compromising platform agility?
- How to conduct and manage multiple tradeoffs in platform design?
- How to ensure that the acquisition process supports the ability to generate benefits across multiple systems and across a system's lifetime?

Answering the aforementioned questions is crucial for institutionalizing PBE in DoD and military systems development.

2.3.6 PBE Thought leaders

Table 6. PBE Thought Leaders

Thought Leader	Institution	Specialty
Mark Maier, James Martin	Aerospace Corp.	Architectural Frameworks and Patterns
Samuel Ajila	Carleton U. , Canada	Software Product Line Evolution
David Garlan	CMU-CS	Reuse Critical Success Factors
Paul Clements, Linda Northrop, Grady Campbell	CMU-SEI	Product Line Architecting and Development
Martin Griss	CMU West	HP Product Line Architect
David Weiss	Iowa State	Lucent Telecom Product Line Architect
Jeff Poulin, Will Tracz	LMCO	Product Line Economics and Architectures
Cihan Dagli	Missouri S&T	Families of Systems Architecting
Olivier DeWeck, Joel Moses	MIT	Platform Based Design and Engineering
David Bixler, Neil Siegel	Northrop Grumman	C4ISR Product Line Engineering
Erich Gamma	Object Tech Intl.	Design Patterns
Tim Simpson	Penn State	Product Family Design
Don Reifer	Reifer Consulting	Software Product Line Design and Analysis
Ted Biggerstaff	Software Generators	Product Line Domain Engineering
Brian Sauser	Stevens	Platform Based Engineering Processes & Tools
Susan O'Brien	UA Huntsville	Product Line of Systems Planning Tools
Ralph Johnson	U. Illinois	Design Patterns
Neno Medvidovic, Gaurav Sukhatme	USC	Robotic Product Line Architecture Tools
Barry Boehm, JoAnn Lane	USC	Product Line Total Ownership Cost Modeling
Azad Madni	USC, ISTI	Platform Based Engineering Processes & Tools
Don Batory	U. Texas	Software Infrastructure Product Line Engr.
Kevin Sullivan	U. Virginia	Rapid Multi-Platform Model-Based Synthesis

2.3.7 Critical Areas of Research

To realize the game-changing capabilities embodied in adaptive PLAs, Agility Platforms, and Architectural Patterns, research investment is needed in several areas. The critical research areas are presented in Table 7.

Table 7. Critical Research Areas

Adaptive Product Line Architectures	• mechanisms for introducing real options to enable downstream adaptation of PLA to required changes in cost-effective fashion • methodology for designing resource and schedule buffers for "minimum perturbation" product line • methodology for determining how far a product line can be adapted before it loses its value proposition and competitive advantage
Agility Platforms	• develop tradeoffs analysis methodology to analyze multiple tradeoffs impacting platform agility • develop tradeoff analysis framework to evaluate potential compromises between product line scope and platform agility, and between product line robustness and platform agility
Architectural Patterns	• scale proven software architectural patterns to systems • specify integration patterns for hardware-software integration and human-system integration • develop reasoning mechanisms based on system architectural patterns

2.4 Capability on Demand (COD) Analysis

2.4.1 Introduction

COD involves technologies for significant improvements in system adaptability to unforeseen threats and opportunities. Fully realizing the advantages of rapid field-adaptability involves several critical-success-factor elements:

1. Building in reserve capacity. Frequently, "optimized" designs operate dangerously near a drop off point in system performance (e.g., computer-communications capacity; carrier-based aircraft weight). COD facilitates such adaptation to change.

2. Detecting threat or opportunity trends, patterns, or clusters. Example capabilities are recommendation engines, automated agents, machine learning, crowd sourcing, and combinations of human and machine pattern recognition and "bad-smells" intuitions.

3. Analyzing options for countering threats or realizing opportunities. Examples are advances in game theory and model-generated simulations, and emerging use of multicore chips and parallel processing for rapid concurrent option generation and analysis.

4. Rapidly reconfigurable components and services. Examples are rapid manufacturing of new components via Model Based Engineering, rapid adaptation of product line variability via Platform Based Engineering, and user-programming of C4ISR and battle management capabilities.

5. Rapidly and concurrently verifying and validating reconfigured components and services. Examples are model-driven simulation, test case generation, and test execution and analysis.

6. Associated processes and incentives for rapid adaptation to change. Recent fixed-price, build-to-spec contracting for rapidly changing systems of systems have encountered serious problems and delays. This is outside the technical scope of Systems 2020, but is a critical enabler or disabler of successful adaptation to change.

Three key COD game-changing opportunity areas are presented below:

- Rapid Human-Computer Adaptation
- Engineering Adaptable Systems
- Embedded Sensors and Computing.

2.4.2 Rapid Human-Computer Adaptation

2.4.2.1 Opportunity

The incidence of unanticipated challenges to DoD systems is increasing, as is the complexity of decision situations. Difficulties arise from committing to human decisions made with limited opportunities to evaluate numerous complex options. Emerging commercial technologies and research results provide attractive opportunities to leverage the relative advantage of humans to deal with ambiguity and the relative advantage of technology to enable speed of processing and information visualization. Thus, wherever possible, there is great benefit to be derived from having synergetic human-computer option formulation and analysis before committing to a decision.

2.4.2.2 Approach

Many unanticipated challenges and opportunities are preceded by leading indicators that can be subjected to combinations of autonomous analysis and inference, human-collaborative option generation, and synergetic human-computer option analysis. The approach here is to bring together the strengths of adaptive systems (for aggregation, recommender systems, and visualization) and emerging capabilities on the human side (such as crowd sourcing, virtual collaboration technology, social networking, and virtual exercise technology). As a result, combined human-computer intelligent systems will present the user with a set of options linked to needed and available resources to achieve a desired result in a specified environmental context. The leading indicators of unanticipated threats and opportunities will be more rapidly detected and analyzed, a greater number of response options will be generated and analyzed, and decisions will be more rapidly negotiated among the key stakeholders due to their participation in the process.

2.4.2.3 Impact

Syntheses of artificially intelligent and human-driven systems will dramatically improve situation awareness, situation understanding, and decision option analysis to inform cognitive and decision processes. Adaptation to complex environments and missions will be faster and more effective.

2.4.2.4 State of Practice and State of the Art

The state of the practice largely succeeds by drawing on domain knowledge to produce relevant automated decision suggestions reviewed and adjusted by human decision-makers in the domain. A good example, entering early use, is the Army-USC Institute of Creative Technology's virtual-reality Army leadership training system (Swartout 2010). In cases, such as the DARPA Pilot's Associate program, rapid human-computer adaptation has brought about significant improvements in flight safety and mission effectiveness.

The current state of the art involves more sophisticated combinations of artificial intelligence reasoning and learning techniques and more sophisticated human-computer interaction techniques, such as social networking technologies. A recent paper summarizes critical success factors in generating innovative and effective human-computer solutions (Lane et al. 2010).

2.4.2.5 Gaps

Autonomous computing and software can quickly converge on solutions, but their weaknesses in autonomous agent collaboration and commonsense reasoning make it too easy for them to commit the system to a bad decision that may be very hard to undo once discovered.

A variety of additional gaps pervade the problem space and drive each approach to different degrees. These gaps include the following:

- Limited ability to address diversity in prioritization of resources across stakeholders and actors in the operational environment, i.e., who can control/influence/inhibit availability of resources;
- Lack of sufficiently intelligent tools to understand human cognitive functions and embed them as part of the system;
- Lack of socially intelligent tools that can interpret behaviors and trends to cue decision makers;
- Insufficient ability and tools to elicit and define needs in a common semantic framework that bridges diverse stakeholder lexicons to enable wide-ranging analyses of alternatives.

2.4.3 Engineering Adaptable Systems

2.4.3.1 Opportunity

With advances in technology, fielded systems continue to become more complex, and new systems are deployed into an ever-evolving complex operating context with legacy systems. Much of this complexity is related to designing for an increasing number of mission contingencies. As systems build in reserve capacity, and more options for countering threats and realizing opportunity, they grow in complexity. These pre-planned contingencies, added throughout acquisition and development, make the system more robust to those contingencies, but may make it less adaptable to truly novel situations at the tactical edge. Complexity due to foreseen possibilities limits adaptability to un-foreseen actualities.

Recent advances in the study of complex adaptive systems allow us to add organic embedded adaptability, and make trades between flexibility and adaptability in system architecture and design. System metrics and engineering tools based on these research advances will allow large system

integrators and acquisition decision makers to effectively and quickly manage a wider range of design possibilities. Further, these metrics will provide much greater insight into the complex adaptive DoD systems we build and deploy so that flexibility, adaptability, and complexity can be managed throughout the life cycle.

2.4.3.2 Approach

Psychologists and sociologists have studied subjects playing computer simulations with complex interaction of underlying variables, some of which are hidden. This work aims at understanding successful and unsuccessful strategies and techniques for managing complex problems both individually and in teams. One striking result is the tendency to isolate variables that are coupled and even mathematically sophisticated subjects tend to make linear estimates of properties that evolve geometrically (Dorner 1996). Human first-order solutions to complex problems generally fail due to neglect of second and third-order phenomena. Another important insight is that in complex policy situations, decisions judged wrong historically are in fact more cogent in terms of the bad assumptions that the decision makers implicitly or explicitly held—the wrong policy is logical from the point of the flawed context (Margolis 1987). This convincing body of work emphasizes the need for appropriate metrics and continuous outcome prediction via modeling and simulation. Without these metrics and tools we cannot develop the associated processes and incentives for rapid adaptation to change.

There is a convergence of research in complex adaptive systems from several different points of view. Research on multi-agent games, self-optimizing systems, and control in complex engineered and biological systems sheds new light on how adaptable systems can be engineered. We can use this research to more quickly and effectively engineer systems with embedded organic adaptability. Current research application areas are very diverse including physical condensed phase systems (Feldman et al. 2008), biological systems (Csete-Doyle 2004, Tanaka et al. 2005), economic systems (Anon 2007, Goeree et al. 2006, Wolfers-Kitzewitz 2004),psychological and social systems (Dorner 1996), policy and decision-making (Margolis 1987), and technology/network systems (Willinger et al. 2009).

Research is now moving well beyond “toy problems” to synthesize analysis, modeling, and empirical data from real world systems. This maturity in selected research disciplines is the basis for our approach. First generation system metrics, supporting tools, and adaptable system patterns can be developed by closing gaps between current research and engineering practice in the near term with appropriate investment. These advances will help us succeed in analyzing options for countering threats or realizing opportunities.

Multi-agent Games. Classical approaches attempt to completely predict outcomes within limited scenarios for each agent, usually with stringent assumptions on agent behavior. In the past, the emphasis was on closed form solutions or simulation of relatively simple mechanisms to gain insight into real system behavior. Mechanism design is now developing rapidly as researchers look more to real world systems (Jordan et al. 2010), consider broader ranges of assumptions (Wolpert 2006), and use combined empirical game data with simulation (Jordan et al. 2010).

