Dynamics of Integrating Software Assurance Engineering Activities into the System Acquisition Life Cycle

Dr. Kenneth E. Nidiffer

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Software Engineering Institute | **Carnegie Mellon University**

Dynamics of Integrating Software Assurance Engineering Activities into the System Acquisition Life Cycle

Software Engineering Institute | **Carnegie Mellon University**

Distribution Statement A: Approved for Public Release; Distribution is Unlimited

© 2017 Carnegie Mellon University
Dr. Kenneth E. Nidiffer
NDIA 20th Annual SE Conference
October 23 - 26, 2017

2

Software-Enabled Systems Are Today's Strategic Resource

Dr. Bill Scherlis*

"Software is the building material for modern society"

Software

Oil

Steam

Water

Manual Labor

Increasing Globalization, Productivity, and Complexity

Context: Increasingly Software Assurance Is a Moving Target

- **Definition:** Software assurance provides the required level of confidence that software functions as intended (and no more) and is free of vulnerabilities, either intentionally or unintentionally designed or inserted in software, throughout the life cycle*

- **Perspective:** The changing and expanding role that software plays in cyberspace means that the development of software-intensive systems must continue to evolve while we pursue software assurance

* NDAA 2013, Section 933

Source: SEI

4

Challenges: Integrating Software Assurance Engineering Activities into the System Acquisition Life Cycle

1. Increasing complexity of software-intensive systems
2. Satisfying unique operational mission and business needs
3. Solving the vulnerability identification chasm
4. Addressing system sustainment as a strategic initiative
5. Handling the expanding code base
6. Understanding attack patterns, vulnerabilities, and weaknesses
7. Increasing vulnerabilities
8. Designing-in software quality throughout the life cycle
9. Reducing technical debt
10. Working in the infancy of the software engineering discipline

Context: Software Assurance/Cyber Imperative

- Software is a foundation of the DoD's military power and the building material for modern society

    - *Software assurance is a moving target*

- The Office of the Under Secretary of Defense for Acquisition, Technology, and Logistics (USD(AT&L)) updated DoDI 5000.02 to include a new Enclosure 14 - 2017. The policy states, in part,

    - *Program managers, assisted by supporting organizations to the acquisition community, are responsible for the cybersecurity of their programs, systems, and information…"*

- Direct link between cybersecurity engineering and systems and software assurance engineering*

Cyber Security Engineering: A Practical Approach for Systems and Software Assurance, Carol Woody and Nancy Mead, 2017

Software Engineering Institute | **Carnegie Mellon University**

Distribution Statement A: Approved for Public Release: Distribution is Unlimited

© 2017 Carnegie Mellon University
Dr. Kenneth E. Nidiffer
NDIA 20th Annual SE Conference
October 23–26, 2017

6

Context: Dynamics of Software

- Software is ubiquitous and growing in importance

- Codebases are increasing

- Vulnerabilities (defects, flaws) are increasing

- Software represents increasingly more system functionality and cost

- Research is needed to address emerging software challenges

- Software-reliant systems are becoming more complex and intertwined

- There is national and global dependence on software

- We need to improve the management of software-intensive systems

- Software assurance is increasingly important, and achieving it is a moving target

Context: DoD Stakeholders and Different Perspectives on Software Assurance

Source: DAU

Software Engineering Institute | **Carnegie Mellon University**

Distribution Statement A: Approved for Public Release; Distribution is Unlimited

© 2017 Carnegie Mellon University
Dr. Kenneth E. Nidiffer
NDIA 20th Annual SE Conference
October 23–26, 2017

8

Context: The Fabric of Computing Is Changing, and Achievement of Software Assurance Is More Challenging

Networks are becoming software defined

- Generic network nodes can adapt function to usage and demand

Field-Programmable Gate Arrays (FPGAs) are proliferating

- Now even the processor's function is determined by software and malleable after fielding

Emergence of Artificial Intelligence and Machine Learning

- Tasks change from direct programming to data curation and feature discovery

Source: SEI

Software Engineering Institute | **Carnegie Mellon University**

© 2017 Carnegie Mellon University
Dr. Kenneth E. Nidiffer
NDIA 20th Annual SE Conference
October 23–26, 2017

9

Distribution Statement A: Approved for Public Release: Distribution is Unlimited

Increasing Complexity of Cybersecurity Systems
Complexity and How We Interpret It Are Key Drivers in Assurance
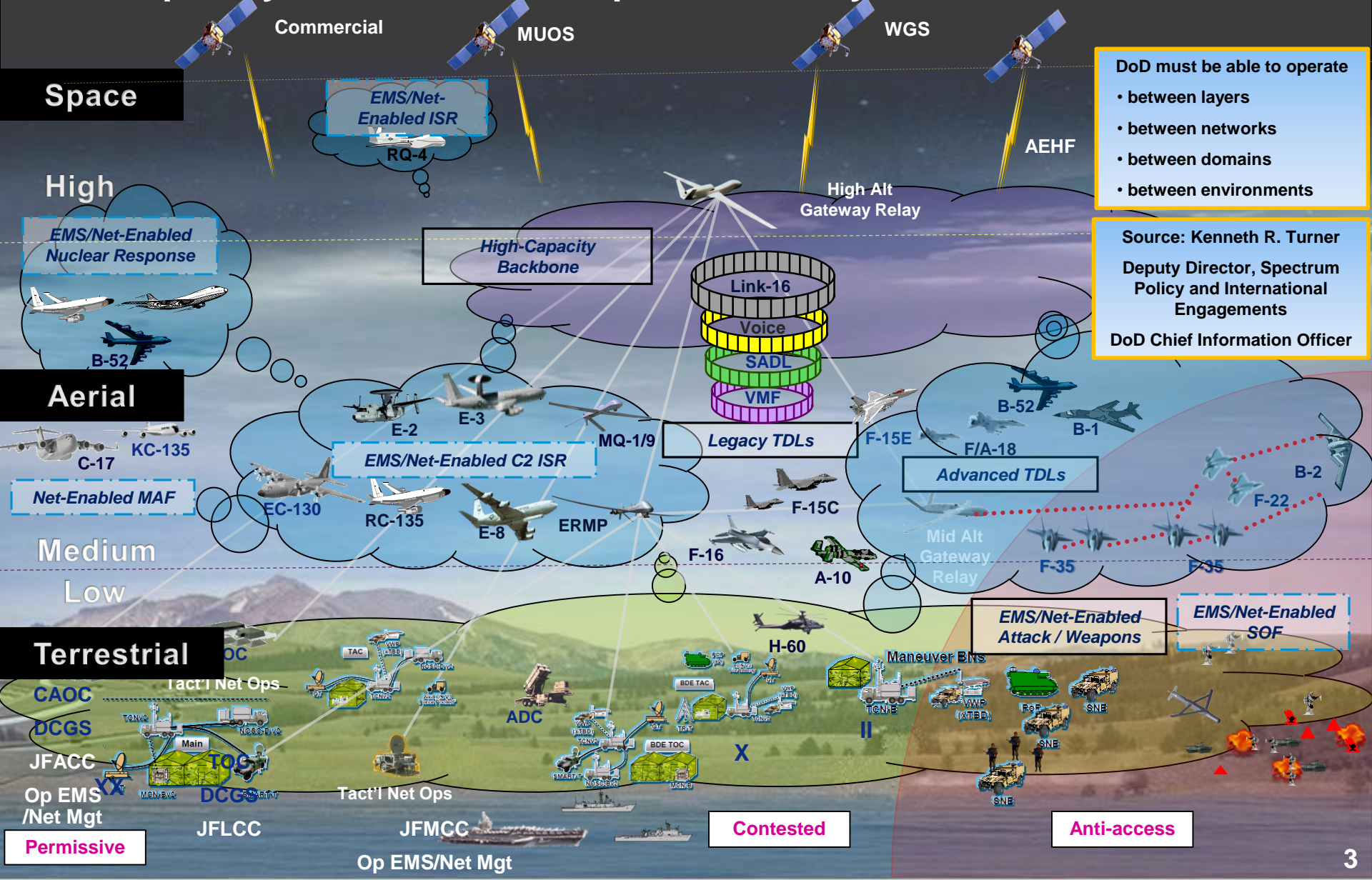
Satisfying Unique Operational Mission and Business Needs as Commercial Products Are Integrated into Military Systems

