Measurement Challenges: Engineering-In Software Quality Assurance to the System Acquisition Lifecycle

Dr. Kenneth E. Nidiffer

Eighteenth Practical Software and Systems Measurement Users’ Group Workshop
Measurement in a Complex Environment

Lockheed Martin Global Vision Center
Crystal City, Arlington, VA

12-16 June 2017
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213
Measurement Challenges: Engineering-In and Transitioning Software Assurance into the System Acquisition Lifecycle

Copyright 2017 Carnegie Mellon University. All Rights Reserved.
This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.
The view, opinions, and/or findings contained in this material are those of the author(s) and should not be construed as an official Government position, policy, or decision, unless designated by other documentation.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution. Please see Copyright notice for non-US Government use and distribution.
This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at permission@sei.cmu.edu.
DM17-0302
Software-Enabled Systems Are Today’s Strategic Resource

“Software is the building material for modern society”

Dr. Bill Scherlis*

Manual Labor

Water

Steam

Oil

Source: SEI

Increasing Globalization, Productivity, and Complexity
Engineering-In Software Quality Assurance Throughout the SDLC* - Measurement Challenges

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

* SDLC: System Development Life Cycle
Context: Software Assurance/Cyber Imperative

- In early 2017, 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. 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…”

- Persistent pursuit of software quality assurance is a constant struggle in part because processes, methods, and measurements across the lifecycle are not mature

- Program managers’ guidebooks for engineering software quality assurance into the system acquisition lifecycle are being worked on
Context: Increasingly Software Quality Assurance Is a Moving Target

- **Definition**: The level of confidence that software functions as intended and is free of vulnerabilities, either intentionally or unintentionally designed or inserted as part of the software throughout the lifecycle.

- **Moving target**: The changing and expanding role that software plays in cyberspace means that the development of software-enabled systems must continue to evolve while we pursue software quality.

*Source: DoDI 5200.44 Protection of Mission Critical Functions to Achieve Trusted Systems and Networks (TSN), and 2013 NDAA S933*
Intelligence Model – Support for Decision Making

Relationship of Data, Information and Intelligence

Source: Joint Intelligence / Joint Publication 2-9 (Joint Chiefs of Staff)
Context: Enduring Questions That Drive Hard Choices About This Imperative

- How much is enough?
- How much does “enough” cost?
- Is “enough” affordable?
- How does one decide?
- How does one evaluate the “goodness” of the decision?

Measures Supporting Advancements in Program Management, With Operational Participation, are Needed

Source: SEI
Context: Traditional Project Measures Integrated with Big Data Measures

Nexus Drivers

- Dynamic policies and domain characteristics
- Rapidly changing needs
- Blurring of traditional organizational lines
- Complex governance
- $, resources, workforce constraints

- Technology optempo!
  - big data collection
  - processing information
  - analytics and production

- Measurement model may be domain dependent

Source: SEI
Context: Example Domain - Autonomous Systems

- Algorithmically driven agents will work in 5% of economic transactions
- 20% of all business content will be authored by machines
- 6 billion connected things will be requesting support
- 50% of the fastest growing companies will have fewer employees than smart machines
- More than 3 million workers globally will be supervised by “robobosses”

DoD is increasingly employing autonomous capabilities across a diverse number of systems

Source: DSB Study – June 2016
Autonomous Systems in Use Today Are the Result of Decades of R&D

R&D areas include:
- digitization of sensors
- adaptive algorithms
- natural user interfaces
- machine learning
- machine vision
- data analytics

Source: SEI
Impact of Increasing Software-Intensive Autonomous Systems

- Emergent behavior
- Continuous and asynchronous delivery
- Continuous system evolution
- Hard-to-define system boundaries
- Human-machine interface issues
- Data-rich environment
- Growing gap between information obtained using traditional project measures and project managers information needs
Increasing Complexity of Cybersecurity Systems

Complexity and How We Interpret It Are Key Drivers in Measurement

DoD must be able to operate
• between layers
• between networks
• between domains
• between environments

Source: Kenneth R. Turner
Dep. Director, Spectrum Policy and International Engagements
DoD Chief Information Officer

DoD must be able to operate
• between layers
• between networks
• between domains
• between environments

Source: Kenneth R. Turner
Dep. Director, Spectrum Policy and International Engagements
DoD Chief Information Officer

UNCLASSIFIED
Satisfying Unique Operational Mission and Business Needs

Practical measures for a complicated world:
Reassessment of information measures

Source: SEI
Solving the Vulnerability Identification Chasm

First line of defense in software assurance is the application (software) layer: Classic Issue – Tackling action on the measurement data we have