Today mechanism design is literally exploring a wider space of rules, agent behavior, communication, and protocols to model real world complex system behavior. This allows us to better understand how the actual system would behave as the architecture is varied. There are several features of this research that are compelling for application to DoD systems. First, we are able to gather empirical results from games with human agents and combine that with models and simulations of different mechanisms, so in some sense we have a new and different ability to capture and analyze the human in the loop as part of the system model. Second, within certain bounds, results may be extrapolated beyond the empirical measurements so that unforeseen alternate scenarios can be considered very rapidly. Finally, the concept of combining empirical data (prototyping), analysis, and modeling (simulation) is quite familiar to experienced system engineers. With this approach, we can create system metrics and engineering tools that work in a different region of the design space where complexity due to pre-planned scenarios can be traded against "what if's" either at design time or once the system is deployed. In fact, we can now explore stability and robustness of the system in a different way. This allows us to design a margin for adaptability.

Implicit in empirical mechanism engineering is the large amount of data now available about the behavior of systems and the behavior of operators using systems over the Internet. It is precisely this sort of telemetry that has allowed the empirical approach to flourish. Certainly available bandwidth and computing power continues to increase, but the collection and processing of this sort of broadband system telemetry could pose problems for DoD due to security and last mile bandwidth considerations. On the other hand, there is outstanding opportunity to understand how systems of systems are evolving, and to model the impact of mission or environment change. Also these risks can be mitigated in training and simulation situations. Gaming based on existing trainers and simulations may also allow us to assess the impact of new systems on legacy architectures.

Self-Optimizing Systems. Self-optimization in complex adaptive systems is evolving in a similar way and moving from simple models with closed form solutions to consider extended approaches that are applied to problems formerly considered intractable. The approaches include multi-agent reinforcement learning with incremental feedback (Singh et al. 2009), a variety of Monte Carlo methods (Wolpert 2006), evolutionary approaches that mimic biology (Wolpert-Macready 2005), and surprising self-modeling methods in robotics (Bongard et al. 2006). Once again, the key to progress appears to be adding heuristics to analysis and combining empirical data from experiments or sensor measurements to assist in predicting system behavior. Just as in multi-agent games, there is the ability to self-optimize using empirical data rather than merely an empirical test after optimization. In fact, the line between multi-agent games and self-optimization is rather blurred though the techniques, and to some extent the preferred application areas, are still distinct. These self-optimization approaches give us insight into the definition and "measurement" of system level performance metrics as we explore the adaptability vs. flexibility trade space at a relatively high level of system abstraction.

Control in Complex Systems. An interesting line of research has clarified a number of issues in complex adaptive systems science. John Doyle and his collaborators have made studies of complex natural systems (biological systems) (Csete-Doyle 2002), and complex engineered systems (modern technological systems) (Carlson-Doyle 2002). The earlier complexity science archetype was the chaotic

dynamical system whose system behavior appears complex though the equations of motion are very simple [Kauffman 2000]. Now we see serious investigation of the behavior of real world complex systems that are predictable and controllable over a range of environments. In all of these systems there is a trade between the adaptability of a complex system and its fragility to specific classes of failure. With increasing commercial investment and accelerating research pace in systems biology and bioengineering, there is an excellent opportunity to further explore architectural patterns from dynamic biological systems for use in successfully engineering complex systems (Ingeber 2008). This is analogous to the impact that Alexander's study of patterns in architecture and city planning had on pattern programming in software development (Alexander 1977; Gamma et al. 1995).

2.4.3.3 Impact

This game changer gives us a new set of metrics and a new set of tools to manage adaptability, flexibility, and complexity during system acquisition and design. This allows us to

- Rapidly make critical acquisition decisions using flexibility and adaptability metrics
- Embed organic adaptability into our systems to respond to changes at the tactical edge
- Speed development by effectively managing complexity (flexibility vs. adaptability)

In both engineered and biological systems the overwhelming majority of system complexity is created to deal with internal contingencies (robustness) and external contingencies (flexibility) rather than primary functions or missions (Alderson-Doyle 2010). Further, we know that requirements volatility and architecture rework are major impacts on acquisition and development. If we manage complexity by making flexibility vs. adaptability trades, we can reduce time to deploy and field systems with improved adaptability to unforeseen threats and opportunities.

2.4.3.4 State of Practice and State of the Art

The state of practice for engineering adaptable systems is largely identifying and implementing planned or foreseen contingencies. Adaptability is primarily an issue of technology insertion, and a narrow range of site-to-site installation or operating issues. Methods for dealing with complexity or emergent phenomena are best described as a craft, with ad hoc tools used by each engineering team.

The state of practice in multi-agent games commercially is focused on economics, using restricted models of behavior (rational agent) and simple equilibria (Nash 1950) with a narrow view of information and preferences. Mechanism engineering uses nearly optimal design methods in these restricted economic settings including combinatorial auctions, resource allocation (Yaiche 2000), and contract theory (Wolfers et al. 2004). In particular, this technology has been used effectively in spectrum auctions (Anon 2007). While some game based simulation of complex systems has been done, more general methods and insights for engineering adaptive systems have not been developed.

State of the art research in multi-agent games addresses issues such as robust implementation, non-parametric mechanisms, computational and distributed mechanism design, and exploring equilibrium manifolds (Nash, Mean-Field, QR, Distribution based) (Wolpert 2006, Jordan et al. 2010, Bongard et al. 2006). Robust implementation seeks to design mechanisms that are robust to misspecification of the

system environment. By understanding and exploring equilibrium manifolds, designers can understand which system outcomes are viable, identify improving paths in the outcome space, and create mechanisms to follow those paths. Non-parametric mechanisms are also valuable when the designer does not have enough information about how distributed decisions lead to outcomes.

The state of the art in self-optimization includes self-inferencing robotic optimization (Bongard et al.), reinforcement learning (Singh et al. 2009), Monte Carlo methods (Wolpert 2006), evolutionary approaches (Wolpert-Macready 2005), and metric approaches based on computational mechanics (Crutchfield 1994, Feldman et al. 2008, Shalizi et al. 2004). The research trend is toward models that can explore parameter and architectural variants to discover an objective function for the system. Research is also very active in self-optimizing hardware or more properly self-optimizing hardware software subsystems (Kephart-Chess 2003). This is generally called autonomic computing, and these self-reporting and self-diagnosing capabilities are appearing in commercial offerings. However, autonomic control issues at the system level remain challenging because of emergent behaviors due to control interactions between autonomous subsystems. In fact, with autonomic hardware and software coming online, we must be aware of a new generation of hacking attacks based on spoofing. We do not want fielded systems to be vulnerable to adversary behavior that manipulates an emergent autonomic response of the system.

The state of the art in control in complex systems is summarized in a recent thoughtful paper (Alderson-Doyle 2010). We are in a transitional phase, and although there is not yet a unified theory, the research community is moving away from over-simplified models to a more considered view of how to understand and engineer complex adaptive systems. It is already clear that there are control, communication, and protocol motifs or patterns of architecture in successful complex systems--those that can operate over a range of conditions, and persist over periods of time. At present, these are merely observations that lead to general design rules, and the work is largely network control oriented. However, comparisons across biological and engineered systems demonstrate similar motifs and there is work to formalize the architectural patterns of robustly controllable complex systems (Csete-Doyle 2004).

2.4.3.5 Gaps

The primary gap is that current research in multi-agent games, self-optimization, and control in complex systems must be tailored to DoD problems, and there has not been a research focus on adaptability except as self-optimization. With the exception of some Internet based economic mechanisms, there has not been enough available empirical data to fully develop the potential of these methods.

The best strategy to close this gap is funding applications of multi-agent game research and related self-optimization research to a pilot program. Stage one investment builds a combined empirical/simulated model for a pilot system using the pilot system architecture for mechanism constraints and “telemetry” supplied from use of the pilot system. By telemetry, we mean transaction and state data for the system, and interview data with some subset of users and operators. Planning and executing this pilot requires collaboration between researchers in multi-agent games and self-optimization, and system engineering professionals with appropriate mission domain knowledge for the pilot program.

An initial pilot for this investment is technology application to a trainer, simulator, or exercise model where trainee and system telemetry are used to model/analyze the core system architecture(s) that are the subject of the training/simulation/exercise. This is a low risk, data rich environment with potentially a wide variety of systems engaged in a given training scenario. Another approach is to pilot an existing program, where telemetry from the operational system, users, and administrators is used to build the empirical database, and the existing system architecture is used to constrain the mechanism research. In both cases we examine the adaptability of an existing system using self-optimizing systems research to develop key measures of effectiveness for the system architecture. The combined empirical and simulation approach allow us to analyze variations of the system architecture, the environment, and the mechanism beyond the empirical measurements.

A secondary gap is that the engineering of complex adaptive architectures does not yet have a well-developed methodology, set of design rules, and patterns. The pilot application will allow us to more effectively explore the complex design space, but we also need to close the gap in adaptive architecting. For instance, when we compose autonomic and machine-learning systems, what patterns of communication, control, and interaction (negotiation rules) will avoid emergent issues of deadlock and race conditions? Investments in architecting complex adaptive systems theory will extend current work to focus on defense issues and also extend the analysis beyond control theory to discover adaptive patterns relevant to our missions. We will compare and contrast how the motifs and patterns of complex DoD systems compare with commercial and biological complex adaptive systems. An added benefit is that as we improve our ability to architect flexibility and adaptability, we get much better measures of reliability/interoperability of adaptive systems as well.

The initial payoff from these investments is identification of adaptability vs. flexibility (complexity) metrics that can be used to more effectively manage other programs. In fact, identifying metrics based on early pilots makes it possible to better define appropriate system tools to simulate, measure, and analyze these metrics. In concert with these evolving metrics and tool sets, we will be developing and evaluating architectural building blocks for future complex adaptive systems.

As we invest to fill these gaps, the craft of dealing with complex adaptive systems can move toward becoming an engineering discipline using metrics, tools, and adaptive system patterns.

2.4.4 Composable DoD Components

2.4.4.1 Opportunity

A major obstacle to rapid DoD systems adaptability is the lack of composability of legacy components or subassemblies that are being integrated into a new DoD system or reused as adopted system components. These have been developed with complex interdependencies such that they cannot be decomposed if only a part of their capability is needed. Further, when they are adopted, they often include subfunctions that interfere with the execution and performance of the system adopting them.

On the other hand, the commercial world is developing and making available numerous consumer applications for geolocation, search, music, recommendations, etc., and people have become

accustomed to having capabilities on demand from their cell phones, smart phones, net books, laptops, as well as desktop computers. If one needs a new application or service, one just looks on the Internet, finds it, downloads it, and often composes it with other services. This is now an expectation on the part of the general public, but is poorly met in existing DoD environments. The next generation of DoD engineers and warfighters will have grown up with cellular, computer, and gaming technologies and will be extremely innovative in how they use these technologies.

2.4.4.2 Approach

The key to success in such composability is to develop service-oriented architectures and components (Marks-Bell 2006). The components have self-contained capabilities and come with metadata about their composability characteristics. Hardware components will come with their size, weight, power, environmental constraints, and services performed. Informational components will come with metadata about their inputs, outputs, interaction protocols, and performance. These and the self-contained nature of the components can enable systems engineers to rapidly compose hardware-software capabilities on demand. Such capabilities would enable DoD users to combine software-intensive capabilities, communications support, and sensor devices into systems to be rapidly deployed, managed, upgraded, and adapted to unforeseen situations.