Source: SEI

Software Engineering Institute | **Carnegie Mellon University**

Distribution Statement A: Approved for Public Release; Distribution is Unlimited

© 2017 Carnegie Mellon University
Dr. Kenneth E. Nidiffer
NDIA 20th Annual SE Conference
October 23–26, 2017

11

Solving the Vulnerability Identification Chasm
First line of defense in software assurance is the application (software) layer

The Application Layer
Internal Devices
Internal Network
Internal Firewall
DMZs
Security Router
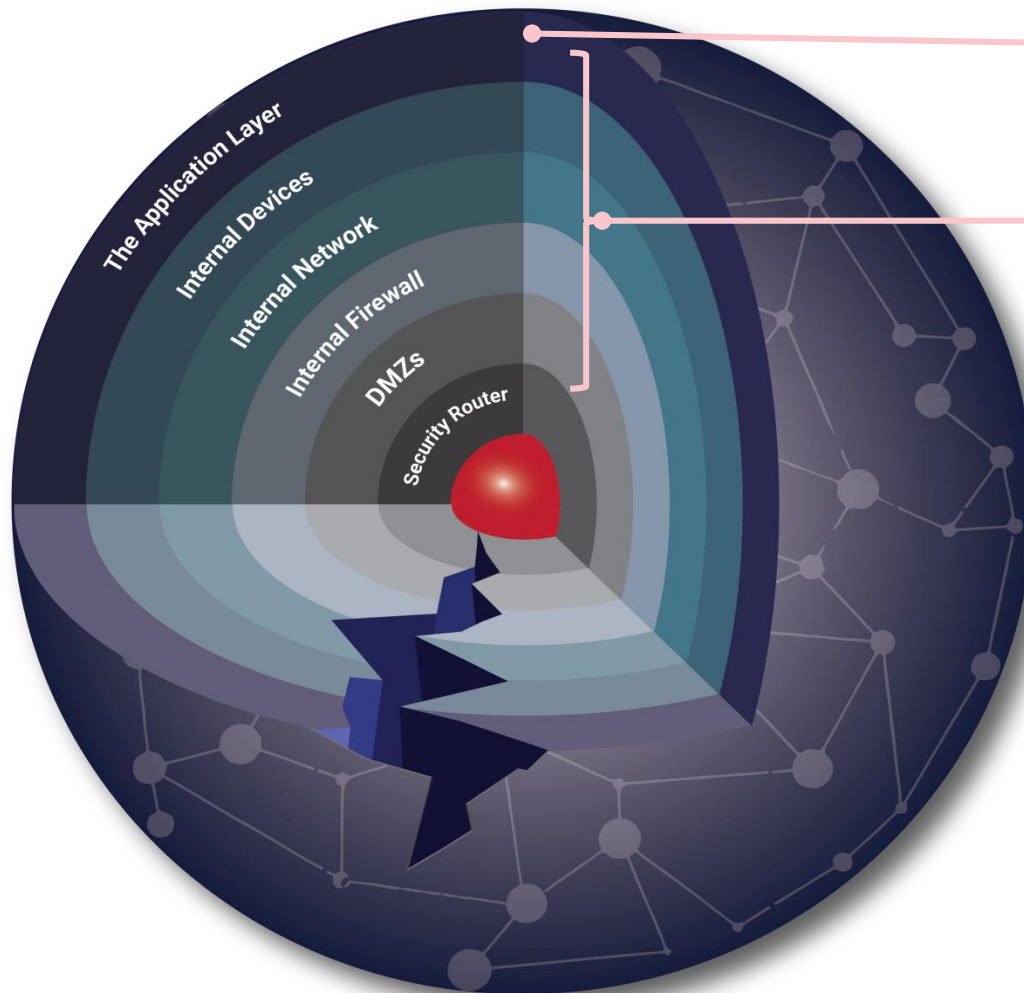
84% of breaches exploit vulnerabilities in the application[1]

Yet funding for IT defense vs. software assurance is 23 to 1[2]

1. Clark, Tim, "Most Cyber Attacks Occur from This Common Vulnerability," *Forbes*, 03-10-2015

2. Feiman, Joseph, "Maverick Research: Stop Protecting Your Apps; It's Time for Apps to Protect Themselves," *Gartner*, 09-25-2014

Software Engineering Institute | **Carnegie Mellon University**

Distribution Statement A: Approved for Public Release; Distribution is Unlimited

© 2017 Carnegie Mellon University
Dr. Kenneth E. Nidiffer
NDIA 20th Annual SE Conference
October 23–26, 2017

12

Addressing System Sustainment
Strategic Trends Impacting Software Sustainment

- The future is software-centric…dependent…

- Advanced DoD capabilities manifested in algorithms/software
  - Autonomy, machine learning, collaborative systems, competitive networking, system resiliency, IoT, hybrid cloud computing, human-computer interaction, digital security, analytics for decision making…
  - Anticipating/adapting to new dynamic threats and requirements

- Warfighter needs coupled with technology innovation dynamics drive need for strategic enterprise focus on evolving a DoD enterprise life-cycle software strategy to be aligned with this changing environment
  - Policies
  - Current and future infrastructure capabilities
  - Business models
  - Investment strategies

THE FUTURE SOLDIER

HELMET
Extreme head protection

HEADS-UP DISPLAY
Enhances situational awareness, provides critical data

SOLAR PANELS
Captures solar energy

BIOSENSORS
Sensors in fabric measuring vitals

ARMOR
Complete ballistic protection

BODY
Nanoparticles protect against impact

LEG BRACE
Captures kinetic energy

EXOSKELETON
Improves performance and endurance

Software Engineering Institute | **Carnegie Mellon University**

Handling the Expanding Code Base
Software is dramatically expanding with limited natural governance

Size of Codebase (SLOC)

