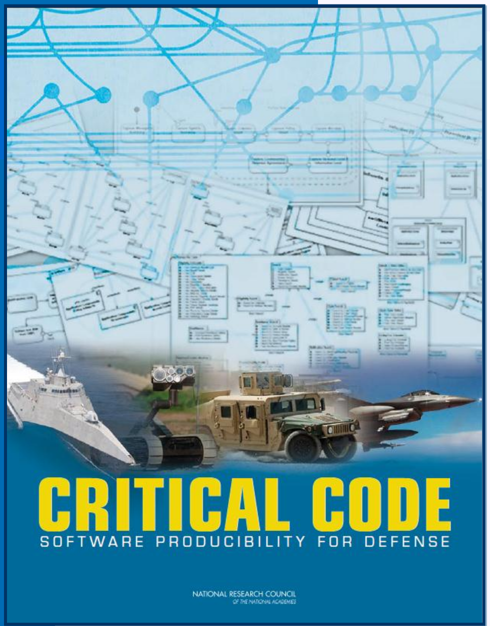
THE NATIONAL ACADEMIES
Advisers to the Nation on Science, Engineering, and Medicine



Critical Code

Software Producibility for Defense

***Summary points** from the final report of the*
Committee on Advancing Software-Intensive Systems Producibility (ASISP)

William Scherlis, *Chair*

Enita Williams, Study Director
Jon Eisenberg, CSTB Director

December 2010

National Research Council (NRC)
Computer Science and Telecommunications Board (CSTB)

v03

One slide summary:
Goals and Enablers

Mission goals ← Practice improvements ← Research

- Improve critical areas of current **practice**
  - **Enable incremental iterative development at arm's length**
    - **Process and measurement**
  - **Enable architecture leadership, interlinking, flexibility**
    - **Architecture**
  - **Enable mission assurance at scale, with rich supply chains**
    - **Assurance and security**
- Undertake **research** to support the critical areas of practice
  1. **Architecture modeling and architectural analysis**
  2. **Validation, verification, and analysis of design and code**
  3. **Process support and economic models for assurance**
  4. **Requirements**
  5. **Language, modeling, code, and tools**
  6. **Cyber-physical systems**
  7. **Human-system interaction**

One slide summary:
Recommendations

Key Findings and Recommendations

1. **Software has become critical in its role and strategic significance for DoD**
   - Software enables capability, integration, and agility in defense systems
   - DoD needs to actively and directly address its software producibility needs
   - NITRD data reveal the extent of the S&T disengagement that must be reversed
2. **Innovative software-intensive engineering can be managed more effectively**
   - Apply advanced practice and supporting tools for iterative incremental development
   - Update earned-value models and practices to support management process
3. **DoD needs to be a smarter software customer**
   - There is insufficient DoD-aligned software expertise within and around DoD
4. **Assert DoD architectural leadership**
   - In highly complex systems with emphasis on quality attributes, architecture decisions may need to dominate functional capability choices
5. **Adopt a strategic approach to software-intensive mission assurance**
   - Integrate preventive practices into development to support ongoing creation of evidence in support of assurance
   - Do not lose leadership in software evaluation and assurance (DSB'07)
6. **Reinvigorate and focus DoD software engineering research**
   - Apply appropriate criteria in identifying goals for research programs
   - Focus research effort on identified goals in seven technical areas

One slide summary:
Assurance

Adopt a strategic approach to software assurance

- Finding from DSB2007, reiterated in Critical Code
  - It is an essential requirement that the United States maintain advanced capability for "test and evaluation" of IT products. Reputation-based or trust-based credentialing of software ("provenance") needs to be augmented by direct, artifact-focused means to support acceptance evaluation.
- Context
  - Challenges
    - Inadequate + costly legacy approaches – based on inspection and sampled tests
    - Newly rich and globally diverse supply chains, with arms-length relationships
    - Assurance reqts can dramatically limit systems capability, and vice-versa
  - Opportunities
    - Significant advances and potential for preventive/evaluative practices
      - *Evidence production* – *Isolation / encapsulation*
      - *Architecture design* – *Configuration management*
    - Potential for new approaches to "evaluation standards" for legacy / ongoing / new
- Conclusion
  - DoD must directly foster advanced software practice and tools for highly assured high capability systems -- nobody is doing this for DoD

Outline

- Task and prior reports
- Committee, process, background
- Areas of practice
  - Process and measurement
  - Software expertise
  - Architecture
  - Assurance and security
- Topics of research
- Economic argument
- Next steps