2.4.4.3 Impact

The introduction of such service-oriented embedded sensors and computing will provide users with small, lightweight devices (e.g., smart phones, computers) with network access, supported by connectivity that is rapidly reconfigurable and composable based on fast-changing user needs. These devices can be linked together via mobile secure networks or access the world-wide network via cell phone towers. The devices will employ an internal infrastructure that allows users to pull in, as well as generate, new applications (similar to iPhones and Google phones) while reconfiguring and integrating these applications. The ability to find quickly appropriate/desired applications will be managed by recommender-like systems similar in concept to those used by Amazon and other on-line businesses. The infrastructure will include self-healing and self-optimizing capabilities and will utilize low power chips such as those recently announced by Qualcomm to reduce power needs. To complete the picture, alternative power sources will be available to extend basic power capabilities.

2.4.4.4 State of Practice and State of the Art

In terms of state of the practice, smartphone manufactures and developers have defined and publicized an infrastructure that allows people to quickly develop new applications and interface to a multitude of devices that can be downloaded and integrated quickly by the user on demand. This advance represents some gains and limits:

- Simple transactions and services are relatively easy to compose;
- Transactions with complex semantic content will not be easy to compose;
- Algorithms with data dependent computation are not well modeled;
- Open source software is available to experiment with low barriers to entry;
- Low power, power aware networks are demonstrated in various prototypes;

- As many users have found, there are many devils in the details in attempting to compose applications across competing vendors;
- Tools are under development to detect common incompatibilities in component composition (Boehm-Bhuta 2008).

2.4.4.5 Gaps

To introduce this capability further requires advances in mobile networking, rapid reconfiguration of hardware and software, low-energy equipment, and small-footprint, readily transportable devices. Populating a DoD environment with commercial capabilities will require the kinds of new assessment and assurance capabilities to be discussed under Trusted Systems Design. Major challenges will involve composability with and among legacy systems. However, capabilities are emerging to transform legacy software into service-oriented components at IBM (Hopkins-Jenkins 2008) and SEI (Lewis et al. 2008).

2.4.5 COD Thought Leaders

Table 8. COD Thought Leaders

Thought Leader	Institution	Specialty
Anne Marie Grisogono	Australia DSTO	Complex Adaptive Systems
Richard Pew	BBN	Human-System Integration
John Doyle	Caltech	Complex Adaptive Systems
Shelley Evenson, Bonnie John	CMU	Human Computer Interaction Modeling
Jeff Conklin	CogNexus	Stakeholder Collaboration Models and Tools
Harold Booher	Consultant	Initial Army MANPRINT Director
Paul Eremenko	DARPA	Complex Systems Adaptability Design
Doug Bodner, William Rouse	Georgia Tech	Organizational Modeling for Adaptability
David Parkes	Harvard	Mechanism Design
Alistair Cockburn	Humans&Technology	Agile Methods
Irene Greif, Michael Muller	IBM	Computer Supported Cooperative Work
Owen Brown	Kinsey Tech	Space Systems Adaptability Design
Judith Dahmann	Mitre	Systems of Systems Adaptability Engineering
George Hazelrigg	NSF	Decision-Based Systems Engineering
Terri Griffith	Santa Clara U.	Team Tools and innovation
Ali Mostashari	Stevens	Complex Adaptive Sociotechnical Systems
Richard Turner	Stevens	Agile Methods
Abhi Deshmukh, Marty Wortman	Texas A&M	Distributed Decision Systems, Risk Analysis
Sarah Sheard	Third Millenium Sys.	Complex Adaptive Systems Engineering
Kent Beck	Three Rivers Inst.	Agile Methods
Don Saari	UC Irvine	Decision Systems Engineering
Milind Tambe, Yigal Arens	USC-CS, ISI	Multi-Agent Systems ; Game/Decision Theory
Barry Boehm, Azad Madni, Ann Majchrzak	USC	Rapid Interdisciplinary Collaboration Models and Tools
J.B. Dugan, Kevin Sullivan	U. Virginia	Environment Data Analysis and Adaptation
Paul Collopy	Value-Driven Design Institute	Value-Driven System Adaptation
Darin Ellis, Gary Witus	Wayne State	Adaptive Human-Computer Interaction for Teleoperation

2.5 Trusted System Design (TSD) Analysis

2.5.1 Trusted System Design (TSD)

2.5.1.1 Opportunity

Trusted System Design is facilitated by trust assurance criteria and associated measurement techniques for trust verification. TSD is a continuously expanding set of standard system security architectures for application by system engineers engaged in architecture and design activity. Architecture security solutions will be tightly coupled to the system architecture pattern, enabling transparency in the security effectiveness of design alternatives and associated metrics in multiple domains. These include,

but are not limited to system of systems, continuity of communications, data tracking and tracing, and campaign mission assurance.

Systems engineering trust methods, tools, and standards are expected to provide:

- Motivation for trust assurance solutions at the systems function level
- Trade space analysis techniques that focus on continued operation of critical mission functions in the face of damage to those of lower priority
- Recognition of trade-space security issues via architecture patterns and corresponding security models
- Trade-space calculation methods that enhance quantified benefits of mission critical functions
- Rating system for trust features using factors such as administrative overhead and interference with system functionality.
- Technologies appropriate for consideration in the context of trade-space frameworks, by which is meant various real-world domains and associated trust scenarios

These trust models and metrics will ultimately allow systems to operate in the context of a current and accurate attribution of trust. The security functionality that implements trust measurement will be usable both in the engineering of the system as well as in its testing and independent assessment, both in development and operation. For example, the system of interest would be able to use these techniques to verify its own trustworthiness according to standard criteria used to define that trustworthiness. These criteria may be related to measurable system attributes such as the integrity of configuration and interfaces and mission readiness in the face of evolving threats. The same criteria should be applied to the system of interest or to its communication or interaction with other systems that employ the same or similar techniques. The techniques must be applicable to a wide variety of system models and architectures while avoiding pitfalls due to composability and transitivity.

2.5.1.2 Approach

This approach to TSD changes trustworthiness requirements from *a single "ility" that may be sacrificed in favor of core functional requirements* to being *integral to the satisfaction of those core functional requirements*. It is also a game change in that trust-relevant requirements, whether automatically generated or manually specified, are currently often limited to control measures (security controls, safety controls, fault tolerance controls, etc.) described in standards or policy documents.

In this new approach, specification, architecture, and design of the system will incorporate not only system features but the method and extent to which those features may be able to be trusted. Systems engineers responsible for trust analysis would be accountable for ensuring that trust requirements are considered at the same point in the design process as any other functional component. They would be expected to calculate and weigh system benefits resulting from alternative trust architectures. These trust benefits would be transparent to all engineers on a system design team, and decisions made on system components or interfaces would be evaluated in part based on their trust impact.

The game change should facilitate fast development of trusted systems by providing parameters with which to measure the benefits with respect to trust. The solution must be flexible and adaptable to a variety of systems architectures.

2.5.1.3 Impact

This approach will facilitate faster development of trustworthy systems by tightly integrating trust considerations into the mainline engineering process. Evidence-based trust assurance cases would-be quickly incorporated into all trade-space alternative considerations (ISO 2009). The existence of tangible trust criteria and measures will provide the parameters needed to weigh system benefits with respect to trust assurance.

The approach will replace a checklist approach to security based on standards documents (Ross 2007) with system trust performance metrics. The approach will provide methods, tools, and procedures for verification and reverification of trust attributes throughout the stages of the system engineering process, resulting in a verifiably trusted system.

The solution must be flexible enough to work effectively across the wide variety of current system models and architectures, and engineering methodologies. It must be adaptable enough to maintain validity in an environment where models, architectures and methodologies fluctuate.

2.5.1.4 State of Practice and State of the Art

There are a few alternative ways to view the state of the practice in this field. Identity and authentication trust models usually rely on customized identity and authentication techniques that take advantage of trusted platform module (TPM) features of COTS chips[1](Irvine-Levitt, 2007). Systems operational trust models usually rely on change control, data integrity checking, system responses to vulnerability assessment, and/or security log analysis(Kim-Love-Spafford 2008). Trust certification and accreditation currently rely on security assessment standards (Swanson 2001, Johnson 2004, CommCrit 2009), and wait until completion of development to begin, rather than performing incremental certification.

The state of the art in trusted systems is the incorporation of trusted computing modules into systems architecture (Trimberger 2007). Design is a multi-step process that first identifies the critical program information (CPI) that, if compromised, would adversely impact systems operation (DoD 2008). The second step is to isolate access to the information in hardware chips that segregate memory in response to commands issued by customized software processes (Trimberger 2007). Common applications for these techniques are identification, authentication, and encrypted communication. This type of model could be extended into a trust library that made use of the hardware-embedded keys to verify trust properties such as the integrity of system configuration and software binaries. A combination of verified measures could form a trust metric. Trust metrics candidates under consideration in today's literature include, but are not limited to, size of community, symmetry, transparency, degree of control,

[1] Such as Intel® Trusted Execution Technology, see http://www.Intel.com/technology/security.

consistency of presentation, offsets from expectation, and component integrity (Herzog 2010, Holstein-Stouffer 2010, DoD 2009). Given that cyber attack and defense is a two-sided game, metrics could also include negative effects on attackers, and trust strategies may benefit from the game-theoretic approaches discussed under Capability on Demand.

2.5.1.5 Gaps

The type of trust model proposed has two components: trust evidence and trust computation (Jiang-Baras 2004). Both sets of features require research to ensure adaptability, but a flexible model using state of the practice techniques for trust evidence and state of the art techniques for computation should be possible to develop fast. This approach has been applied to demonstrate that metrics for flexibility in space systems engineering can be used to establish design criteria that will allow maximum utility during the lifecycle of a system (Nilchiani 2007) Similarly, a base set of metrics could be established that enable some trust, and additional metrics may be added as they are developed.