Google = 2,000 MSLOC

Healthcare.gov
Debian 5.0 (all software in package)
Software in typical new car, 2013
Mac OS X 10.4
Debian 5.0 codebase
US Army's Future Combat System
Facebook (without backend code)
Windows Vista (2007)
Microsoft Visual Studio 2012
Large Hadron Collider
Microsoft Office (2013)
Windows XP (2001)
Windows 7
Symbian
Microsoft Office for Mac (2006)
Windows 2000 (2000)
Microsoft Office (2001)
F-35 Fighter
Apache Open Office
Linux 3.1 (recent version, 2013)
Android (upper estimate)
Boeing 787, total flight software

0   100   200   300   400   500   600

Millions of Source Lines of Code

Source: David McCandless, "Information is Beautiful," 21 September 2016, web retrieval

Understanding Attack Patterns, Vulnerabilities, and Weaknesses
Defining software assurance attributes to satisfy information needs

water — known threat actors

flooding — attack patterns (CAPEC)

hatch door in sub — weaknesses & vulnerabilities (CWE/CVE)

close door actions* — System & System Security Engineering Trades

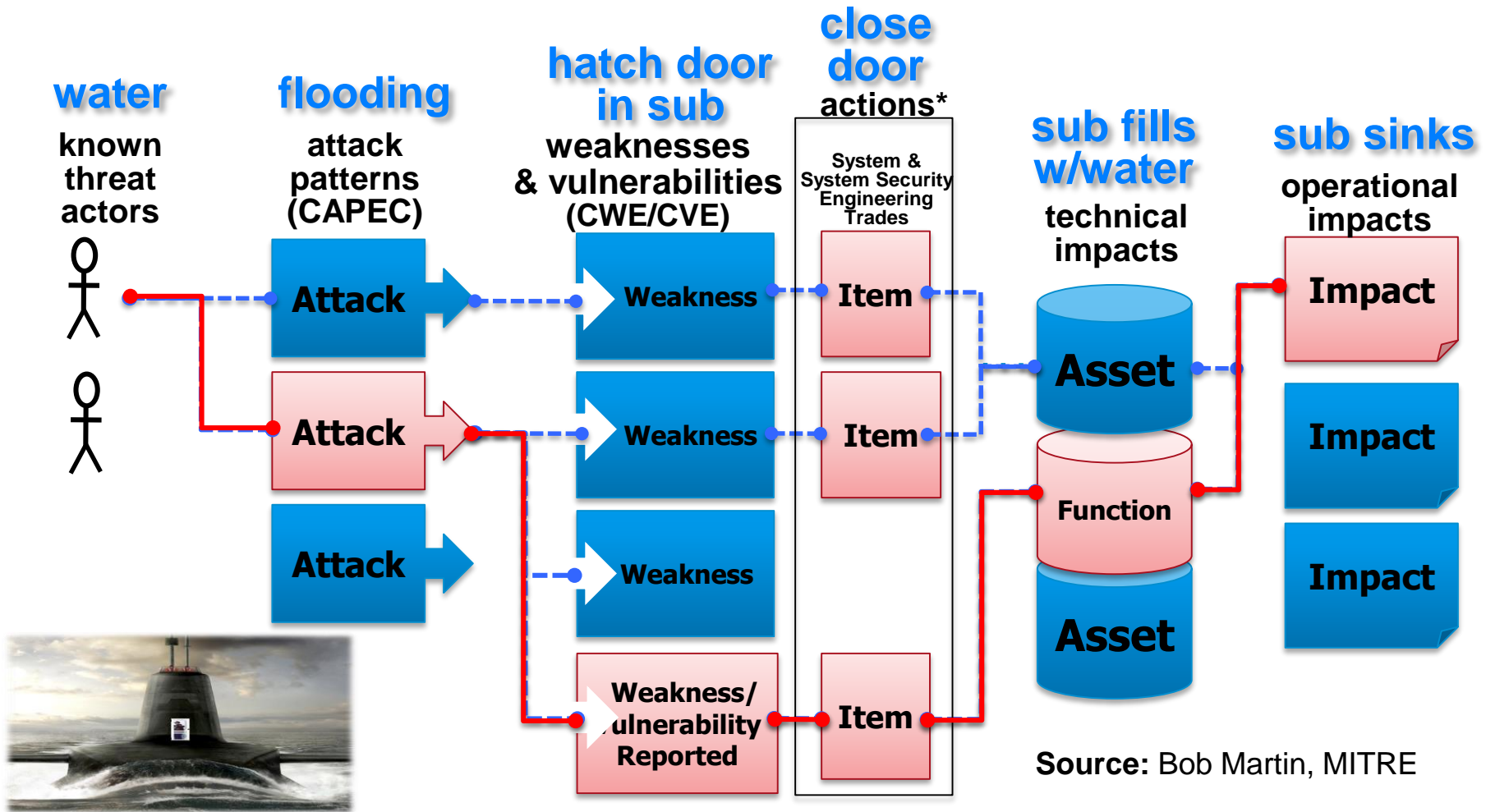**sub fills w/water** — technical impacts

sub sinks — operational impacts

Attack → Weakness → Item → Asset → Impact
Attack → Weakness → Item → Function → Impact
Attack → Weakness → Asset → Impact
Weakness/Vulnerability Reported → Item → Impact

Source: Bob Martin, MITRE

- **Actions** include architecture choices; design choices; added security functions, activities, and processes; physical decomposition choices; static and dynamic code assessments; design reviews; dynamic testing; and pen testing.
- **Vulnerability** is the intersection of three elements: a system susceptibility or flaw, attacker access to the flaw, and attacker capability to exploit the flaw.

