A Research and Transition Roadmap for Producibility Technology

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Grady Campbell <ghc@sei.cmu.edu>
22 April 2009

Context: DoD Acquisition System Needs

- Defense Science Board studies on software problems
- Workshops of Networking and Information Technology Research and Development (NITRD) software coordinating groups
- National Defense Industry Association top problems in software
- Analyses and evaluations of acquisition programs
- Surveys of acquisition, systems, and software practitioners
- Office Secretary of Defense (OSD) Workshops on Producibility with industry/academia

A Producibility Gap

- Projected large exponential growth in needed software capabilities
- Observed modest (5%) annual improvement in production capability (D. Reifer)

Context: Questions to Consider

How well do current approaches to software development scale up to solving today's problems?

Are there better approaches possible? How would software development work if it worked perfectly?

What advances in technology would improve our ability to produce software-intensive systems?

Symptomatic DoD Producibility Problems

Requirements inadequately defining the problem and over-specifying a solution

Difficulties accommodating changing requirements or technology

Testing consuming inordinate resources to find product flaws late

Projects ineffectively balancing function, quality, and cost/schedule

Decisions made to achieve first delivery increasing lifecycle costs

System designs failing to explore software alternatives and tradeoffs

Software not being engineered for system properties but being “fixed” through trial-and-error

Development tools failing to improve cost, time, or product quality

Acquisition focus on products versus investing in establishing domain-specific problem-solution expertise as a capital asset

Producibility

The ability to deliver needed capability in a timely, cost-effective, and predictable manner

Context: Software-intensive Systems (SiS)

Dimensions:

- Developer productivity (efficiency and effectiveness)
- Product value (utility and quality)
- Acquirer acuity (insight and foresight)

Producibility technology means tools and methods used by SiS developers to build SiS capabilities

Charter for an SiS Producibility Initiative (SiSPI)

Objective: Foster the development and adoption of technology that enhances producibility for software-intensive systems

Strategy (for Governance)

- Define an evolving vision-based roadmap for goal-oriented research and transition of technology
- Select and manage efforts to balance near-term incremental improvements with long-term systemic advancement
- Measure progress based on provider targets and acquisition program metrics

Research

- Charter coordinated research and development of producibility-enhancing software/systems engineering technologies

Transition

- Actively facilitate transition of technology from research into practice on SiS acquisition programs

A Reference Vision for Producibility

Computer-Aided Design and Manufacturing (CAD/CAM) for Software-intensive Systems

Model-centric – All problem/solution information is represented in a comprehensive multi-faceted product model