5

The ASISP committee of the NRC

- National Research Council (NRC) ASISP Committee
  - *ASISP:* Advancing Software-Intensive Systems Producibility
  - *Producibility:* the capacity to design, produce, assure, and evolve software-intensive systems in a predictable manner while effectively managing risk, cost, schedule, quality, and complexity.
- Commissioned by the Office of the Secretary of Defense (OSD)
  - DDR&E focal point, with ONR support and NSF assistance
- NRC charge to committee
  - Assess national investment in relevant software research
  - Recommend improvements to DoD software practice
  - Examine needs relating to DoD software research
  - Assess research requirements relating to software producibility

6

ASISP study committee

- William **Scherlis**, Carnegie Mellon University, *Chair*
- Robert **Behler**, The MITRE Corporation
- Barry W. **Boehm**, University of Southern California
- Lori **Clarke**, University of Massachusetts at Amherst
- Michael **Cusumano**, Massachusetts Institute of Technology
- Mary Ann **Davidson**, Oracle Corporation
- Larry **Druffel**, Software Engineering Institute
- Russell **Frew**, Lockheed Martin
- James **Larus**, Microsoft Corporation
- Greg **Morrisett**, Harvard University
- Walker **Royce**, IBM
- Doug C. **Schmidt**, Vanderbilt University
- John P. **Stenbit**, Independent Consultant
- Kevin J. **Sullivan**, University of Virginia

- ***CSTB Staff***
- Enita **Williams**, Study Director
- Jon **Eisenberg**, CSTB Director
- *Thanks also to*: Joan **Winston**, Lynette **Millett**, Morgan **Motto**, Eric **Whitaker**

– Industry integrators
  - Software vendors
  - Defense primes

– Government
  - Government experience
  - FFDRC advisors

– Research
  - Academia
  - Industry

7

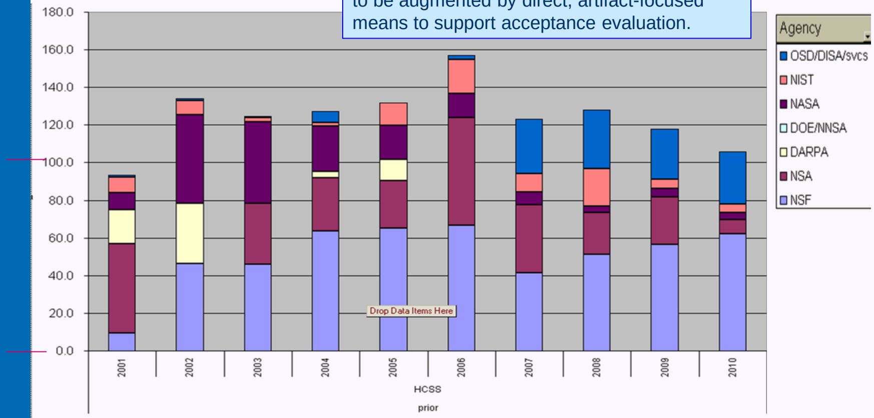
Reviewers of the ASISP reports

- Rick **Buskens**, Lockheed Martin ATL (*final*)
- Grady **Campbell**, Software Engineering Institute (*final*)
- William **Campbell**, BAE Systems (*final*)
- John **Gilligan**, Gilligan Group (*letter, final*)
- William **Griswold**, University of California, San Diego (*final*)
- Anita **Jones**, University of Virginia (*letter, final*)
- Annette **Krygiel**, Independent Consultant (*final*)
- Butler **Lampson**, Microsoft Corporation (*letter*)
- Steve **Lipner**, Microsoft, Inc. (*final*)
- David **Notkin**, University of Washington (*workshop, letter, final*)
- Frank **Perry**, SAIC (*final*)
- William **Press**, U Texas Austin (*final review monitor*)
- Harry D. **Raduege**, Jr., Deloitte Center for Network Innovation (*letter*)
- Alfred Z. **Spector**, Google, Inc. (*workshop, letter, final*)
- Daniel C. **Sturman**, Google, Inc. (*final*)
- John **Swainson**, CA, Inc. (*final*)
- Mark N. **Wegman**, IBM (*final*)
- John **Vu**, Boeing Corporation (*workshop*)
- Peter **Weinberger**, Google, Inc. (*workshop*)
- Jeannette **Wing**, Carnegie Mellon University (*workshop*)

8

HCSS (High Confidence Software and Systems)

***DSB2007*: It is an *essential requirement* that the United States maintain advanced capability for "*test and evaluation*" of IT products.** Reputation-based or trust-based credentialing of software ("provenance") needs to be augmented by direct, artifact-focused means to support acceptance evaluation.