Increasing Vulnerabilities: CVE 1999 to 2017, Reported Common Vulnerabilities and Exposures (CVE)
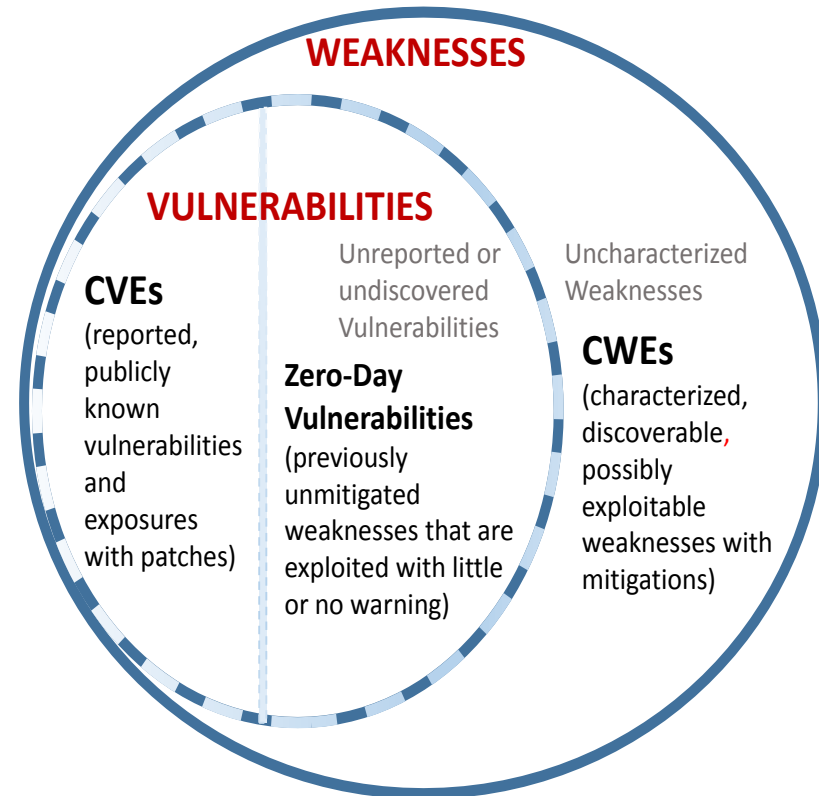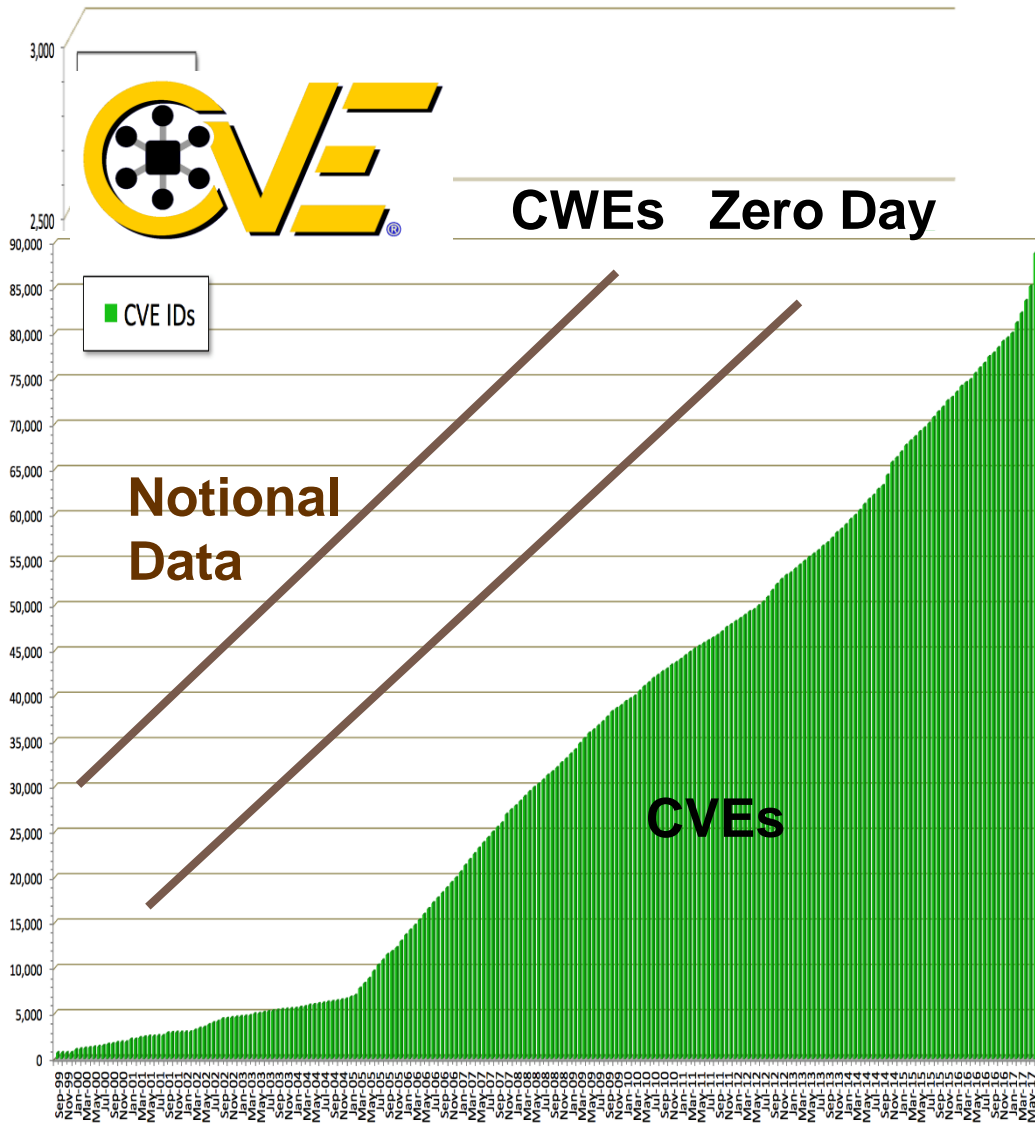
Source: Dr. Robert A. Martin, MITRE Corporation, May 2017

Designing-in Software Assurance Throughout the System Life Cycle

Where Software Flaws Are Introduced

| 70% | | | 20% | 10% |

| Requirements Engineering | System Design | Software Architectural Design | Component Software Design | Code Development | Unit Test | Integration | System Test | Acceptance Test | Operation |

| 3.5% | | | 16% | 50.5% | 9% | 21% |

Where Software Flaws Are Found
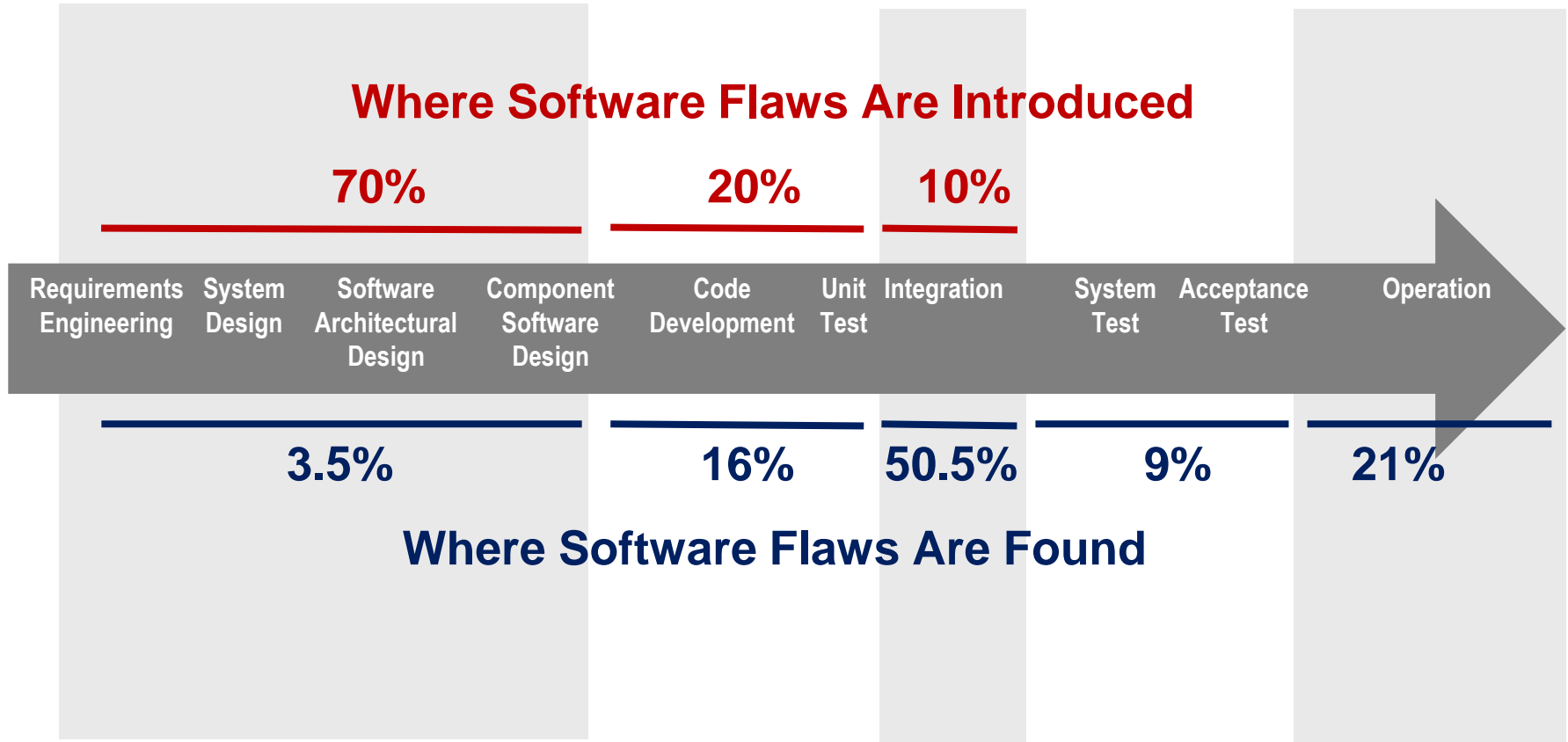
Special emphasis needed up-front in the system life cycle