2.5.2 TSD Thought Leaders

Table 9. TSD Thought Leaders

Thought Leader	Institution	Specialty
Drew Hamilton	Auburn U.	Cybercombat Modeling and Simulation
Gary McGraw	Cigital, Inc.	Security Systems Engineering, Risk Analysis
Julia Allen, Rich Pethia	CMU-SEI	Threat Analysis, Intrusion Detection
William Scherlis	CMU-CS	Efficient Assertion Checking
Steve Bellovin	Columbia U.	Network Security; Authentication
Fred Schneider	Cornell U.	Trusted Systems
Vic Basili, Lucas Layman	Fraunhofer-MD	Systems of Systems Trust Analysis
Dorothy Denning	Goergetown U.	Security Systems Engineering
Steve Lipner	Microsoft	Secure Information Infrastructure
Jeff Voas	NIST	Security Systems Engineering, Risk Analysis
Trent Jaeger	Penn State	Information Theoretic Security
Gene Spafford	Purdue	Trusted Composition
Shari Pfleeger	Rand Corp.	Security Systems Engineering
Ron Rivest	RSA, Inc.	Public Key Security Engineering
John Viega	Secure SW Solutions	Security Systems Engineering, Risk Analysis
Jennifer Bayuk	Stevens	Security Systems Engineering
Brian Sauser	Stevens	Resilience Systems Engineering
Karl Levitt, Matt Bishop	UC Davis	Vulnerability Analysis and Defense
Matt Blaze	U. Pennsylvania	Trust Management; Secure Systems Design
Cliff Neuman, Terry Benzel	USC-ISI	Network Security, DETER Testbed
Len Adelman, G.J. Halfond	USC-CS	Public Key Security Engineering, Data Security
Barry Horowitz	U. Virginia	Secure Systems Engineering, Risk Analysis
Anita Jones, William Wulf	U. Virginia	Cyber Security

2.6 Systems 2020 Program Risks

The successful realization of proposed S2020 benefits requires the mitigation of several program risks to achieve full return on investment:

- **Risks of insufficient DDR&E-external stakeholder buy-in.** Success-critical DDR&E-external stakeholders include potential DoD Systems 2020 technology users, potential DoD research community research sponsors, the defense industry systems engineering community, and systems engineering tool vendors, who may see tool interoperability as a threat rather than an opportunity. These risks can be mitigated via early pilots that involve the external stakeholders, utilize mature technologies that are more attractive to tool vendors, and outreach to organizations such as NDIA and INCOSE.
- **Risks of either prematurely or belatedly adopting desired technologies.** These risks can be mitigated by early careful assessment of the maturity of technologies of interest relative to DoD user needs, requiring evidence of technology maturity as well as providing role-model pilot projects to accelerate maturation, and by pro-actively monitoring need and technology trends.
- **Risks of failing to identify and address emerging DoD needs and opportunities.** These risks can be mitigated by careful progress monitoring of System 2020 research and piloting results, such as scalability of tools and applicability to legacy systems and systems of systems, and again by pro-actively monitoring need and technology trends.
- **Risks of Technology/Governance/Culture/Team Behavior mismatches.** Examples of such mismatches could be development of platform based technology while preserving a stovepipe based acquisition culture and set of practices; development of concurrent engineering technologies while preserving a sequential system acquisition process and set of contractual provisions and incentives; and developing capabilities for rapid adaptability to change within a glacial contract change management structure (on two large systems of systems, changes involving contact modifications averaged 141 calendar workdays).The mitigation factors include using early pilots to identify stress points between rapid and traditional approaches; coordination with initiatives to streamline DoD's acquisition processes; training, mentoring, and hiring experienced individuals; and establishing incentive structures rewarding rapid adaptation to change.

These risks are addressed in the next section, Recommended Systems 2020 Next Steps (Section 3.1).

3.0 Recommendations

3.1 Recommended Systems 2020 Next Steps

The world of technology that Systems 2020 will improve is as dynamic and unpredictable as the world of defense, whose projects Systems 2020 will support. Understanding the concomitant risks as identified in Section 2.6, the set of recommended next steps initiates a flexible approach that allows immediate start of a few high-potential pilot programs and evolves an overall 2020 plan based on early pilot experience and pro-active approaches to identify and address changing threats, new requirements, and emergent technologies.

The approach actively involves DDR&E-external stakeholders in the Systems 2020 definition process; assures a mix of committed research sponsors and performers pursuing high-payoff new technologies with forward-looking user organizations piloting their use; and open solicitation of best ideas and calibration of their maturity with respect to users' needs. It begins by immediately initiating high-potential pilots who start with a strong technology base but with expressed needs for improvement; counterpart technology agents helping them identify, experiment with, and evolve to a next-generation level of velocity, flexibility, and adaptability.

These pilots will then provide an experience source for establishing and evolving a full-up Systems 2020 program definition and strategy; including coordination between developing high-payoff technologies and developing and applying acquisition practices that stimulate their use. Additionally, a set of potential pilot organizations for mature components available from System 2020 technology areas is presented to support the approach.

3.1.1 Pilot-based Approach

Here is a recommended series of steps for executing an evolutionary Systems 2020 course of action.

1. Identify promising DoD research and development (R&D) organizations to partner with – those early-adopters with high needs for the Systems 2020 capabilities and a willingness to serve as project collaborators in piloting the technologies, as well as those most interested in co-sponsoring Systems 2020 research. A proposed way to start would be to contact and interview candidate organizations, provide them information on System 2020 technology areas, and identify a group of people to act as their Systems 2020 technology agents; this could be an internal group, a research contractor or support provided by a local university. These agents would understand the needs and resources of the organization, become closely involved in the technology development activities of the System 2020 initiative, support Systems 2020 planning, and identify user programs for piloting technologies as they mature. The interviews will also identify immediately actionable pilot projects.

2. Based on current knowledge and additional information from these interviews, initiate 2-4 of the identified pilots where the technology is sufficiently mature to support the user's needs. As those pilots are initiated, the Systems 2020 agents will continue their work with identifying

specific user program needs and maintaining contact with the technology research activities to supply input to the Systems 2020 baseline planning team.

3. In concert with the System 2020 technology agents, begin baseline planning for each selected pilot activity. Invite researchers and tool vendors to describe and demonstrate their current and planned capabilities with respect to the pilot projects' expressed needs. Evaluate the contenders on their technology maturity, relevance to the pilot organization's needs, growth potential, and compatibility with other contenders' capabilities. Choose the best combination of initial improvement capabilities, apply them on a pilot project, and evaluate the results.

4. Based on the initial pilots' results, develop a realistic but evolvable baseline Systems 2020 roadmap. This will involve working with the relevant early adopters and research co-sponsors to award a mix of basic research, exploratory development, and advanced development projects focused on the highest-payoff prospective results. Revise the roadmap to reflect further insights on needs, opportunities, and priorities. Based on observed stress points between new-technology approaches and existing acquisition practices, collaborate with the acquisition community to develop, experimentally apply, evaluate and refine compatible next-generation technologies and acquisition practices.

5. Hold semiannual progress checkpoint reviews, annual project reviews, and adjust initial plans and the overall Systems 2020 roadmap as appropriate. This would include further solicitations for emerging and maturing new need and technology areas.

Figure 13 provides an example concept of operation for such pilots. The pentagons in the middle represent DoD RD&E Centers, each of which would work with a research-community Systems 2020 technology agent. The agent would be familiar with both the RD&E Center's missions, projects and current technology, and the levels of research and advanced technology available and being pursued by relevant technology providers at the left of the diagram. The RD&E Center and technology agent would identify the technologies most relevant to the Center's user needs. Together they would design one or two programs to create an evolutionary series of increasingly capable and well-integrated new Systems 2020 systems engineering methods, processes and tools. Each evolutionary increment would be tested in a pilot application, its results evaluated, and priorities determined for pursuing the next increment of capability. Preferably, the methods, processes, and tools would be developed in ways that they could be easily tailored for use by other R&D Centers or provide a component for a different evolutionary program, as illustrated by the red vertical arrows.

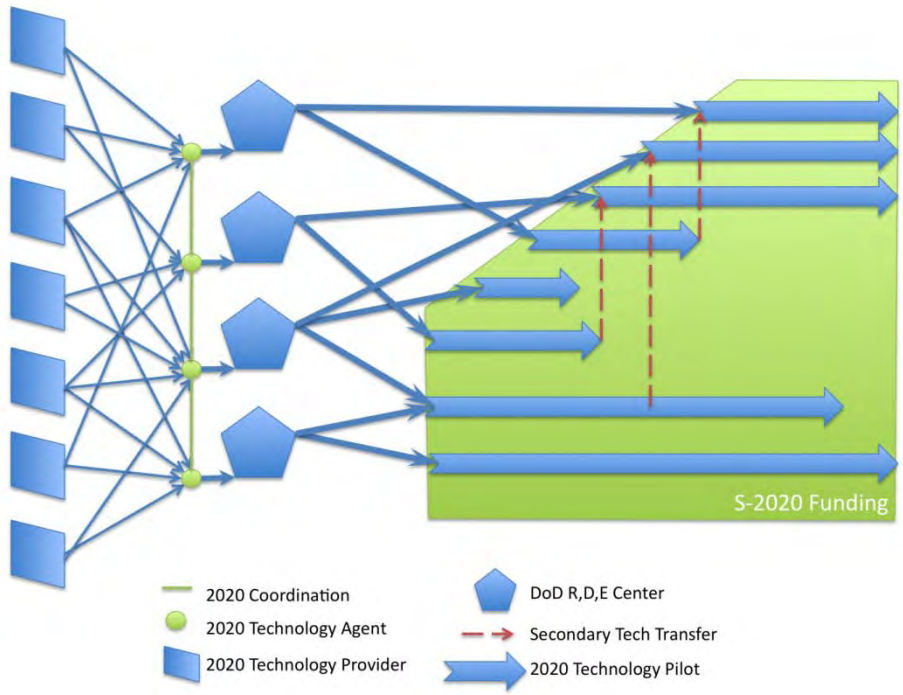

Figure 13. Pilot Program Coordination

An initial example of this approach has been developed in discussions of Systems 2020 pilot projects in the space domain between USC as a technology agent and Aerospace Corporation and the Air Force's Space and Missile Systems R&D Center. Aerospace has a significant Concept Design Center (CDC) that consists of a large room with roughly 30 workstations that are operated by specialists in structures, propulsion, power management, communications, reliability, cost, operational concept formulation, etc. These are supported by numerous models to accelerate the preliminary design and feasibility evidence generation for alternative solution concepts.

The CDC has been used on over 100 space projects, and can usually telescope two years of traditional sequential interactions between users, customers, and solution specialists into 60 days of collocated collaborative effort that explores more options and produces better satellite system solutions. However, the discussions of potential Systems 2020 pilot projects have identified several areas of CDC improvement that the R&D Center would like to pursue in all four of the Systems 2020 areas. They would like to use metamodeling capabilities developed at USC and elsewhere to better integrate their various specialty models. They would like to have better rapid concept formulation and virtual collaboration capabilities and tools being developed at several universities and companies. They are finding that their satellites are increasingly operating in Cyberspace and need more cybersecurity capabilities. They would like single or multiple space platforms with composable components that could be rapidly reconfigured for addressing unforeseen circumstances. They are considerably ahead of most DoD R&D Centers in having their current CDC capabilities, and are open to developing the extended capabilities in ways that they could be exported to and tailored by other DoD R&D Centers. Several similar combinations of DoD R&D Centers and technology-agent counterparts are available, such as the

University of Alabama in Huntsville-AMRDEC modeling and simulation capabilities, the Stevens-NAVSEA surface naval ships modeling and simulation capabilities, and the Wayne State-TARDEC ground vehicle modeling and simulation capabilities.