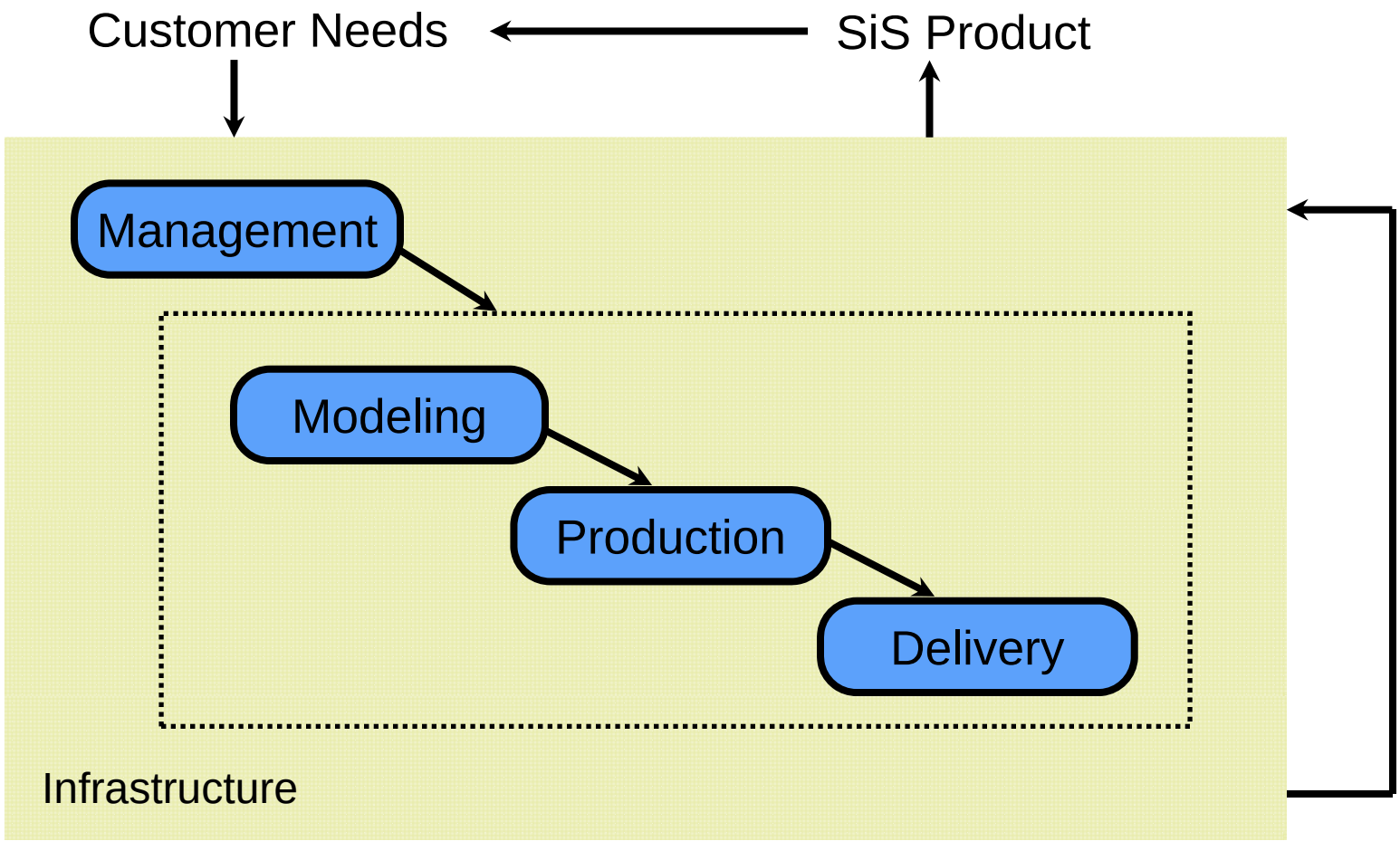
Virtualized – A system is defined by building, pre-deploying, and validating it in software within a hardware/software virtual environment

Predictable – Software and dependent system properties of interest are able to be accurately predicted and mutually optimized