Sources: *Critical Code*, NIST, NASA, INCOSE, and Aircraft Industry Studies

Reducing Technical Debt Over the System Life Cycle
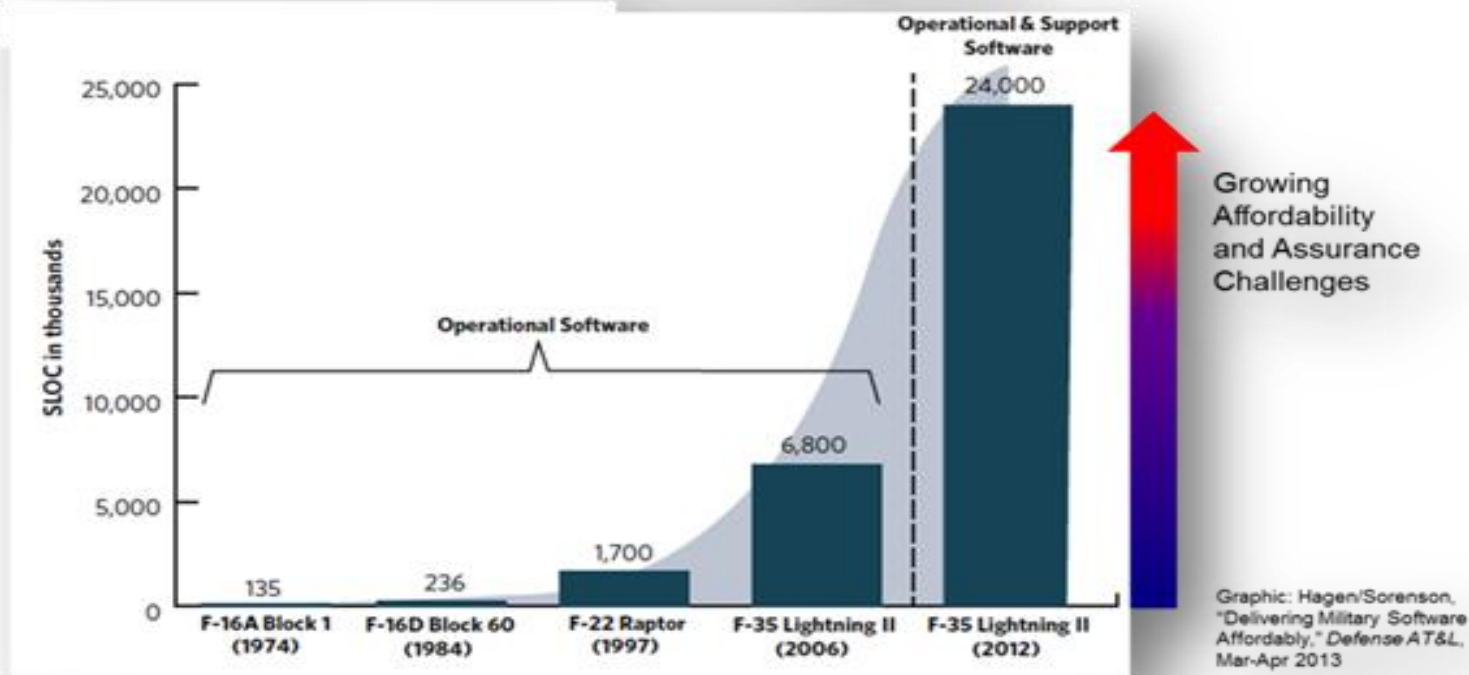
Source: U.S. Air Force Scientific Advisory Board. *Sustaining Air Force Aging Aircraft into the 21st Century* (SAB-TR-11-01). U.S. Air Force, 2011.

Reducing Technical Debt

RDT&E Funding by DAE Tenure Period (1997–2014)

May Not Be Right Direction if Pushing Costs to Sustainment

Right Direction if More Assured Programs

Source: *Performance of the Defense Acquisition System 2015 Annual Report*, Sep. 16, 2015