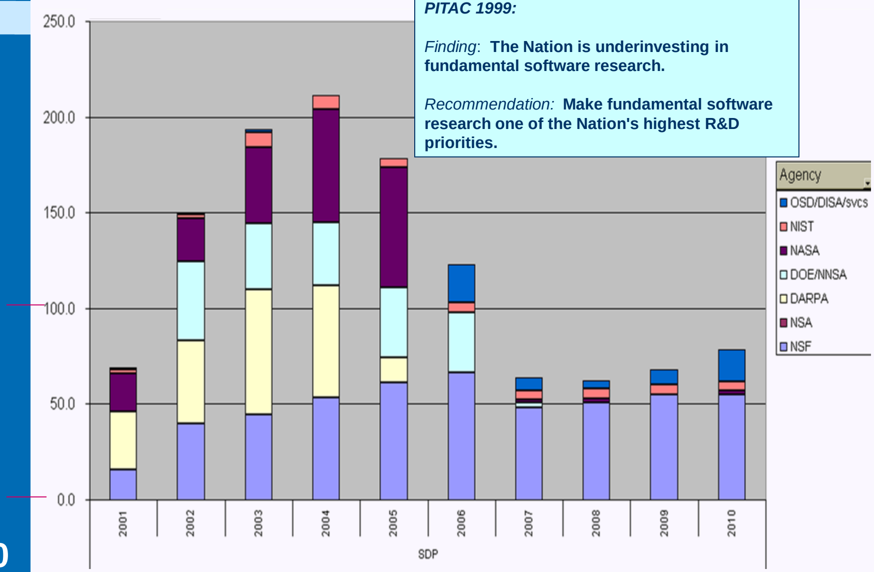
9

SDP (Software Design and Productivity)

PITAC 1999:

Finding: **The Nation is underinvesting in fundamental software research.**

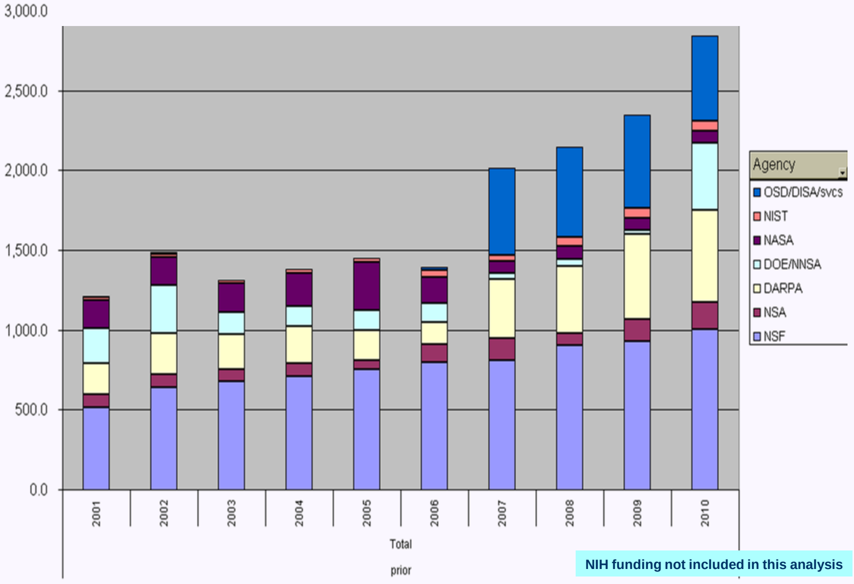
Recommendation: **Make fundamental software research one of the Nation's highest R&D priorities.**



10

11

Total NITRD investment *("prior year" amounts)*



12

SDP+HCSS relative to total NITRD

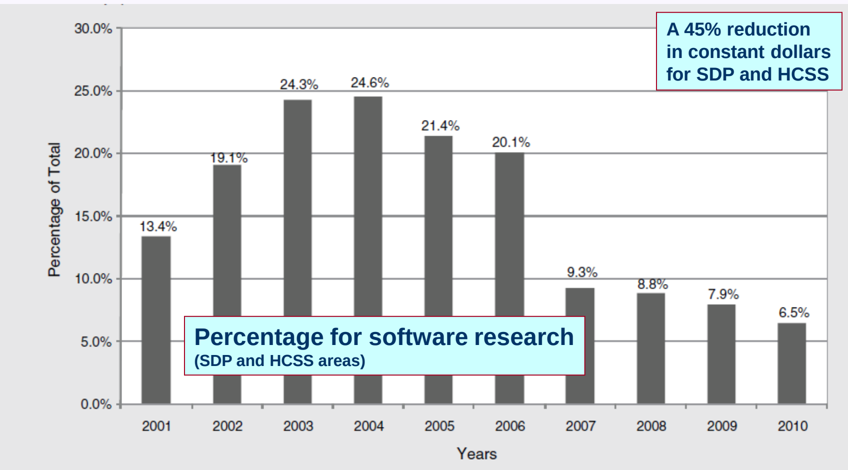


Figure 1.5.3 Percentage of total NITRD investment in either SDP or HCSS.