3.2 Description of Potential Pilot(s) for Innovative MBE Approaches

Model Based Engineering, comprised of VESCCE, Virtual Design & Modeling, Model Driven Manufacturing Services, and Complementary Process, Property, Environment, and Mission Models game changers represents four separable and rather significant R&D efforts. It is recommended that for the early pilots these be treated as distinct areas to be evaluated in parallel. Ultimately as the MBE based projects mature, all four game changers should be evaluated in their entirety by a single pilot. Depending on the level of funding and stakeholder interest, an additional very early pilot that would try to implement the full MBE paradigm using today's tools could be very beneficial to flush out any hidden technical and/or policy issues that may only be discovered by going through the process. This early pilot could help adjust and prioritize out year R&D objectives and funding.

Specific pilots are currently under evaluation but early candidates are listed in Table 10 below.

- Surface Naval Ships: NAVSEA is focused on an Affordable Future Fleet, which is centered on capability (war) driven design, and that encompasses service life as key attribute of fleet design. The suggested research will focus on approaches and technologies that enable model based ship design using life cycle costs and service life as key tradeoff criteria. The suggested research focuses on the reduction of design complexity and making realistic investment decisions under uncertain mission scenarios.

- Ground Vehicle: TARDEC offers several pilot opportunities from theater driven upgrades, like the Mine Resistant Ambush Protected (MRAP) vehicles, to new ground vehicle developments (JLTV & GCV). Suggested flow would be to start with an upgrade pilot and then migrate to subsystem and finally to full system MBE pilot as the System 2020 technology and tools mature. A natural pilot owner would be the Army though TARDEC.

- Missile: Since missiles are well-defined systems with a lower degree of complexity than aircraft, they are a natural candidate for a MBE pilot. This pilot would focus on the integration of all the components (operations, performance analysis, product design, process design, manufacturing analysis, supportability) from a stakeholder point of view (integration of the disks). A natural pilot partner/owner could be Raytheon Missile Systems.

- Jet Engine: Jet engines provide a unique opportunity for a creation of a pilot. A jet engine pilot would explore how far one can model a very sophisticated product with very complex cold and hot section physics requiring multi-physics/multi-scale simulations at high levels of fidelity. A natural partner for such a pilot could be Pratt & Whitney.

- Electronic Warfare Systems: Rapid spirals of EW system upgrades make this a good domain for both MBE and PBE. Focus is on virtual design and modeling, with particular emphasis on

electromagnetic compatibility with the battlefield system of systems, open architectures, rapid manufacturing services and trusted supply chains. Natural partners would be ONR and PM JCREW Systems Engineering for future Naval Fleets: Establish a collaborative Navy/CISD-DDR&E/SERC pilot project to develop and apply systems engineering methods for decision making under uncertainty and impact of complexity on costs to affordable fleet modularization. An initial pilot application could address ship structural design standards and ship machinery control systems commonality.

- Space Systems: Aerospace Corporation has a highly effective Concept Design Center that includes numerous models of space system structures, materials, propulsion, power supply, communications, sensors, attitude control, orbital analysis, reliability, cost, and schedule; and corresponding ground station capabilities. They are very interested in SERC architecture and metadata capabilities to enable more rapid configurability and composability of the models.

Table 10. Proposed MBE Pilot Projects

Proposed Pilot	Pilot Owner	Level of Engagement	Description	Maturity
Surface Naval Ships	NAVSEA	Closely aligned	Focus on mission driven capability design, couple with service life of the fleet	Approaches successfully applied in commercial aerospace and automobile industries
Ground Vehicle	Army (TARDEC)	Closely aligned	TARDEC offers several pilot opportunities from theater driven upgrades MRAP) to new ground vehicle developments (JLTV & GCV	Pilots and System 2020 technology will increase in sophistication over time
Missile	Raytheon Missile Systems	Loosely Aligned	Focus on the integration of all the components from a stakeholder point of view	Disk models fairly mature, system tool and date multi-scale not very mature
Jet Engine	Pratt & Whitney	Loosely Aligned	Path finder pilot would explore how far one can model a very sophisticated product with very complex cold and hot section physics requiring multi-physics/multi-scale simulations at high levels of fidelity	Product mature but pilot will try implementing MBE with today's tools

Electronic Warfare Systems	Navy (ONR and PM JCREW)	Closely aligned	Rapid system upgrade spirals make this a good domain for both MBE and PBE. Focus is on Virtual design and modeling, with particular emphasis on electromagnetic compatibility with the battlefield system of systems; Open architectures; and, Rapid manufacturing services and trusted supply chains.	TRL 7+ for applying these concepts to current EW product lines. Less mature for emerging agile multi-function RF systems
Space Systems	Aerospace Corporation	Closely aligned	Also interested in PBE and ways to reconfigure composable components in space	Concept Design Center has supported over 100 space system engineering efforts

3.3 Description of Potential Pilot(s) for Innovative PBE Approaches

Our vision of Platform-Based Engineering comprises Adaptive Product Line Architectures, Agility Platforms, and Architectural Patterns. These three "game-changers" represent distinct but complementary efforts. Depending on sponsor interest and funding level, any or all can be pursued. Specific pilots (also called out in Table 11) currently under evaluation are:

- **Unmanned Aerial Vehicles:** The Air Force would like to develop approaches to improve UAV design for multiple mission scenarios. Several UAVs have been developed for either a single, or limited number of targeted missions. This pilot will explore a platform based approach to designing multi-mission UAVs, by implementing standard, scalable technologies, modularity and design margins to accommodate rapid adaptation to multi-mission environments.

- **Surface Naval Ships:** NAVSEA is focused on an Affordable Future Fleet, which is centered on capability (war) driven design for the future fleet, using a platform-based approach to support multiple and changing mission scenarios. The suggested research focuses on the reduction of design complexity, and making realistic investment decisions under uncertain mission scenarios.

- **DDR&E Electronics Platform:** Advances in electronic components are increasingly enabling multi-purpose electronic systems with inherent adaptability. These include highly flexible and capable hardware (e.g., wideband front-ends, arbitrary waveform generators, substantial back-end processing) and software-driven functionality (e.g., modes, techniques, signal processing to extract data product) that can be rapidly updated. The opportunity that exists stems from the blurring of radar, EW, communications, and navigation systems with a high degree of flexibility and adaptability. A DDR&E thrust to ensure U.S. leadership in the design, development, and

testing of next generation flexible, adaptive, electronic systems would provide a substantial impact on national security.

- **Ground Platform:** TARDEC offers several pilot opportunities from theater-driven upgrades to new ground vehicle developments (JLTV and GCV). Starting with an upgrade pilot, one of these could be migrated to a subsystem and ultimately to a PBE pilot as Systems 2020 technology and tools mature.

- **Test and Training Platform:** The Test Resource Management Center (TRMC) is interested in "discovering technology" for Unmanned and Autonomous Systems Test and Evaluation (UAST). TRMC is interested in investing in Autonomous Test technologies that represent the fifteen-year challenge of testing systems without growing computational intelligence at the component, system, system of systems, and mission level. We have the opportunity to develop a Test and Training platform that features next generation instrumentation and enables full-spectrum Electronic Warfare Testing. Initial discussions between Dr. Madni and TRMC personnel relative to Systems 2020 appear most promising.

Table 11. Proposed PBE Pilot Projects

Proposed Pilot	Pilot Owner	Level of Engagement	Description	Maturity
Ground Vehicle	Army (TARDEC)	Closely aligned	TARDEC offers several pilot opportunities from theater driven upgrades MRAP) to new ground vehicle developments (JLTV & GCV	Pilots and System 2020 technology will increase in sophistication over time
Missile	Raytheon Missile Systems	Loosely Aligned	Focus on the integration of all the components from a stakeholder point of view	Disk models fairly mature, system tool and date multi-scale not very mature
Jet Engine	P&W	Loosely Aligned	Path finder pilot would explore how far one can model a very sophisticated product with very complex cold and hot section physics requiring multi-physics/multi-scale simulations at high levels of fidelity	Product mature but pilot will try implementing MBE with today's tools
Electronic Warfare Systems	Navy (ONR and PM JCREW)	Closely aligned	Rapid system upgrade spirals make this a good domain for both MBE and PBE. Focus is on: Virtual design and modeling, with particular emphasis on	TRL 7+ for applying these concepts to current EW product lines. Less mature for

			electromagnetic compatibility with the battlefield system of systems; Open architectures; and, Rapid manufacturing services and trusted supply chains.	emerging agile multi-function RF systems

3.4 Description of Potential Pilot(s) for Innovative COD Approaches

Key criteria for COD pilots are the following:

- Continuing streams of unforeseeable threats. Examples are the SOCOM and RRTO organizations in the table below. Other examples are electronic warfare, cited as a candidate MBE pilot above; and cybersecurity threats to be determined under TSD.
- Major benefits from shortening operational timelines. An example is the F-35 Autonomous Logistics System in the table below.
- Limited upgrade capability and need for self-generated adaptation. Examples are satellites and submarines.
- Many independently-evolving external interfaces. Examples are net-centric systems and systems of systems.

Table 12 describes the proposed pilots for COD.

Table 12. Proposed COD Pilot Projects

Proposed Pilot	Pilot Owner	Level of Engagement	Description	Maturity
Mobilization of Social Networks for Force Protection	SOCOM	Closely aligned	Building on the DARPA Network ("Red Balloon") Challenge, show field-adaptive capability for incident location and reporting.	Successful demonstration in CONUS
Reconfigurable/Adaptable UAV Capabilities for Non-traditional Missions	Rapid Reaction Technology Office (RRTO)	Closely Aligned	Use pre-defined UAV capabilities/services in unanticipated circumstances.	Systems analysis based on MOEs has limited success.
Embedded Sensors & Computing and/or Autonomic Computing	USAF F-35 Autonomic Logistics Information System (ALIS)	IOC under development	Detect needs for maintenance during flight, notify maintenance center to have fixes ready when aircraft lands	Well under development; further supply chain extension opportunities

Space Systems	Aerospace Corp.	Closely aligned	Needs for self-adaptive space mission performance adjustment, V&V of self-adaptive systems	Extensive work in autonomic satellite anomaly detection

3.5 Description of Potential Pilot(s) for Innovative TSD Approaches

Potential pilots for innovative TSD approaches are under discussion. They often involve security-level considerations.

Appendix A: Systems 2020 Workshop at INCOSE International Symposium (July 13, 2010)

Hosts: Barry Boehm (USC) & Art Pyster (Stevens)

Participants: Jon Wade (Stevens), Rick Adcock (Cranfield; UK MoD), Sarah Sheard (Third Millenium Systems), Bud Lawson (Consultant), Hillary Sillitto (Thales-UK), Sandy Friedenthal (LMCO), Daniel Hastings (MIT), Richard Beasley (Rolls Royce)

The meeting began with a brief overview of the current state of the Systems 2020 discussion by Barry Boehm. The participants were asked to review and comment on the presented material and identify any gaps.

Overall, the participants had a positive reaction to the initiative and responded well to the goals, as stated in terms of desired improvement (time-to-first article, flexibility, adaptablility).