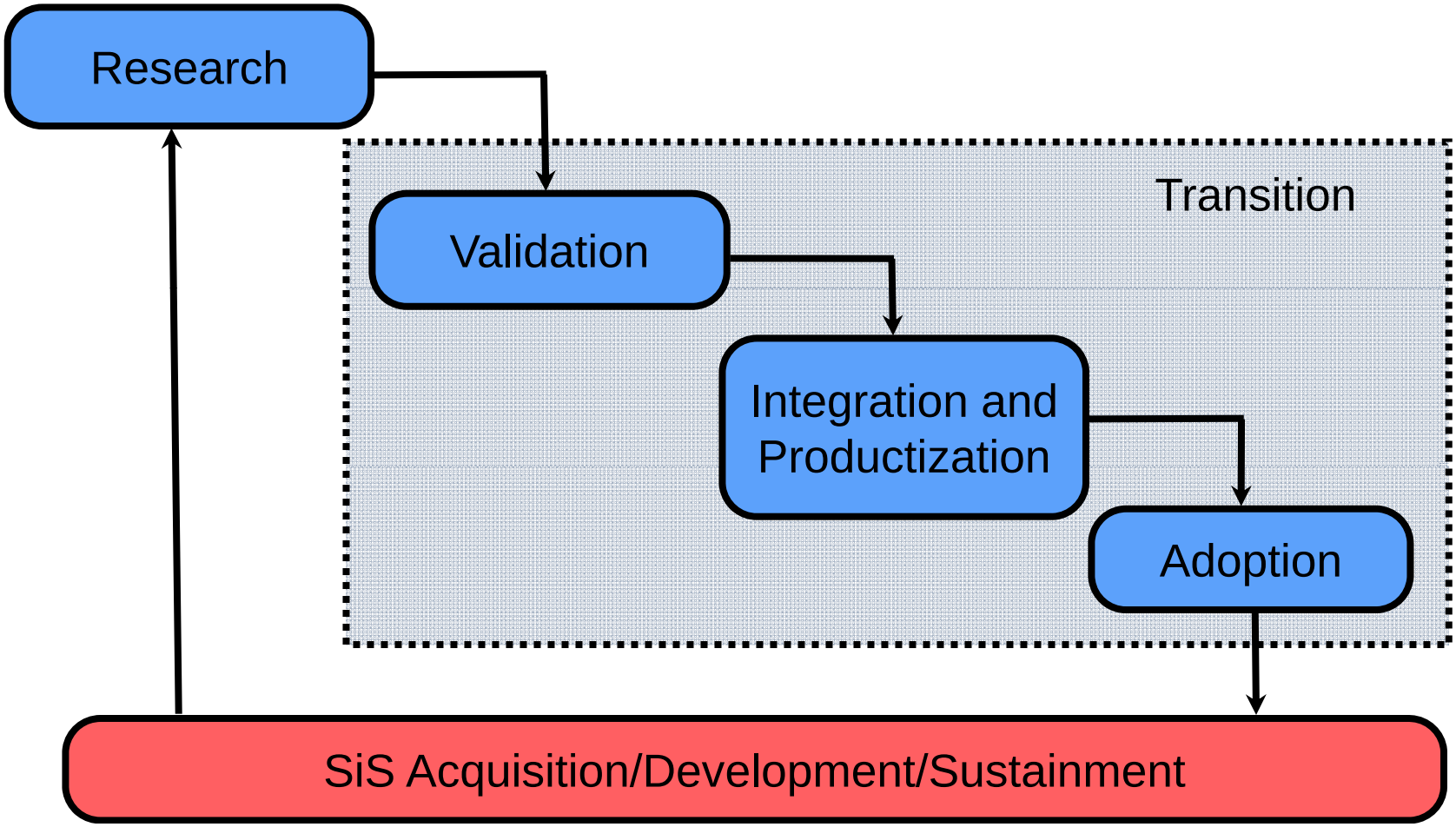
Decision-focused – Multiple alternative solutions can be modeled, produced, and empirically evaluated based on identified customer and engineering choices

Evolvable – The problem/solution model can be continuously evolved to create product variants that meet anticipated differing or changing needs