Deliberation and challenges – Software Myths

- Long-standing **incorrect folklore** regarding defense software producibility (*digested from the report*)

1. DoD software producibility challenges are predominantly challenges of management and process but **not of technology**.

2. DoD and contractors can **rely on industry to innovate** at a rate fast enough to solve the DoD’s hard technical problems and to stay ahead of its adversaries.
   Regardless, there is **sufficient software research already underway** through NSF and other sponsors.

3. Software technology is **approaching a plateau**, which diminishes the need to invest in technology innovation.

13

Deliberation and challenges – Software Myths

- Long-standing **incorrect folklore** regarding defense software producibility (*digested from the report*)

4. We will **never create perfectly reliable and secure** software, so we should focus primarily on provenance—trusted sources—rather than attempting to achieve assurance directly.

5. Earned value management approaches based on **code accumulation** are a sufficient basis for managing software development programs, including incremental iterative development.

14

Outline

- Task and prior reports
- Committee, process, background
- **Areas of practice**
  - **Process and measurement**
  - **Software expertise**
  - Architecture
  - Assurance and security
- Topics of research
- Economic argument
- Next steps

Chapter 2 of the report

15

Incremental and iterative software dev't practices

- Findings
  - Modern processes for innovative software systems is geared toward **incremental identification and mitigation of engineering uncertainties**.
    - In other words: ***Innovative engineering does not necessarily increase programmatic risk***
    - For defense software, challenges derive from (2) larger scale, (2) linking with systems engineering, and (3) arm's-length contractor relationships.
  - **Technology and improved measurement** have significant roles in **enabling** modern incremental and iterative software development practices at all levels of scale.
    - Extensions to **earned value management models** are needed to enable incremental iterative development.
      - These include evidence of feasibility and time-certain development.
      - Additionally, supplement the prescription of DoDI 5000.02 to better support ongoing management of engineering risks

Incremental and iterative software dev’t practices

- Engineering risk can be decoupled from programmatic risk
  - *Iterative engineering of innovative software can be successfully managed*
- Recommendations
  - Take aggressive actions to identify and remove barriers to the broader adoption of **incremental development methods**.
    - These include iterative approaches, staged acquisition, evidence-based systems and software engineering, and related methods that involve explicit acknowledgment and mitigation of engineering risk.
  - The DoD should take steps to **accumulate high-quality data** regarding project management experience and technology choices.
    - This data can be used to inform cost estimation models, particularly as they apply to innovative software development.

There is insufficient DoD-aligned software expertise

- Finding
  - The DoD has a **growing need for software expertise**
  - It is not able to meet this need through intrinsic DoD resources.
    - Nor is it able to fully outsource this requirement to DoD primes.
    - The **DoD needs to be a smart software customer**
      - Particularly for large-scale innovative software-intensive projects.

Outline

- Task and prior reports
- Committee, process, background
- **Areas of practice**
  - Process and measurement
  - Software expertise
  - **Architecture**
  - Assurance and security
- Topics of research
- Economic argument
- Next steps

Chapter 3 of the report

19

Assert architecture leadership

- DoD needs to play an active role in **software architecture**.
  - Software architecture
    - Definition: *The structure or structures of the system, which comprise software components, the externally visible properties of those components, and the relationships among them*.
    - Good architecture entails a **minimum of engineering commitment** that yields a maximum value.
    - Architecture design is an engineering acitvity that is **separate from ecosystems certification** and other standards-related policy setting
- For complex innovative systems:
  - Architecture embodies planning for **flexibility**—defining and encapsulating areas where innovation and change are anticipated.
  - Architecture most strongly influences **quality attributes**
  - Architecture embodies planning for **product lines and interlinking** of systems

continued…

20

Assert architecture leadership

- For innovative systems
  - **Consideration of architecture and quality attributes may best precede commitment to specific functionality.**
  - This approach can reduce the overall uncertainty of the engineering process and yield better outcomes.
  - Architecture includes the earliest and often most important design decisions – those that are most difficult to change later
  - Architecture is profoundly influenced by precedent
    - Small changes can open and close opportunities to exploit rich **ecosystems**, greatly influencing cost, risk, and supply chain structure
- Findings
  - **An early focus on architecture is essential for systems with innovative functional or quality requirements.**
  - **Architecture practice, as seen in industry, is sufficiently mature for DoD to adopt** (Finding3-2)

21

continued…

Assert architecture leadership

- Recommendations (Rec3-2,3-3)
  - **Follow architecture-driven acquisition strategies**
    - **Use as basis for product-line and for systems with multiple leads**
    - **Use architecture to encapsulate innovative elements**
    - **Use architecture to maximize opportunity to build on existing ecosystems**
    - **Support early and continuous validation of architectural decisions**

Outline

- Task and prior reports
- Committee, process, background
- **Areas of practice**
  - Process and measurement
  - Software expertise
  - Architecture
  - **Assurance and security**