In order to more adequately address these goals, the participants had the following observations:

- The recommended technology solutions (and others discussed below) have the potential to provide many benefits. However, **without organizational change as well as technology investment, DoD will not be able to achieve the goals set out for Systems 2020**. It was observed that as much as 50% of the time in the average DoD acquisition project was spent waiting for decisions. **Streamlining the decision-making process and ensuring that the recommended technology solutions also support decision-makers will be critical to meeting the desired goals**.
    - Some of these initiatives will take a lot of time and effort. Participants indicated that DoD should not wait to change the organization when this technology is available. In general, people now aren't ready to understand how to use these tools. Current efforts should include an initial look at how people work now, help them to understand that their jobs are to look for alternatives and do analysis of alternatives and embed simple tools for visualization now. This will give both a better answer now and, over time, get people ready to use these technologies when they are available.
- Participants voiced some concerns that not enough time has been spent on the problem space to truly understand requirements of the department. However, they agreed that the given goals (time-to-first-article, flexibility, adaptability) seem reasonable.
- An additional area which may be considered in order to support the Systems 2020 goals is rapid prototyping; this area may help to improve cycle time.
- The idea of having a standard platform (either in modeling or general system constructs) is useful. However, it is time-consuming to both build and adequately maintain this; it is not always easy to make the business case or to readily 'see' the ROI for this.
- Lean approaches may be helpful in achieving the Systems 2020 goals. According to research, the average defense program could see up to a 50% improvement in schedule and cost. But it is critical to use lean principles correctly.

- The technologies recommended to approach System2020 goals are useful. However, they do not align with the current approaches to acquisition. For example, change-processing decision time required does not enable easy flexibility in design. Two specific recommendations discussed were to find ways to shorten change-processing decision times to enable the use of more rapid development methods, or to have parallel teams doing stable current-increment development and agile rebaselining of next-increment specs and plans.
- MBSE: Modeling can be an incredibly useful tool. However, there were some cautions and constraints.
    - In general, models are good but shouldn't be used outside of their designed purpose without potentially high negative consequences. If not used correctly, modeling will not give gains toward faster delivery because any time gained will be lost in multiple iterations to correct the models.
    - The reuse of modeling was also discussed.
    - In general, the requirements process as currently structured is a huge time sink. If this is incorporated into modeling, there may be a major compression point for time.
    - Modeling the environment is critical. Improving physics-based models is critical to making MBSE successful.
    - Modeling work done outside of delivery timeline will compress the delivery schedule.
    - There is potentially a huge modeling payout in shortening time required for IV&V. However, this requires that testablility considerations be incorporated in the modeling environment.
    - Different model management approaches should be examined. For example, it was suggested that different functions could own their components of a model, instead of specific organizational components owning the entire model.
    - Models should be inter-connected. Some new technologies are emerging which will translate one modeling language into another. Having this type of platform that allows a single model to flow across the life cycle is critical to improving model use. (No longer looking at several models in which information is not shared and changes are not propagated).
- Reuse and the development of reusable components in a standard platform has huge potential benefits and has some proven success in the commercial world (e.g. Rolls-Royce Commercial Marine Division). However, to enable reuse not only in the design process but in the operational process (the "Lego" concept), training and education of users is a critical component which can't be ignored in the acquisition process.
    - Change management (configuration management) is a critical component in reuse.
- Economic modeling (value-driven design) should also be considered. Processes that look at how uncertainty relates to value in terms of the desired flexibility and adaptability will be important for success (i.e. research should examine the trade space between flexible/adaptable and the uncertainty associated with these).
- Possible research areas to support Systems 2020:
    - how to model the 'goodness' of system components
    - trade-space between flexibility and uncertainty/risk
    - smallest testable part that can be trusted in hardware, software, human components, etc. (This could also include consideration for the balance between small, predictable components and fewer, larger components—the variety/parsimony paradox.)

Appendix B: Rapid Application Development (RAD) Opportunity Tree

The RAD Opportunity Tree is a hierarchical taxonomy of sources of cycle time reduction, which can be used as a framework for assessing various mixed strategies for tailoring a rapid application development approach to a given organization's environment, culture, technology, and constraints.

This taxonomy is developed in the context of software development as an activity network of tasks with backtracking, as illustrated in Figure B-1.

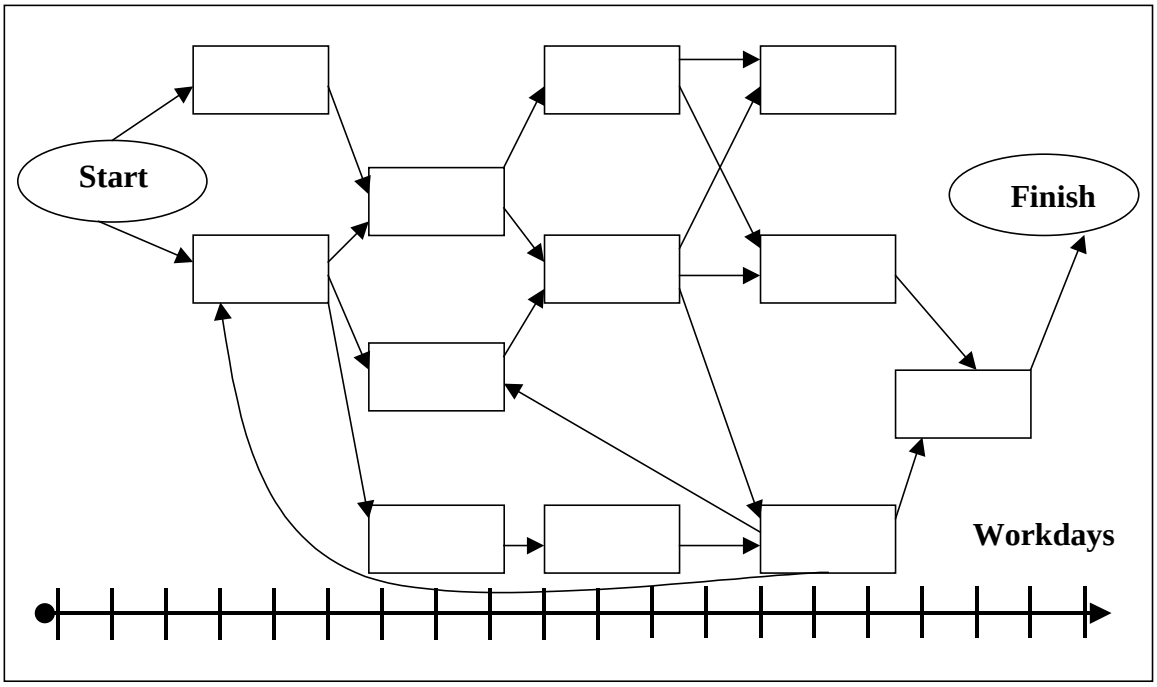

Figure B-1. Activity Network with Backtracking

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 cycle time reduction.

With respect to the model in Figure B-1, the RAD Opportunity Tree of sources of cycle time reduction is presented in Figure B-2. Each major source is elaborated below.

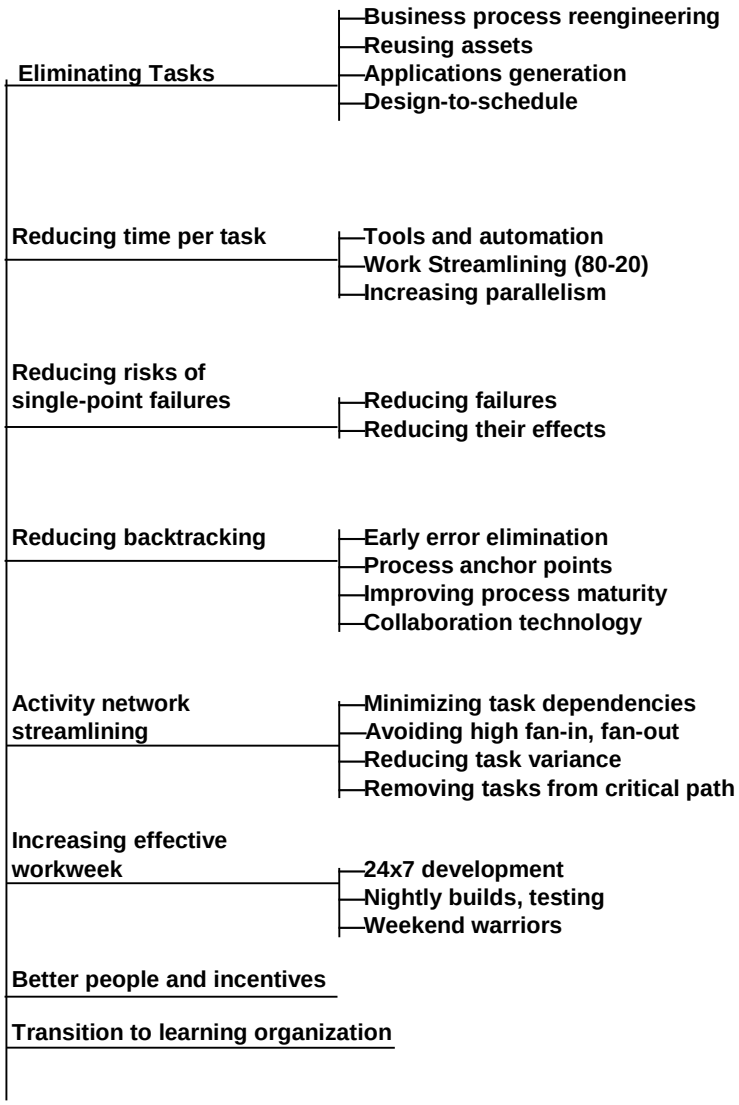

Figure B-2. RAD Opportunity Tree

1.0 Eliminating tasks

Business process re-engineering can discover and eliminate non value-adding tasks, such as unnecessary purchase approvals or change control boards operating at too low a level. Reusing assets and automated applications generation can eliminate many tasks, but they require up-front investment in domain architecting and product line infrastructure. Design-to-schedule can also be highly effective, but again requires up-front investment in prioritizing features and architecting so that features can be dropped without ripple effects.

2.0 Reducing time per task

Reducing time per task can be addressed through technology or management. Tools and automation may be employed when existing tasks lack appropriate technology support, where such supporting mechanisms (e.g., interactive development environments, automated testing packages) are commercially available. When tasks involve getting people together to review or agree on system features or capabilities, but the people involved are not co-located, then the use of collaboration technology, groupware or wide-area workflow management tools may be helpful. 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%. One way to increase effective parallelism in developing a number of components is to ensure precise, well-validated component interface specifications. Then, the development effort for each component can proceed in parallel with minimal delays due to interface reconciliation or cross-component ripple effects.

3.0 Avoiding single-point task failures

System development projects are sometimes prone to single point failures, which can negatively impact completion schedules. For example, hardware platforms or components 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 mitigate single point failures is to provide some form of back-up or parallel capability. However, what is key is detecting or anticipating where these point failures may occur, and taking preventive measures, such as scheduling back-up hardware or providing an apprentice or "stand-in" to key project personnel. Hardware design and software inspections, primarily considered as a defect-detection activity, are also excellent for spreading key product knowledge across the project team.