19
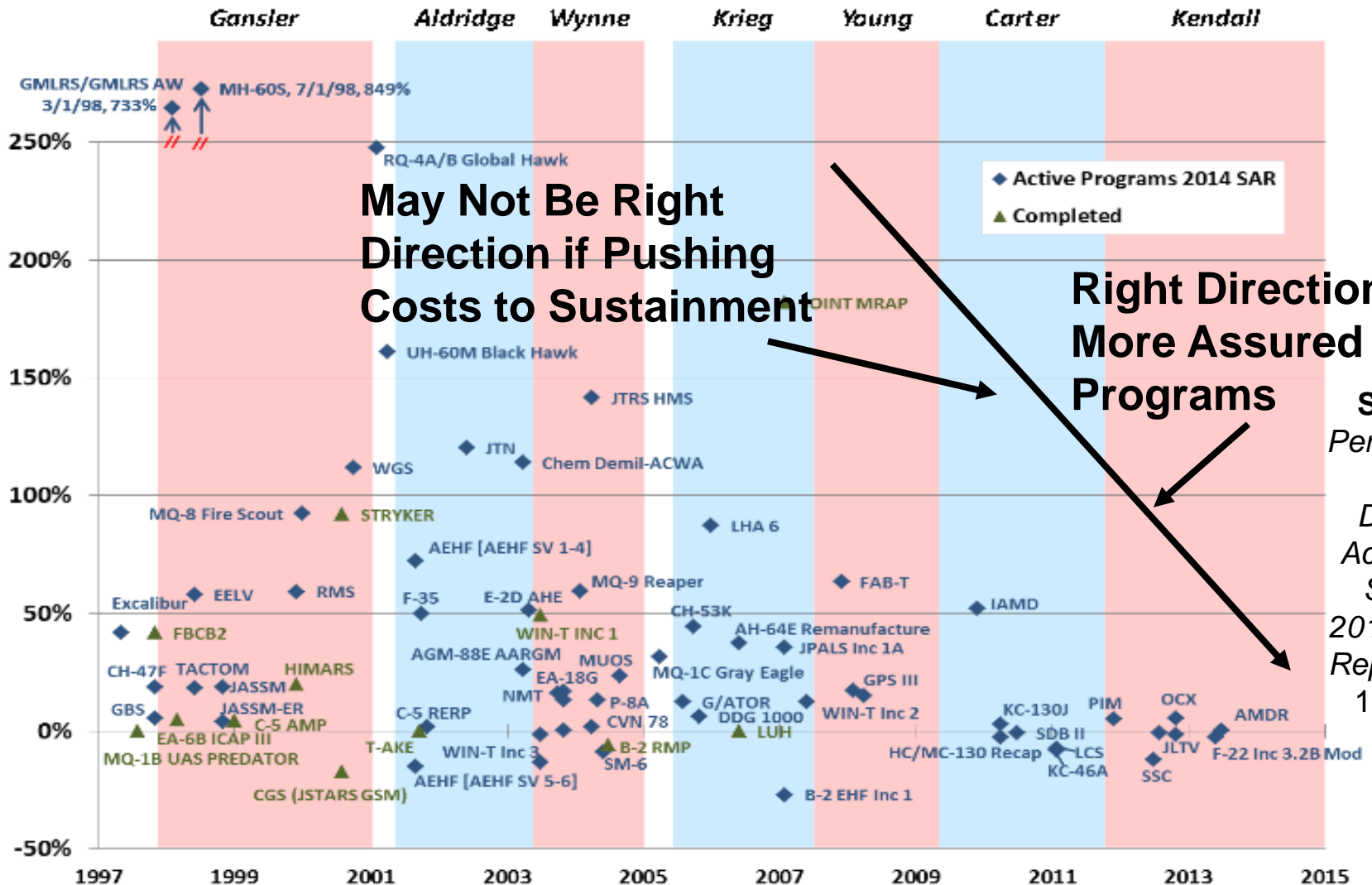
Reducing Technical Debt

Software Development Lifecycle
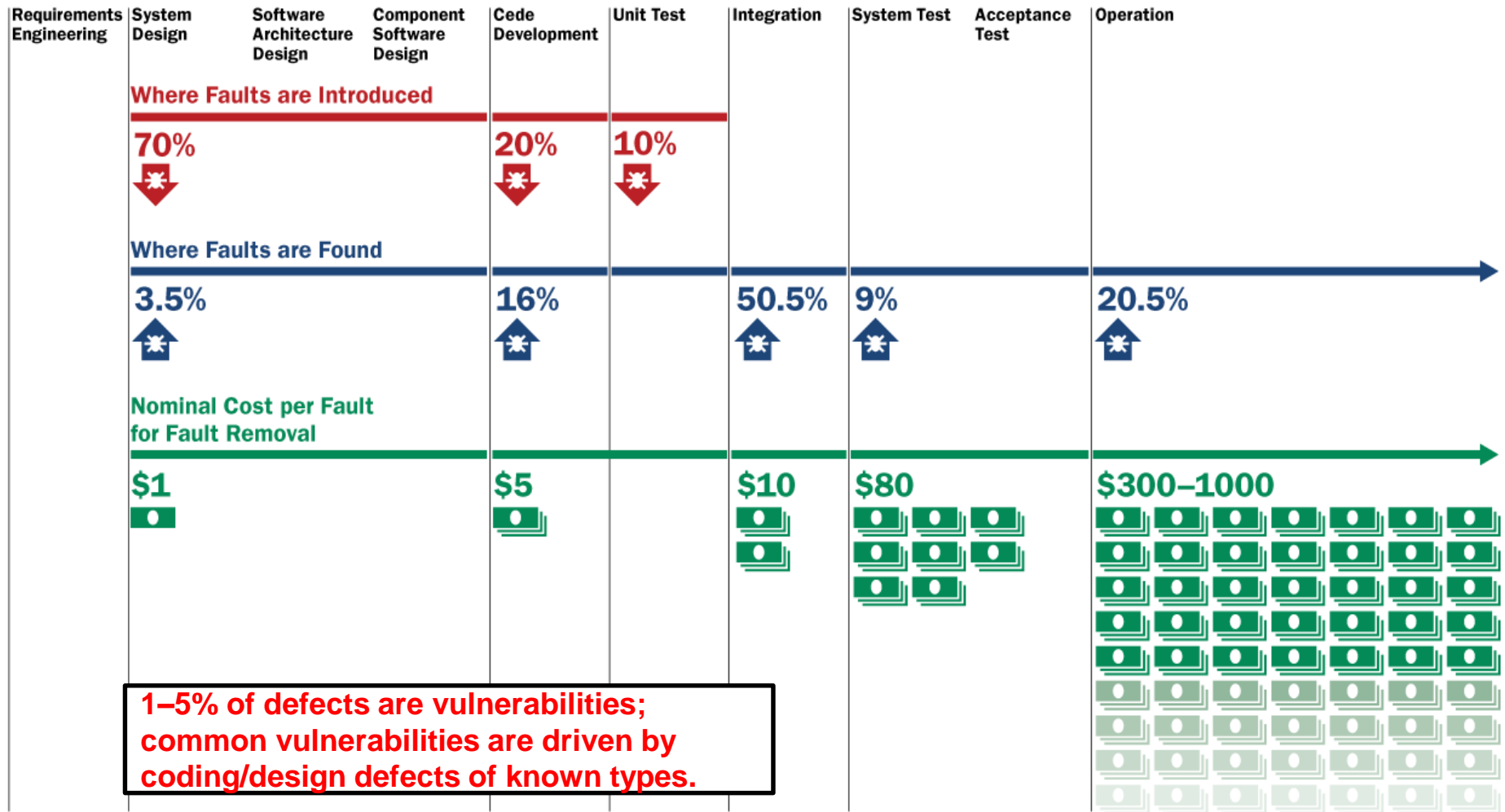
Requirements Engineering	System Design	Software Architecture Design	Component Software Design	Cede Development	Unit Test	Integration	System Test	Acceptance Test	Operation

Where Faults are Introduced

| | 70% | | | 20% | 10% | | | | |

Where Faults are Found

| | 3.5% | | | 16% | | 50.5% | 9% | | 20.5% |

Nominal Cost per Fault for Fault Removal

| | $1 | | | $5 | | $10 | $80 | | $300–1000 |

1–5% of defects are vulnerabilities; common vulnerabilities are driven by coding/design defects of known types.