Chapter 4 of the report

- Topics of research
- Economic argument
- Next steps

23

Adopt a strategic approach to software assurance

- **Current technical approaches to software assurance are inadequate**.
  - *Assurance*
    - *A human judgment regarding reliability, safety, security, etc.*
  - Current technical approaches need to be augmented
    - Costs range from 30-50% for typical major projects
    - Testing and inspection techniques are inadequate for modern software devt
- **Assurance conclusions are difficult to draw.**
  - Not analogous to reliability models for physical systems
  - Cannot be achieved entirely through *post hoc* acceptance evaluation
    - Quality and security are built in, not “tested in”

Adopt a strategic approach to software assurance

- **DoD faces particular challenges to assurance**.
  1. The **arms-length relationship** between a contractor development team and government stakeholders
  2. Modern systems of all kinds draw on components from **diverse sources**
     - This implies that **supply-chain attacks** must be contemplated, along with attack surfaces within the software application
       - There will necessarily be differences in the levels of trust conferred on components.
       - There may also be opacity in the supply chain for vendor and sub components
     - **Evaluative and preventive approaches** can be integrated to enhance assurance in complex supply chains with diverse sourcing.
  3. **High consequences** due to roles in war-fighting and protection of human lives and national assets
  4. Failure to maintain a lead in the ability to prevent and evaluate confers **advantage to adversaries** (*DSB2007, paraphrased*)

Assurance: models, process, and traceability

- Finding
  - Assurance is **facilitated by advances in diverse aspects** of software engineering practice and technology.
    - These include modeling, analysis, tools and environments, traceability, programming languages, and process support.
    - After many years of slow progress, recent advances have enabled more rapid improvement in assurance-related techniques and tools
    - Advances focused on **simultaneous creation of assurance-related evidence with ongoing development effort have high potential** to improve the overall assurance of systems.
- Finding from DSB2007
  - It is an essential requirement that the United States maintain advanced capability for “test and evaluation” of IT products. Reputation-based or trust-based credentialing of software (“provenance”) needs to be augmented by direct, artifact-focused means to support acceptance evaluation.

Assurance: models, process, and traceability

- **Traceability**: Assurance best practice for development
  - Connect code to be executed with functional and quality attributes
    - Create and sustain ***chains of evidence*** **that link software-related artifacts**
      - Examples: test cases, inspection reports, analysis, simulation, models, etc.
    - **Employ a mix of preventive and evaluative approaches**
    - Address assurance considerations **throughout the process lifecycle**
    - Attend to the means by which design-related information and traceability links are represented
      - Formality, modeling, consistency, and usability
- Finding
  - Early engineering choices strongly influence feasibility of achieving high assurance.
    - Successful approaches involve a diverse set of evaluative and preventive techniques
    - Particularly architecture, modeling, tooling

Assurance concepts in the report – examples

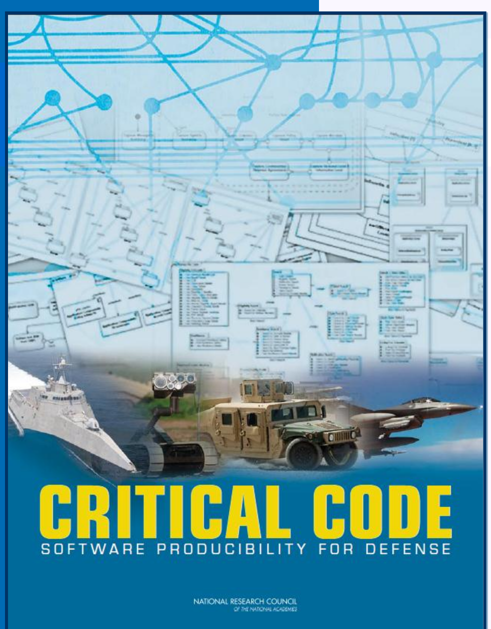
- Scenario structure – combine evaluation and prevention
  1. Hazard and requirements analysis
  2. Architecture and component identification
  3. Component-level error and failure modeling
  4. Supply-chain and development history appraisal
  5. Analysis of architecture and component models
  6. Identify high-interest components
  7. Develop a component evaluation plan
  8. Assess individual components
  9. Select courses of action for custom components
  10. Select courses of action for opaque components and services
  11. Refine system-level assessment
- Two additional security-related challenges
  - Separation
    - E.g., red / green and finer grained
    - Isolation and sandboxing
  - Configuration
    - Including issues related to dynamism

Preventive
- Requirements analysis
- Architecture design
- Ecosystem choice
- Detail design
- Specification and documentation
- Modeling and simulation
- Coding
- Programming language
- Tooling

Evaluative
- Inspection
- Testing
- Direct analysis
- Measurement
- Monitoring
- Verification