4.0 Reducing backtracking

Rework is perhaps the most common form of time-sink that system development projects experience. Generally, rework does not add value. Thus, the challenge is how to minimize its occurrence. If rework is due to errors of various kinds, it can be reduced by developing and employing a matrix/taxonomy of errors and corresponding repair actions. While repairs may seem obvious to task experts, to task novices they can be of profound time-saving benefit. Even more effective, however, are techniques for error avoidance and prevention, and early error elimination when back tracking is easier. Process anchor points provide a management framework to help determine process goals, objectives (milestones), and progress measures. Process anchor points and baselining help to establish attainable progress markers and overall project development "velocity." In turn, project velocity should increase as development processes mature, stabilize, and get reused. Such maturity happens most rapidly through top management commitment and resource investment to make it happen. Finally, rework can be reduced by tightening convergence loops or by articulating where progress disconnects occur. For example, when design reviewers review designs outside the presence of the designers, then some record of their discussions and understanding must be prepared, conveyed, and re-explained to the designers. Instead, it is far more efficient to co-locate design reviewers and designers together, or employ collaboration technology to help capture and fill-in the gap between the reviewers and designers.

5.0 Activity network streamlining

As displayed in Figure B-1, 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 projects cross organizational boundaries, then project activities should preferably do so off the critical path. When activity networks get too "bushy" (when certain activities have a high number of input or output paths), then bottlenecks can occur. Decompose and spread out these high fan-in-fan-out nodes, and increase parallelism. Last, as the critical path determines the shortest route to project completion, then look for ways to get time-consuming tasks off the critical path. This may be possible through task decomposition and parallelization, or through network reconfiguration. Some strategies, such as pre-positioning facilities, components, tools, experts, or data, may add somewhat to the cost but be worth it in schedule savings. A good example is "overinvestment" in reusable components. Typical component reuse ROI models conclude that you should expect to use a component at least 3 times to achieve a net payoff, but a lower expected number of uses is appropriate if schedule is more important than cost.

6.0 Increasing duration or number of workdays

Getting project staff to work harder is seldom a viable project management strategy. However, it is all too frequently employed. Staff burnout and untimely turnover can result, which in turn can slow progress and project completion. If non-critical development tasks can be outsourced to others (e.g., off-shore providers, or other corporate divisions in global locations), then so-called 24X7 or round-the-clock, round-the-globe development efforts may be possible. However, this usually requires some amount of up-front investment in creating a shared product vision, establishing the ground rules for inter-firm collaboration, and ensuring consistent technical decision-making in order to succeed. Similarly, swing-shift workers or swing-shift automation mechanisms may be employed, ala Microsoft's nightly builds, developer-tester buddy system, and continuous automated testing. Finally, second or third shift developers, or "weekend warriors" can be employed during project surge or crunch periods, but with varying results and quality outcomes.

7.0 Better people or incentives

Better people usually can get a development effort done with less extraneous effort. However, everyone wants to hire the "best people" and most developers see themselves among the best, when in fact they aren't. When you have to go with "best available" rather than "best," the objective is to establish the sustained means for how to get the most from the people on the job. Motivation is key, but motivation backed by personal commitment and job/career incentives are most effective. This is a proven project management technique that is often not employed, since many software project managers lack management education and experience. The biggest payoffs from incentivizing and getting the best people come because they will be best at selecting and articulating how to employ the other cycle time reduction techniques noted above.

8.0 Transition to learning organizations

The "sixth" level of software process maturity is the transition to a learning organization. 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. Learning organizations are adaptive. They are less concerned about whether they should adopt some new improved technology, simply because it's new or it's supposed to be better. Learning organizations have the resources, staff, and slack to be able to perform and master new development methods. They need not lock themselves into a single tool, technique, paradigm, or fad, since they maximize productivity and minimize cycle time as the normal mode of work.

Appendix C: References

(Al Said 2003). Al Said, M., "Detecting Model Clashes During Software Systems Development," PhD Dissertation, Department of Computer Science, University of Southern California, December 2003.

(Alderson-Doyle 2010). Alderson, D, and J C Doyle. 2010. Contrasting Views of Complexity and Their Implications For Network-Centric Infrastructures. *IEEE TRANSACTIONS ON SYSTEMS MAN AND CYBERNETICS PART A SYSTEMS AND HUMANS* 40, no. 4: 839-852.

(Alexander 1977). Alexander, Christopher. 1977. *A pattern language : towns, buildings, construction*. New York: Oxford University Press.

(Ambrust et al, 2009), Ambrust, A et al. (2009) Above the Clouds: A Berkeley View of Cloud Computing. Technical Report UCB/EECS-2009-28.

(Anon 2007) Anon. Spectrum Auction Pits Google's Game Theorists Against the FCC's | Epicenter | Wired.com. http://www.wired.com/epicenter/2007/11/so-two-game-the/.

(Babar et al. 2010), Babar, M.A., Chen, Lianping, Shull, F. "Managing Variability in Software Product Lines," IEEE Software, published by the IEEE Computer Society, May/June 2010, pp. 89-94.

(Baldwin-Clark 2000). Baldwin, C., and Clark, K., Design Rules: The Power of Modularity, MIT Press, 2000.

(Blanchard-Fabrycky 1998). Blanchard, B., and Fabrycky, W., Systems Engineering and Analysis, Prentice Hall, 1998.

(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, pp. 4-9.

(Boehm et al. 2010). Boehm, B., Lane, J., Koolmanojwong, S., and Turner, R., "Architected Agile Solutions for Software-Reliant Systems," Proceedings, INCOSE 2010.

(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-Lane 2010). Boehm, B., and Lane, J., "Better Management of Program Risks: Early Feasibility Evidence," INCOSE INSIGHT, July 2010, pp. 11-19 (also Proceedings, CSER 2009).

(Boehm-Scherlis, 1992). Boehm, B., and Scherlis, W., Megaprogramming, Proceedings , DARPA Software Technology Conference, April 1992.

(Boehm-Valerdi-Honour 2008). Boehm, B., Valerdi, R., and Honour, E., "The ROI of Systems Engineering: Some Quantitative Results for Software-Intensive Systems," Systems Engineering 11 (3), Fall 2008, pp. 221-234.

(Bongard et al. 2006). Bongard, Josh, Victor Bongard, and Hod Lipson. 2006. Resilient Machines Through Continuous Self-Modeling. *Science.* 314, no. 5802: 1118.

(Booch 2007) Booch, G. Architectural Patterns for a Competitive Advantage, 2007.

(Bosch 2010), Bosch, J., "Toward Compositional Software Product Lines," IEEE Software, published by the IEEE Computer Society, May/June 2010, pp. 29-34.

(Carlini 2009). Carlini, J. "Technology Tools for Rapid Capability Fielding," DDR&E Tools Briefing. 15 December 2009.

(Carlson-Doyle 2002). Carlson JM, and Doyle J. 2002. Complexity and robustness. *Proceedings of the National Academy of Sciences of the United States of America* 99: 2538-45.

(Chandra et al. 2009). Chandra, F, G Buzi, and J.C. Doyle. 2009. Linear control analysis of the autocatalytic glycolysis system. *Proceedings of the American Control Conference*: 319-324.

(Chrissis 2007). Chrissis, M., Konrad, M., and Shrum, S., CMMI (2nd ed.), Addison Wesley, 2007.

(Clements-Northrop 2003). Clements, P. and Northrop, L. *Software Product Lines: Practices and Patterns*, Addison Wesley, 2003.

(Cloutier et al. 2010a). Cloutier, R., Muller, G., Verma, D., Nilchiani, R., Hole,. E., Bone, M. "The Concept of Reference Architectures." INCOSE SE Journal 13.1 (2010): (14-27).

(Cloutier et al. 2010b). Cloutier, R., Mostashari, A., McComb, S., Deshmukh, A., Wade, J., Kennedy, D., Korfiatis, P. "Investigation of a Graphical CONOPS Development Environment for Agile Systems Engineering." Final Technical Report SERC-2009-TR-003. October 2009.

(CommCrit 2009). Common Criteria Recognition Agreement, Common Criteria for Information Technology Security Evaluation Version 3.1. 2009.

(Conway-Mead 1979). Conway and Mead. Introduction to VLSI Systems. Addison-Wesley, 1979.

(Crutchfield 1994). Crutchfield, J. 1994. The calculi of emergence: computation, dynamics and induction. *Physica D. Nonlinear phenomena.* 75, no. 1/3: 11.

(Csete-Doyle 2002). Csete, M, and J C Doyle. 2002. SYSTEMS BIOLOGY - REVIEWS - Reverse Engineering of Biological Complexity. *Science.* 295, no. 5560: 1664.

(Csete-Doyle 2004). Csete, M, and Doyle J. 2004. Bow ties, metabolism and disease. *TRENDS IN BIOTECHNOLOGY* 22, no. 9: 446-450.

(DOD 2008). Department of Defense, Critical Program Information (CPI) Protection. 2008, DoD Instruction (DoDI) 5200.39.

(DoD 2009). The Under Secretary of Defense for Acquisition, Technology, and Logistics and The Assistant Secretary of Defense for Networks and Information Integration/ DoD Chief Information Officer', Report on Trusted Defense Systems. 2009.

(Dorner 1996).Dörner, D. The logic of failure : recognizing and avoiding error in complex situations. Cambridge Mass.: Perseus Books, 1996.

(DSB 2007). Defense Science Board, Mission Impact of Foreign Influence on DoD Software, September 2007.

(Feldman et al. 2008). Feldman, D.P., C.S. McTague, and J.P. Crutchfield. 2008. The organization of intrinsic computation: Complexity-entropy diagrams and the diversity of natural information processing. *Chaos* 18, no. 4.

(Gamma 1995). Gamma, Erich et al. 1995. *Design patterns : elements of reusable object-oriented software*. Reading Mass.: Addison-Wesley.

(GAO 2008). U.S. General Accountability Office, "Defense Acquisitions: Assessments of Selected Major Weapon Programs," March 2008.

(Goeree et al. 2006). Goeree, J. K., C. A. Holt, and J. O. Ledyard, "An Experimental Comparison of the FCC's Combinatorial and Non-Combinatorial Simultaneous Multiple Round Auctions." Prepared for the Wireless Communications Bureau of the Federal Communications Commission, 2006.

(Herzon 2010). Herzog, P., Open Source Security Testing Methodology Manual (OSSTMM) 2010, ISECOM.

(Holstein 2010). Holstein, D.K. and K. Stouffer, Trust but Verify Critical Infrastructure Cyber Security Solutions, 43rd Hawaii International Conference on System Sciences (HICSS). 2010 IEEE. p. 1–8.

(Hopkins-Jenkins 2008). Hopkins, R., and Jenkins, K., *Eating the IT Elephant: Moving from Greenfield Development to Brownfield*, IBM Press, 2008.

(IMTI 2010). Ten Imperatives for Model-Based Enterprise." IMTI Update. Winter 2010 Edition. www.imti21.org/newsletter/winter2010/model.html

(InfoWeek 2010). http://www.informationweek.com/news/government/cloud-saas/showArticle.jhtml?articleID=225200398&queryText=cloud%20computing Accessed on June 25, 2010.

(Ingber 2008). Ingber, Donald. 2008. From Molecular Cell Engineering to Biologically Inspired Engineering. *Cellular and Molecular Bioengineering* 1, no. 1 (March 1): 51-57.