A Notional Producibility-Based SiS Process



A Technology Advancement Lifecycle



Measuring Progress

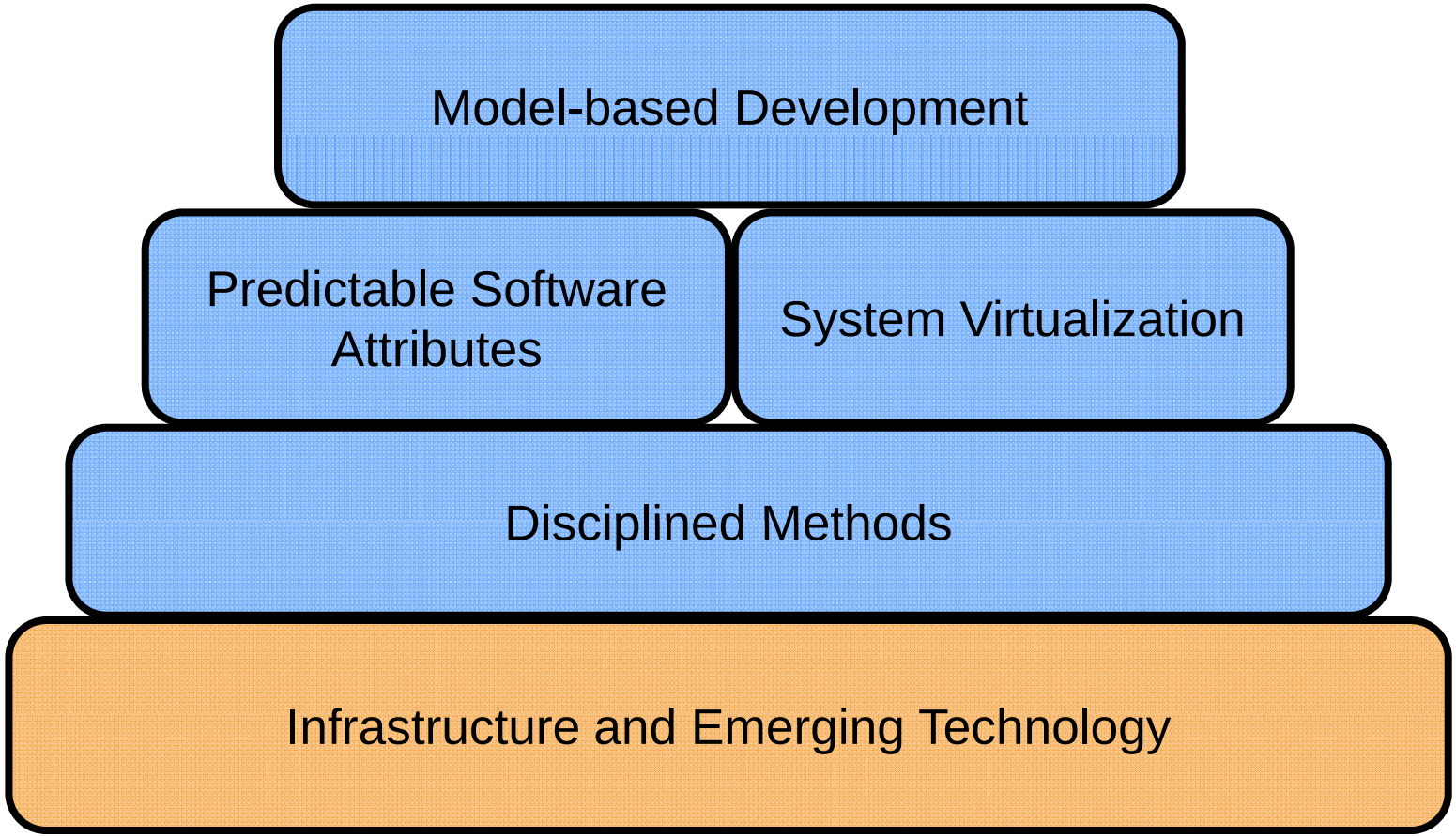
Near-term: Does a proposed technology have the expected benefit within its scope of application in an acquisition?

Medium-term: Are individual acquisition programs seeing benefits expected for SiSPI technology adopted?

Long-term: Is the efficiency and effectiveness of the DoD acquisition-sustainment system being improved as a result of SiSPI efforts?

What metrics will answer these questions?

Themes for Producibility Research - 1



Themes for Producibility Research - 2

Model-based development

- Bridging the conceptual gap between customers and product developers to rapidly formulate, build, and evaluate alternative solutions to evolving needs

Predictable software attributes

- Measuring, predicting, and controlling SiS software properties and tradeoffs

System virtualization

- Creating virtualized environments for controlled pre-verifying of SiS real-world behaviors

Disciplined methods

- Applying effective methods for engineering discipline in the interdependent development of software within systems

Infrastructure and emerging technology

- Exploiting and accommodating advances in infrastructure and enabling computing technologies