Engineering choices influence ability to assure

- Recommendations
  - **Institute effective incentives for preventive software assurance** practices and production of evidence across the lifecycle.
    - Do this throughout the supply chain
    - **Examine commercial best practices for transitioning** assurance-related best practices into development projects (Rec4-3)
    - Including contracted custom development, supply-chain practice, and in-house development practice.
  - **Expand research/investment focus** on assurance-related software engineering technologies and practices (Rec4-2)



THE NATIONAL ACADEMIES
Advisers to the Nation on Science, Engineering, and Medicine

Critical Code

Software Producibility for Defense

Supplementary Material

v03

Outline

- Task and prior reports
- Committee, process, background
- Areas of practice
- **Topics of research**
  - **Seven technology areas**
  - **Four considerations**
  - **Reinvigoration plan**
- Economic argument
- Next steps

Chapter 5 of the report

31

Reinvigorate DoD software engineering research

- Focus research effort in seven technology areas that directly enable producibility improvements

1. **Architecture modeling and architectural analysis**

   Goals:

   (1) Early validation for architecture decisions

   (2) Architecture-aware systems management
   - Including: Rich supply chains, ecosystems, and infrastructure

   (3) Component-based development
   - Including: Architectural designs for particular domains.

2. **Validation, verification, and analysis of design and code**

   Goals:

   (1) Effective evaluation for critical quality attributes

   (2) Components in large heterogeneous systems

   (3) Preventive methods to achieve assurance
   - Including: Process improvement, architectural building blocks, programming languages, coding practice, etc.

3. **Process support and economic models for assurance**

   Goals:

   (1) Enhanced process support for assured software development

   (2) Models for evidence production in software supply chains

   (3) Application of economic principles to process decision-making

continued…

Reinvigorate DoD software engineering research

- Focus research effort in seven technology areas that directly enable producibility improvements

4. **Requirements**

   Goals:

   (1) Expressive models, supporting tools for functional and quality attributes

   (2) Improved support for traceability and early validation

5. **Language, modeling, coding, and tools**

   Goals:

   (1) Expressive programming languages for emerging challenges

   (2) Exploit modern concurrency: shared-memory and scalable distributed

   (3) Developer productivity for new development and evolution

6. **Cyber-physical systems**

   Goals:

   (1) New conventional architectures for control systems

   (2) Improved architectures for embedded applications

7. **Human-system interaction**

   Goal:

   (1) Engineering practices for systems in which humans play critical roles

   (*This area is elaborated in another NRC report*)

Considerations in identifying research topic areas

(1) **Significant potential value** for DoD software producibility
- Process and measurement, architecture, and assurance (chapters 2, 3, 4)

(2) **Feasible progress** in a well-managed research program
- Well-managed with respect to "Heilmeier Questions" (Box5.1)
  - For the identified "Goals" within the seven areas
- There is past success in software research
  - This is now well documented (Box5.2)

(3) **Not addressed sufficiently** by other federal agencies
- Primarily other NITRD-coordinated agencies

(4) **Might not otherwise develop** at a sufficient pace
- In industry or through research sponsored elsewhere

Reinvigorate DoD software engineering research

- Technology role (Finding5-2)
  - **Technology has a significant role in enabling modern incremental and iterative software development practices**
    - At levels of scale ranging from small teams to large distributed development organizations.
    - In all three areas: Process and measurement, architecture, assurance
    - *Myth: DoD's producibility challenges are predominantly challenges of management and process, not technology* (M1)
- Recommendations (Rec5-1,2)
  - **DoD take immediate action to reinvigorate its investment** in software producibility research
    - Undertake through diverse research programs throughout DoD
    - Include academia, industry labs, and collaborations
  - Undertake research programs **in the seven areas**, as critical to advancement of defense software producibility

continued…

Reinvigorate DoD software engineering research

- The *research operating environment*: challenges and success influences
  1. Software engineering is **maturing** as a research discipline.
     - Improved research methods and lower risk in technology transition
     - Facilitating more satisfactory responses to the Heilmeier Questions
  2. **Diffusion pathways** are complex, and there is variability of timescale.
     - Some results can readily transfer to DoD practice
     - Others, often most significant, take longer and are more indirect – raise all ships
  3. **Novelty** is often more about timeliness.
     - Readiness (infrastructure, exponentials) rather than technical novelty
     - *What are the ideas whose time has come?* (E.g., thin/rich clients; utility & cloud)
  4. We can accept **non-quantitative means** to assess progress.
     - Often focus of research is on developing such measures
     - Example: how to assess the benefits of strong typing? In a quantitative way?
- Context:
  - There is a broad challenge in assessing **ROI for basic science** and for research related to enabling technologies.
    - NRC reports address this difficulty for computing technology and software