Sources: Critical Code, NIST, NASA, INCOSE, and Aircraft Industry Studies

Reducing Technical Debt: Engineering-in Software Assurance Activities Across the Life Cycle

Understand SwA Risks · **Design Principles** · **Design Inspection** · **Secure Coding Practices** · **Code Inspection** · **Test Coverage Metrics** · **Pen Test** · **Monitor Threats/ Attacks** · **Respond to Incidents Defects** · **Upgrade SwA**
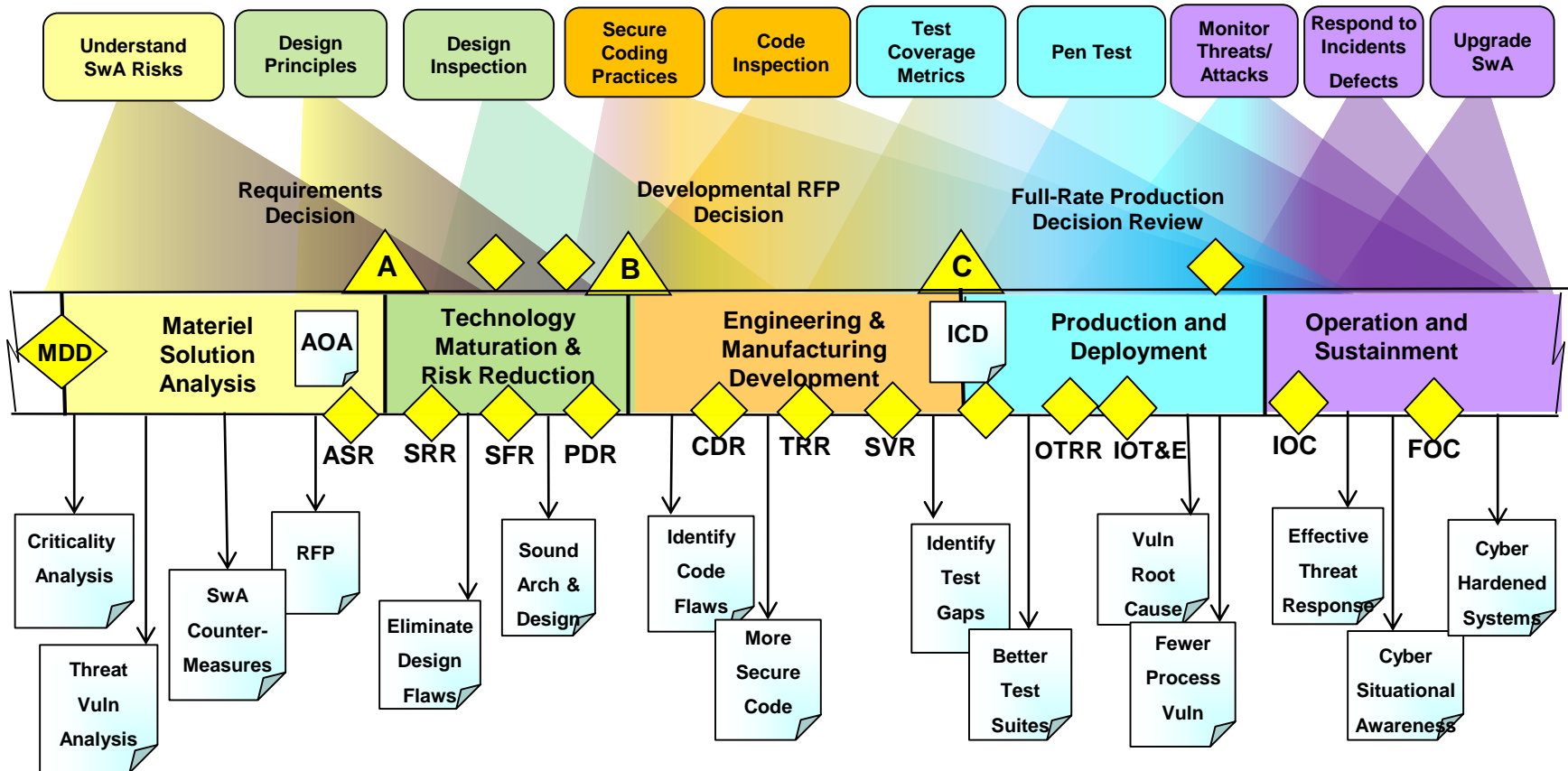
Requirements Decision · Developmental RFP Decision · Full-Rate Production Decision Review

MDD — Materiel Solution Analysis — AOA — A — Technology Maturation & Risk Reduction — B — Engineering & Manufacturing Development — ICD — C — Production and Deployment — Operation and Sustainment

ASR · SRR · SFR · PDR · CDR · TRR · SVR · OTRR · IOT&E · IOC · FOC

Criticality Analysis · Threat Vuln Analysis · SwA Counter-Measures · RFP · Eliminate Design Flaws · Sound Arch & Design · Identify Code Flaws · More Secure Code · Identify Test Gaps · Better Test Suites · Vuln Root Cause · Fewer Process Vuln · Effective Threat Response · Cyber Situational Awareness · Cyber Hardened Systems

DoD PM's Guidebook for Integrating Software Assurance Engineering Activities into the Systems Acquisition Lifecycle (Being Developed)