Transition Theme: Validation

Demonstrate the applicability and practical value of research results for building software-intensive systems

- Operate an open framework for researchers to collaboratively advance experimental tools and methods
- Evaluate technology with respect to DoD systems producibility
- Facilitate integration among emerging and productized tools and methods
- Publish case studies that illustrate the techniques, value, and challenges of applying Producibility improvements

Transition Theme: Integration and Productization

Engineer research results into integrated engineering tools and methods suitable for production use

- Coordinate with tool/method suppliers on acquisition community needs
- Demonstrate use of validated technology in DoD production environments
- Create/enhance commercial or open source products with capabilities of demonstrated tools and methods
- Establish education, training, and support in the use of demonstrated tools and methods

Transition Theme: Adoption

Facilitate the adoption of productized Producibility technologies by acquisition programs

- Establish Producibility-motivated Acquisition policies, guidance/training, and practices
- Reward suppliers for introducing life-cycle measurable Producibility improvements
- Charter DoD service-level agents to work with targeted programs to improve Producibility across the Acquisition-Sustainment Life Cycle

Roadmap Status and Plans

Software-intensive Systems Producibility: A Vision and Roadmap (V 0.1) (CMU/SEI-2007-TN-017, December 2007)

Possible future activities:

- Iteratively refine and elaborate goals for potential research and transition efforts
- Better characterize current state of practice and identify feasible transitional paths to the Producibility vision
- Formulate feasible near-term advances (Producibility objectives -> impediments -> improvement opportunities)
- Explore how to measure Producibility as experienced during acquisitions and sustainment

Prerequisite: Government and/or industry funding for Producibility

Supplemental Slides

Dimensions of Producibility

Developer productivity (efficiency and effectiveness)

- Domain knowledge and expertise, effective methods, multi-discipline integration
- Engineering discipline, process capability, systems-software engineering synergy
- Technology base (applicability, effort reduction)
- Leveragable resources (legacy, COTS, open source, domain-specific)
- Addressing uncertainty, diversity, and change

Product value (utility and quality)

- Functionality cost-effectively responsive to business/mission needs
- Quality attributes as determinants of system properties
- Compatibility with system and operational environment

Acquirer acuity (insight and foresight)

- Producibility-enabling acquisition policies and practices
- Effective technical direction, oversight, and feedback
- Mechanisms for capability-cost-schedule predictability and tradeoffs
- Infrastructure for technology development, evaluation, transition-into-use, and evolution

Research Theme: Model-based Development

Model: A representation of a product that enables approximate answers to a designated set of questions about the product

Representation: what problem-solution information is needed to define a system, what purposes should it serve, and how should it be represented?

Problem analysis and specification: can the problem-solution space be abstracted into domain-specific representations that will reduce the development process to an iteratively converging decision process?

Solution analysis and validation: what capabilities are needed to provide rapid visualization and empirical resolution of alternative solutions for a specified problem?

Product generation: What mechanisms will enable rapid correct generation of customized software, documentation, and support materials from a model?

Model-product verification: what capabilities are needed to ensure that a derived solution product correctly implements a model?

Research Theme:
Predictable Software Attributes

Design => Analysis of Alternatives and Risk Mitigation

Identification: What are critical software-affected attributes and how do they interact? (Performance, reliability, availability, security, safety, usability, …?)

Measurement: What behavior does the system exhibit in terms of critical attributes and how is this determined by implemented software?

Prediction: Given a proposed software design and implementation, how will critical attributes be affected?

Optimization: What requirements/design/implementation decisions lead to the best combination of critical attribute values?

Research Theme:
System Virtualization

Platform independence: What form should implementations take to permit alternative physical realizations while avoiding unnecessary dependence on any specific realization?

Hardware abstraction: How is hardware represented to enable simulated use for verifying software and supporting hardware/software codesign?

Environment simulation: How can capabilities and constraints of the operational environment (systems, devices) be represented to enable simulated use for software validation?