Roles for academia and industry in research

- Recommendation
  - Academic, industry, and government researchers must all participate (Rec5-1)
- Understand the scope of value of academic research
  - **Workforce**
    - University graduates – prepared for emerging new challenges
    - Next generation technical leadership – from PhD programs
  - **New knowledge**
    - Industry labs under greater ROI pressure
    - Game changing and disruptive technologies
      - Ongoing disruption characteristic of the first 50 years of IT innovation
  - **Non-appropriable invention**, as well as appropriable invention
    - Raise all boats
  - **Surprise reduction**
    - Very rapid change in computing technology, at undiminished pace
    - "Surprise" can include rapid shifts of innovation center of gravity

Outline

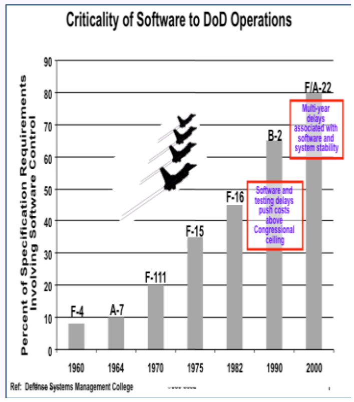
- Task and prior reports
- Committee, process, background
- Areas of practice
- Topics of research
- **Economic argument**
  - **Software has a critical role for DoD**
    - **DoD must take action to address its needs**
    - **DoD must maintain innovation leadership**
    - **Innovation leadership requires sustained R&D**
    - **Software technology is not at a plateau**
- Next steps

Chapter 1 of the report

38

Broadening role of software in DoD, with benefits

- Software has emerged as a key enabler of **capability**, **flexibility**, and **integration** in diverse DoD systems
  - Mission capability embodied in software has become a unique source of strategic and military advantage
  - Extent of system function performed in software, examples (DSB)
  - *Multiple DSB, NRC studies:*
    - *At the core of the ability to achieve integration and maintain mission agility is the ability of the DoD to produce and evolve software*
- Finding
  - **Software has become essential to a vast range of military system capabilities and operations, and its role is deepening and broadening** (1-1)
    - Increase in scale, complexity, and role in manifesting functional capability
    - Increase in interlinking diverse system elements
    - Increase in use for systems development, modeling and simulation



Software's critical role

The strategic significance of software US and globally

- Software has become a principal force multiplier for DoD
  - Rapid growth in extent **and** criticality of software to DoD operations
- Software is a key competitive factor in commercial business
  - Software is now a **strategic source of competitive advantage** in sectors ranging from financial services and health care to telecom and entertainment.
- Disproportionate benefits from software in economic growth
  - ICT industries in US since 1995 [NRC economic policy board]
    - ICT sector is 3% of US GDP
    - ICT drives 20% of US economic growth
  - ICT in Europe
    - ICT sector is 5% European GDP
    - ICT drives 25% of overall growth and 40% of the productivity increase
  - And: Most software development is outside the ICT sector

40

Risks come with the benefits, 1 (Findings1-1,1-2)

- The growing role of software in systems and organizations is creating both **benefits** and **risks**.
  - *Benefit*: Interlinking of systems
    - *Risks for DoD:* Magnitude of failures, cascading failures, security challenges
  - *Benefit*: Direct interaction by users
    - *Risks for DoD:* More individuals can take actions with high consequence
  - *Benefit*: Immediate enactment
    - *Risks for DoD:* Failures and compromises can occur inside human decision loops
  - *Benefit*: Rapid growth in capability and flexibility
    - *Risks for DoD:* Early validation for architecture must be emphasized in the process
    - *Risks for DoD:* Assurance practices and tools need to advance commensurably

Risks come with the benefits, 2 (Finding1-1,1-2)

- **Software supply chains** are increasing complex and diverse.
  - *Benefit*: **Diversification and enrichment of supply-chain structure and geography**
    - *Risks for DoD:* Supply-chain attacks, over-reaction (provenance, ecosystems denial)
    - Enabled by advances in software componentization technology
    - Architectures, frameworks/ecosystems, libraries, and services
    - Technology improvements have enabled modularization and rapid development
    - *Risks for DoD:* Broad component interfaces, complex rules of engagement, assurance
  - *Benefit*: Rich variety of **generally accepted software ecosystems**
    - Ecosystem: conventional structure of infrastructural elements and services that are intended to be combined in a patterned way.
      - Examples: Web services stacks, iPhone, Android, OLAP, LAMP stack, AUTOSAR, SCADA, ERP/SCM/CRM, network hourglasses
    - *Risks for DoD:* security and supply chains, externalities and adoption, compliance practices

DoD software leadership

- **Software capability is strategic**
  - ***At the core of the ability to achieve integration and maintain mission agility is the ability of the DoD to produce and evolve software.***
    (Multiple DSB, NRC studies)

- Findings (Findings1-1,4)
  - Software has become essential to a vast range of military system capabilities and operations, and its role is continuing to deepen and broaden, including interlinking diverse system elements.
  - **The DoD's needs will not be sufficiently met** through a combination of demand-pull from the military and technology-push from the defense or commercial IT sectors.
  - **The DoD cannot rely on industry alone** to address the long-term software challenges particular to defense.