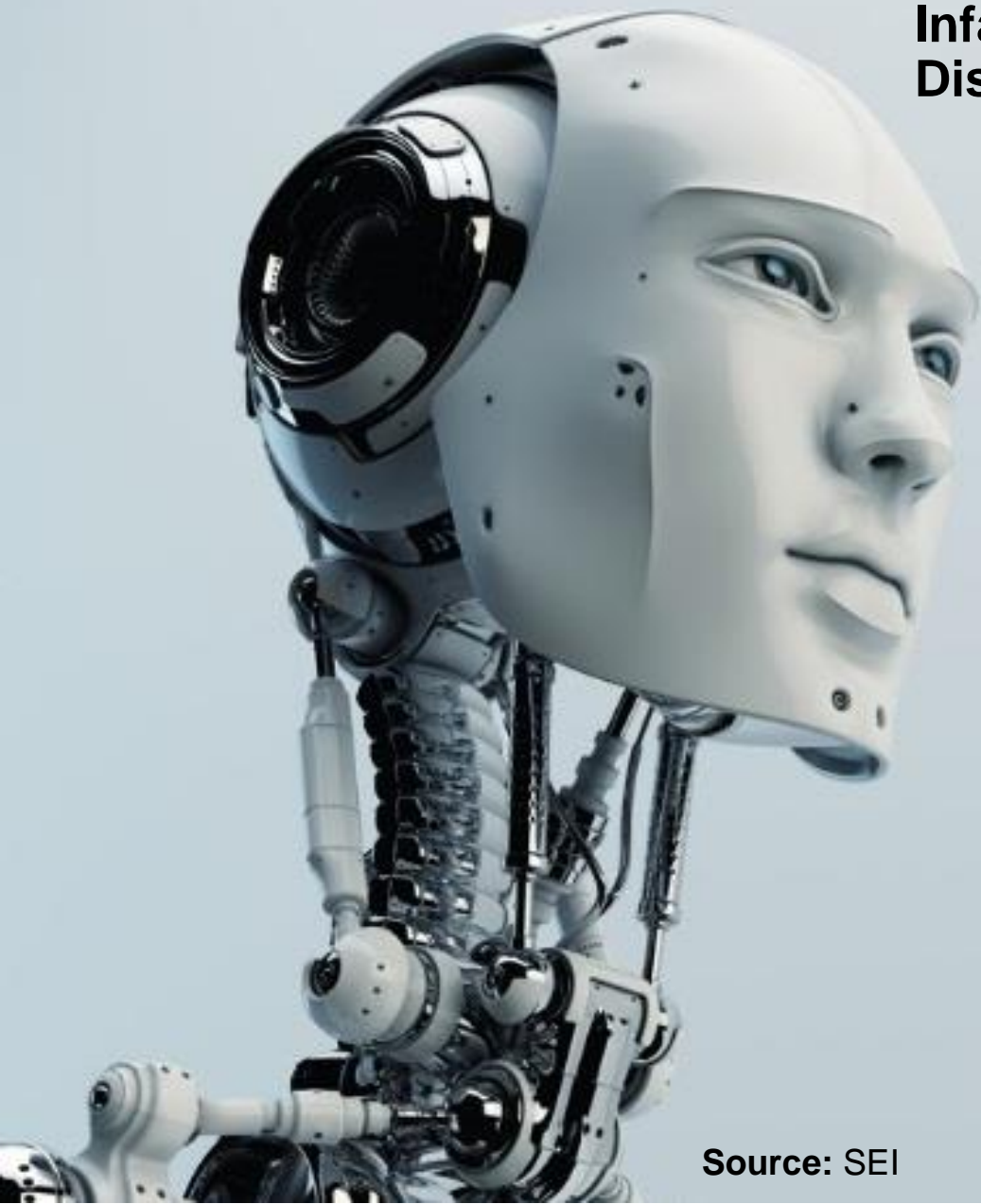# Working in the Infancy of the Software Engineering Discipline

Improving the workforce by developing software core competencies and a DoD career field in software engineering

	Physical Science	Bioscience	Computer/Software/Cyber Science
Origins/History	Begun in antiquity	Begun in antiquity	Mid-20th century
Enduring Laws	Laws are foundational to furthering exploration in the science	Laws are foundational to furthering exploration in the science	Only mathematical laws have proven foundational to computation
Framework of Scientific Study	Four main areas: astronomy, physics, chemistry, and earth sciences	Science of dealing with health maintenance and disease prevention and treatment	• Several areas of study: computer science, software/systems engineering, IT, HCI, social dynamics, AI • All nodes are attached to and rely on a netted system
R&D and Launch Cycle	10–20 years	10–20 years	Significantly compressed; solution time to market must happen very quickly

HCI: human–computer interaction; AI: artificial intelligence

Source: SEI

Infancy of Software Engineering Discipline: Human-Machine Teaming

In the real world, autonomy is usually granted within some context—explicit or implicit

- parents and children
- soldiers, sailors, marines, and airmen

How do we do this for machines?

- Explicit may be easy, but implicit is hard for machines
- Commander's intent
- Mission orders

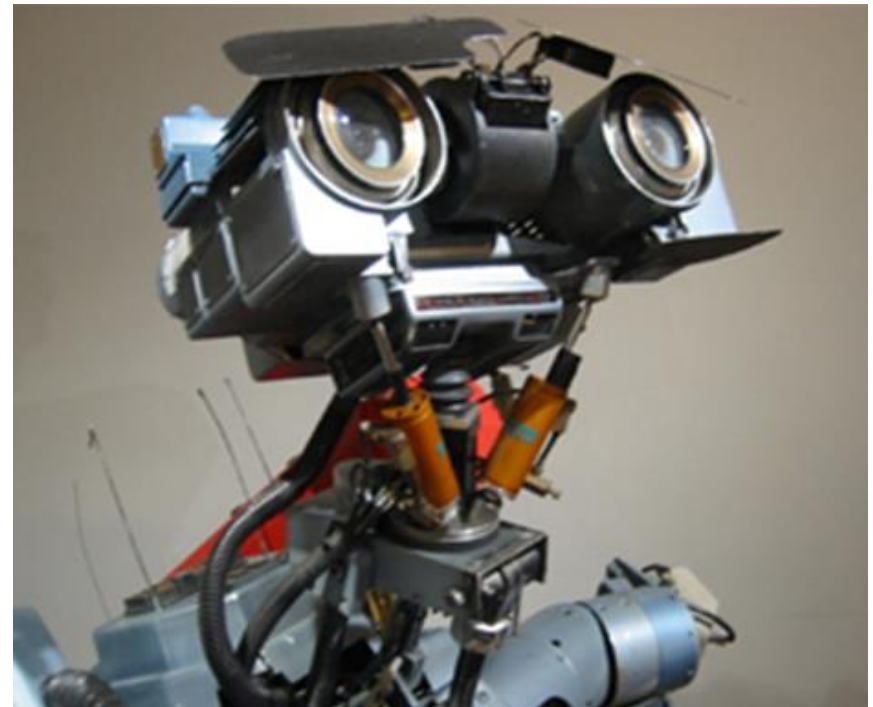Related to need for explainability and predictability

Source: SEI

So Where Does This Lead Us?

- A more robust software assurance approach will be needed…

- Decision makers will need insight and understanding about how to achieve software assurance

- As software-dominated system projects become larger in scope/complexity, capitalizing on opportunities for making better decisions will become more important

  - Critical to shift from asking "what happened?" which is a question of information based on sparse data

  - To seeking insight by asking "what happened, why, how do we solve the problem, and can we evaluate that it has been solved?"

- Enabling an engineering-based approach that seeks to design-in software assurance is becoming more important

- DoD workforce needs a software engineering career field that includes software assurance core competencies

Final Thought: Advanced Software Engineering with Operational Participation

Will determine if we create C-3PO and Johnny 5 . . .

Source: SEI

Software Engineering Institute | **Carnegie Mellon University**

Distribution Statement A: Approved for Public Release: Distribution is Unlimited

© 2017 Carnegie Mellon University
Dr. Kenneth E. Nidiffer
NDIA 20th Annual SE Conference
October 23–26, 2017

25

…or the Borg

Alamy Stock Photo

Source: SEI

Contact Information

Dr. Kenneth E. Nidiffer, Director of Strategic Plans for Government Programs

Software Engineering Institute

Carnegie Mellon University

Office: + 1 703-247-1387

Fax: + 1 703-908-9235

Email: Nidiffer@sei.cmu.edu

Software Engineering Institute | **Carnegie Mellon University**

Distribution Statement A: Approved for Public Release; Distribution is Unlimited

© 2017 Carnegie Mellon University
Dr. Kenneth E. Nidiffer
NDIA 20th Annual SE Conference
October 23- 26, 2017

27