Usage simulation: How are potential uses of a system adequately represented to enable realistic automated testing?

System validation: What techniques enable validation of a solution under realistic (normal and degraded) conditions?

Research Theme: Disciplined Methods

Management: How is iterative, concurrent, multi-version development planned, monitored, and controlled to ensure meeting schedule/budget/quality goals?

Requirements: How do developers accurately and concisely represent the capabilities and limitations of a system/software being produced?

Architectural design: What forms are sufficient to define the structure and composition of software in a system, as a basis for achieving tradeoffs?

Component design: What information does an implementer need to be provided in order to build, use, or safely modify a software component?

Implementation: How can implementation practices be improved to eliminate defects and reduce rework due to requirements or design changes?

Verification: How can inconsistencies be precluded among multiple changing software representations (code, models/specifications, documents, tests)?

Product families: How can an envisioned set of similar/evolving products be represented to eliminate redundant development efforts?

Research Theme: Infrastructure & Emerging Technology

How is producibility enhanced or changed due to capabilities of computing infrastructure and emerging technologies such as these?

Computational technology: Multi-core processors; distributed processing, services, and data; autonomous agents; grid computing

Componentization: Packaged pluggable components and frameworks (COTS, legacy, open-source; defined interfaces outward & underneath)

Customization: Total-product variant and configuration management; product family and generator techniques

Commoditization: Standardized cross-domain frameworks for common system capabilities

Cross-speciality collaborative engineering: Tools and techniques for communication and coordination

Ideas for Near-term Progress

Define a standardized framework for precisely identifying and measuring critical software-system properties

Create a DoD-wide repository exhibiting large-scale use of effective software methods for requirements specification, architectural and component design, and verification

Reformulate relevant systems and software methods to foster collaborative software-based systems engineering

Advocate Federal/DoD acquisition practices that motivate programs to adopt practices that address common lifecycle software problems

Establish a DoD capability for facilitating the packaging and transition of effective R&D technology into use on acquisition programs

Initiate efforts on programs building multi-version solutions to create a model-driven development capability based on product line principles

Proposed Criteria for Funding Research

Long-Term Payoff: Consistent with and advances attainment of the Producibility vision

Near-Term Payoff: Directly addresses some aspect of the producibility problem as experienced by current DoD programs

Pragmatics: Compatible with current/emerging DoD practices and technology trends and not dependent on other advances that are not timely

Research opportunity: Not otherwise adequately funded relative to DoD needs

Transitionability: Credible plan for productization and transition into use

Transition opportunity: Identified DoD acquisition programs anticipate adequate benefit and agree to evaluate viability

Measuring Progress: Long-Term

Is the efficiency and effectiveness of the DoD acquisition system being improved as a result of SiSPI efforts?

- Measurable improvements in the acquisition system as a whole will require a critical mass of SiSPI improvements across many individual programs.
- Individual programs may require large improvements for an improvement in the system as a whole to be detectable.
- Expected improvement timeframe: 5-20 years
- Example metrics:
  - Annual sustainment cost trends
  - Normalized initial crewmember training cost
  - Software failures per operational year

Measuring Progress: Medium-Term

Do individual acquisition programs show benefits expected of adopted SiSPI technology?

- Individual programs will be selective in adopting SiSPI technologies, limiting scope and impact of risk
- Results from different programs, adopting different technology improvements, may be difficult to compare
- Expected improvement timeframe: 3-5 years
- Example metrics:
  - Ratio of pre- to post-deployment defects discovered
  - Percent of development time spent in testing
  - Time required to respond to requests for system improvements

Measuring Progress: Near-Term

Does a proposed SiSPI technology have the expected benefit within its scope of application in an acquisition?

- The developer of a technology must define what measurable benefit adopters should be able to expect.
- An individual technology improvement may have only a minor effect on an acquisition program as a whole
- Expected improvement timeframe: 1-2 years
- Example metrics:
  - Time spent in activities affected by a technology
  - Effort required to satisfy a key system property