The role for DoD in its software leadership

- Findings
  - **Technological leadership in software is a key driver of capability leadership in systems.**
    - DoD relies on US industry to sustain this technological leadership.
  - The DoD relies fundamentally on mainstream commercial components, supply chains, and software ecosystems.
    - Nonetheless, the DoD **has special needs** in its mission systems driven by the growing role of software in systems.
- Recommendations (Rec1-1,5-1)
  - **DDR&E should regularly undertake** an identification of areas of technological need related to software producibility where the DoD has **"leading demand"** and where accelerated progress is needed
  - **DoD take immediate action to reinvigorate its investment in software producibility research**
    - Undertake research programs in the seven areas (Rec5-2,recap)

At a plateau?

- The myth of the plateau
  - We are not at a plateau or near a plateau in overall software capability or technology for software producibility (Finding1-5a)

    "Automatic programming" – 1958 (Fortran), 1980s (4GLs), 1980s (AI), etc.
  - Software has intrinsic unboundedness
    - It lack of natural physical limits on scale and complexity
      - Only human intellectual limits and mathematical limits on algorithms
    - New software-manifest capabilities continue to emerge
      - A "continuous improvement" in capability (as distinct from process)
      - Less fine tuning and more order-of-magnitude leaps
    - Enabled by a steady pace of technological breakthroughs in practices, models, languages, tools, and practices
      - Leveraged through ecosystems and infrastructure
  - There is a consequence necessity of ongoing innovation in software
    - Software innovation, once routinized, is then quickly automated
    - Expensive custom dev't gives way to low-cost component procurement

Consequences of unboundedness

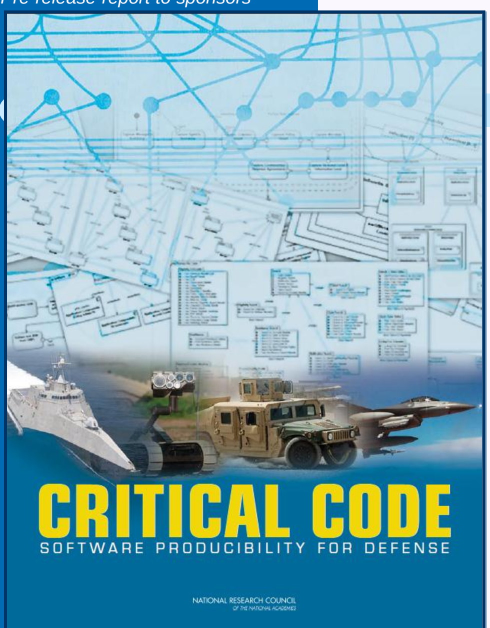
- **Software engineering and other engineering**
  - **A relatively *much larger portion* of overall software engineering effort is creating *innovative* functionalities, as compared with other engineering disciplines**
  - Hence an ongoing focus on *engineering risk*
- **Staying apace**
  - **Mere presence as a software user** requires keeping pace with rapid ongoing innovation and improvement to practices
    - Applies to custom development, components, and ecosystems
  - **Leadership as software producer or consumer** requires more
    - An active organizational role in defining the architecture of systems and influencing ecosystems
    - Participation in technology development

The consequent necessity of ongoing software innovation

- ***Findings*** *(1-3b,5b)*
  - **To avoid loss of leadership, DoD must be more fully engaged in the innovative processes related to software producibility**
    - There is strategic value to DoD in sustaining US leadership in software producibility -- compared with other industries that have moved offshore
  - It is an **essential requirement that the United States maintain advanced capability for "test and evaluation" of IT products**. Reputation-based or trust-based credentialing of software ("provenance") needs to be augmented by direct, artifact-focused means to support acceptance evaluation. *(DSB2007)*
  - DoD needs to address directly the challenge of building on and, where appropriate, contributing to the development of mainstream software that can contribute to its mission.

Pre-release report to sponsors

THE NATIONAL ACADEMIES
Advisers to the Nation on Science, Engineering, and Medicine



DoD Software
Needs and Priorities

Final report of the
Committee on Advancing Software-Intensive Systems Producibility (ASISP)

William Scherlis, *Chair*

Enita Williams, Study Director
Jon Eisenberg, CSTB Director

December 2010

National Research Council (NRC)
Computer Science and Telecommunications Board (CSTB)

v03