(Irvine-Levitt 2007). Irvine, C.E. and K. Levitt, Trusted Hardware: Can It Be Trustworthy?, in Design Automation Conference. 2007, Association of Computing Machinery: San Diego, California, USA.

(ISO 2009). International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC), Systems and software engineering — Systems and software assurance — Part 2: Assurance case (ISO/IEC 15026). 2009.

(Jiang-Baras 2004). Jiang, T. and J.S. Baras, Ant-based Adaptive Trust Evidence Distribution in MANET, in Proceedings of the 24th International Conference on Distributed Computing Systems Workshops (ICDCSW'04) 2004 IEEE.

(Johnson 2004). Johnson, B.C., National Security Agency (NSA) INFOSEC Assessment Methodology (IAM). 2004, SystemExperts Corporation.

(Johnson 2006). Johnson, J., My Life Is Failure, The Standish Group, 2006.

(Jones and Northrop 2010). Jones, L.G. and Northrop, L.M., "Clearing the Way for Software Product Line Success," IEEE Software, published by the IEEE Computer Society, May/June 2010, pp. 22-28.

(Jordan et al. 2010). Jordan, Patrick R., L. Julian Schvartzman, and Michael P. Wellman. 2010. Strategy Exploration in Empirical Games. *Proc. of 9th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2010)* (May 10): 1131-1138.

(Kauffman 2000). Kauffman, Stuart. 2000. *Investigations*. Oxford ;;New York: Oxford University Press.

(Kephart-Chess 2003). Kephart, Jeffrey, and David M Chess. 2003. The Vision of Autonomic Computing - Systems manage themselves according to an administrator's goals. New components integrate as effortlessly as a new cell establishes itself in the human body. These. *Computer.* 36, no. 1: 41.

(Kim-Love-Spafford 2008). Kim, G., P. Love, and G. Spafford, Visible Ops Security. 2008: Information Technology Process Institute.

(Lane et al. 2010). Lane, J., Boehm, B., Bolas, M., Madni, A., and Turner, R., "Critical Success Factors for Rapid, Innovative Solutions," Proceedings, ICSP 2010.

(Lee-Koyonya 2010). Lee, J. and Kotonya, G., "Combining Service Orientation with Product Line Engineering," IEEE Software, published by the IEEE Computer Society, May/June 2010, pp. 35-41.

(Lewis et al. 2008). Lewis, G. et al., "SMART: Analyzing the Reuse Potential of Legacy Components on a Service-Oriented Architecture Environment," CMU/SEI-2008-TN-008, 2008.

(Lim 1998). Lim, W., Managing Software Reuse, Prentice Hall, 1998.

(Lucero 2009). Lucero, S., "Software Sustainment Challenges in Defense Acquisition," Proceedings, AIAA Engineering Conference, April 2009.

(Madni 2008a) Madni, A.M. "Agile Systems Architecting: Placing Agility Where it Counts," *Conference on Systems Engineering Research (CSER)*, 2008.

(Madni 2008b). Madni, A.M. "AgileTecting™: A Principled Approach to Introducing Agility in Systems Engineering and Product Development Enterprises," Journal of Integrated Design and Process Science, Vol. 12, No. 4, December 2008.

(Madni 2010a).Madni, A.M. "Integrating Humans with Software and Systems: Technical Challenges and a Research Agenda," *INCOSE Journal of Systems Engineering*, Vol. 13, No. 3, 2010.

(Madni 2010a).Madni, A.M. "Integrating Humans with Software and Systems: Technical Challenges and a Research Agenda," *INCOSE Journal of Systems Engineering*, Vol. 13, No. 3, 2010.

(Madni 2010b). Madni, A.M. Agility Platforms, USC Report on Platform-Based Engineering for Systems 2020, June 2010.

(Madni-Jackson, 2008).Madni, A.M., and Jackson, S. "Towards a Conceptual Framework for Resilience Engineering," *IEEE Systems Journal, Special issue on Resilience Engineering*, Paper No. 132, 2008.

(Madni-Moini 2007). Madni, A.M., and Moini, A. "Viewing Enterprises as Systems-of-Systems (SoS): Implications for SoS Research," *Journal of Integrated Design and Process Science*, Vol. 11, No. 2, June 2007, pp. 3-14.

(Maier 2007). Maier, M., "Views of System Complexity," Rand Symposium on System Complexity, January 2007.

(Maranzano 2005). Maranzano, J., et al., "Architecture Reviews: Practice and Experience," IEEE Software, March/April 2005.

(Margolis 1987).Margolis, H., Patterns, thinking, and cognition : a theory of judgment. Chicago: University of Chicago Press, 1987.

(Marks-Bell 2006). E. Marks and M. Bell, Service Oriented Architecture, Wiley, 2006.

(McGregor et al. 2010). McGreger, J.D., Muthig, D., Yoshimura, K., Jensen, P. "Successful Software Product Line Practices," *IEEE Software*, published by the IEEE Computer Society, May/June 2010, pp. 16-21.

(Mohan et al. 2010). Mohan, K., Balasubramaniam, R., Sugumaran, V. "Integrating Software Product Line Engineering and Agile Development," IEEE Software, published by the IEEE Computer Society, May/June 2010, pp. 48-55.

(Moritz et al. 2005). Moritz, Max A, Marco E Morais, Lora A Summerell, J M Carlson, and John Doyle. 2005. Wildfires, complexity, and highly optimized tolerance. *Proceedings of the National Academy of Sciences of the United States of America.* 102, no. 50: 17912.

(Moses 2010). Moses, J. "Architecting Engineering Systems," in Philosophy and Engineering, Springer Science, Chapter 23, pp. 275-284, 2010.

(Nash 1950). Nash, J. "Equilibrium points in n-person games" *Proceedings of the National Academy of Sciences* 36(1):48-49.

(Nilchiani 2007). Nilchiani, R., Measuring Space Systems Flexibility: A Comprehensive Six-element Framework. Systems Engineering, 2007. 10(1): p. 305.

(NIST 2010).http://csrc.nist.gov/groups/SNS/cloud-computing/ Accessed on June 24, 2010.

(NRC 1979). National Research Council. Retooling Manufacturing. National Academy Press, 2004.

(OUSD(AT&L) 2008). OUSD(AT&L), Systems Engineering Guide for Systems of Systems, Version 1.0, June 2008.

(Parnas 1979). Parnas, D, "Designing Software for Ease of Extension and Contraction," IEEE Trans. SW Engr., March 1979, pp. 128-137.

(Ratcliff 2009). Ratcliff, Adele. "Rapid Capabilities Toolbox Study." Presentation to DDR&E. 3 Sept. 2009.

(Ross 2007) Ross, R., et al., Recommended Security Controls for Federal Information Systems, SP 800-53 Rev 2, National Institute of Standards and Technology, Editor. 2007.

(Salasin-Madni 2007).Salasin, J., and Madni, A.M. "Metrics for Service Oriented Architecture (SOA) Systems: What Developers Should Know," *Journal of Integrated Design and Process Science*, Vol. 11, No. 2, June 2007, pp. 55-71.

(Sangiovanni-Vincentelli 2009). Sangiovanni-Vincentelli, Alberto. "Managing Complexity in IC Design." Presentation to DARPA. NSF Complexity Workshop, 2009.

(SEI-CMU 2006). Software Engineering Institute, Carnegie Mellon U., Ultra-Large-Scale Systems, Pittsburgh, PA, June 2006

(Shalizi et al. 2004) Shalizi, Cosma, Kristina Shalizi, and Robert Haslinger. 2004. Soft Matter, Biological, and Interdisciplinary Physics - Quantifying Self-Organization with Optimal Predictors. *Physical review letters.* 93, no. 11: 118701.

(Singh et al. 2009). Singh, l, R. L. Lewis, and A. G. Barto. 2009. Where Do Rewards Come From? *Proceedings of the 31st Annual Conference of the Cogniive Science Society*: 2601-2606.

(Swanson 2001). Swanson, M., Security Self-Assessment Guide for Information Technology Systems, National Institute of Standards and Technology, Editor. 2001.

(Swartout 2010). Swartout, W., "Lessons Learned from Virtual Humans," AI Magazine 31 (1), 2010.

(Tanaka 2005). Tanaka, Reiko. 2005. Soft Matter, Biological, and Interdisciplinary Physics - Scale-Rich Metabolic Networks. *Physical review letters.* 94, no. 16: 168101.

(Tanaka et al. 2005). Tanaka, R, M Csete, and J Doyle. 2005. Highly optimised global organisation of metabolic networks. *Systems biology* 152, no. 4 (December): 179-84.

(Trimberger 2007). Trimberger, S., Trusted Design in FPGAs, in Design Automation Conference. 2007, Association of Computing Machinery: San Diego, California, USA.

(USAF 2007). U.S. Air Force, "Probability of Program Success (PoPS) Spreadsheet Operations Guide," Version 9.6, July 2007.

(Version 9 2009). Version 9 Enterprise Edition, The Open Group, February 2009. (NIST 2010) http://csrc.nist.gov/groups/SNS/cloud-computing/ Accessed on June 24, 2010.

(Wade et al 2010a). Wade, J., Heydari, B., Mostashari, A., "Some Simple Thoughts on Complex Systems." Research Colloquium on Complex Systems: Exemplar – Financial Systems, Stevens Institute of Technology, June 24, 2010.

(Wade et al 2010b). Wade, J., Madni, A., 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.

(Whitney 1996). Whitney, Daniel E. "Why Mechanical Design Cannot Be Like VLSI Design." MIT Engineering Systems Division. 4 April 1996. http://esd.mit.edu/esd_books/whitney/whitney_online.html

(Willinger et al. 2009). Willinger, Walter, David Alderson, and John C Doyle. 2009. Mathematics and the Internet: A Source of Enormous Confusion and Great Potential - Graph theory models the Internet mathematically, and a number of plausible mathematically intersecting network models. *Notices of the American Mathematical Society.* (May): 586.

(Wolfers-Kitsewitz 2004). Wolfers, Justin, and Eric Kitzewitz. 2004. *Prediction markets*. Cambridge Mass.: National Bureau of Economic Research.

(Wolpert 2006). Wolpert, D. 2006. An adaptive Metropolis-Hastings scheme: Sampling and optimization. *Europhysics letters.* 76, no. 3: 353.

(Wolpert-Macready 2005). Wolpert, D, and W G Macready. 2005. Coevolutionary Free Lunches. *IEEE transactions on evolutionary computation : a publication of the IEEE Neural Networks Council.* 9, no. 6: 721.

(Yaiche et al. 2000). Yaiche, Haikel, Ravi R Mazumdar, and Catherine Rosenberg. 2000. A game theoretic framework for bandwidth allocation and pricing in broadband networks. *IEEE/ACM TRANSACTIONS ON NETWORKING* 8: 667--678.

(Yang et al. 2007).Yang, Z, et al. "Building a Semantic-Rich Service-Oriented Manufacturing Environment." International Journal of Information Technology and Web Engineering. Volume 2:3, 2007.