84% of breaches exploit vulnerabilities in the application

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

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

Addressing System Sustainment

Software development and sustainment require planning:
Applying software assurance (SwA) measures across SDLC

Break point where software is handed off for sustainment is increasing blurred

Involves coordinating processes, procedures, people, and information

Challenges include
• rising costs
• Recertification/retesting
• dynamic operating environments
• legacy environments
• SDLC SwA measures

Source: SEI
# Handling the Expanding Code Base

Software is dramatically expanding with limited natural governance: Finding measures that scale is a bigger issue

<table>
<thead>
<tr>
<th>Size of Codebase (SLOC)</th>
<th>Millions of Source Lines of Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Healthcare.gov</td>
<td>Google = 2,000 MSLOC</td>
</tr>
<tr>
<td>Debian 5.0 (all software in package)</td>
<td></td>
</tr>
<tr>
<td>Software in typical new car, 2013</td>
<td></td>
</tr>
<tr>
<td>Mac OS X 10.4</td>
<td></td>
</tr>
<tr>
<td>Debian 5.0 codebase</td>
<td></td>
</tr>
<tr>
<td>US Army’s Future Combat System</td>
<td></td>
</tr>
<tr>
<td>Facebook (without backend code)</td>
<td></td>
</tr>
<tr>
<td>Windows Vista (2007)</td>
<td></td>
</tr>
<tr>
<td>Microsoft Visual Studio 2012</td>
<td></td>
</tr>
<tr>
<td>Large Hadron Collider</td>
<td></td>
</tr>
<tr>
<td>Microsoft Office (2013)</td>
<td></td>
</tr>
<tr>
<td>Windows XP (2001)</td>
<td></td>
</tr>
<tr>
<td>Windows 7</td>
<td></td>
</tr>
<tr>
<td>Symbian</td>
<td></td>
</tr>
<tr>
<td>Microsoft Office for Mac (2006)</td>
<td></td>
</tr>
<tr>
<td>Microsoft Office (2001)</td>
<td></td>
</tr>
<tr>
<td>F-35 Fighter</td>
<td></td>
</tr>
<tr>
<td>Apache Open Office</td>
<td></td>
</tr>
<tr>
<td>Linux 3.1 (recent version, 2013)</td>
<td></td>
</tr>
<tr>
<td>Android (upper estimate)</td>
<td></td>
</tr>
<tr>
<td>Boeing 787, total flight software</td>
<td></td>
</tr>
</tbody>
</table>

Source: David McCandless, “Information is Beautiful,” 21 September 2016 web retrieval
Understanding Attack Patterns, Vulnerabilities, and Weaknesses
Defining cybersecurity measurements to satisfy information needs

- "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.

Source: Bob Martin, MITRE
CVE 1999 to 2017: Reported Common Vulnerabilities and Exposures (CVE)

What to Measure

CWEs  Zero Day

Source: Dr. Robert A. Martin, MITRE Corporation, May 2017
Example: Software Quality Measurement over the SDLC

Improving architectural design and measurement over the SDLC

Where Software Flaws Are Introduced

<table>
<thead>
<tr>
<th>Requirements Engineering</th>
<th>System Design</th>
<th>Software Architectural Design</th>
<th>Component Software Design</th>
<th>Code Development</th>
<th>Unit Test</th>
<th>Integration</th>
<th>System Test</th>
<th>Acceptance Test</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

70%  20%  10%

Where Software Flaws Are Found

<table>
<thead>
<tr>
<th>Requirements Engineering</th>
<th>System Design</th>
<th>Software Architectural Design</th>
<th>Component Software Design</th>
<th>Code Development</th>
<th>Unit Test</th>
<th>Integration</th>
<th>System Test</th>
<th>Acceptance Test</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

3.5%  16%  50.5%  9%  21%

Modern development and testing tools will be critical

Sources: Critical Code; NIST, NASA, INCOSE, and Aircraft Industry Studies
Example: Reducing Technical Debt

Measures of accumulating software assurance technical debt?

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

May Not Be Right Direction if Pushing Costs to Sustainment

Right Direction if Better Programs


Example: Reducing Technical Debt

Measures of accumulating software assurance technical debt?

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

May Not Be Right Direction if Pushing Costs to Sustainment

Right Direction if Better Programs


Example: Reducing Technical Debt

Measures of accumulating software assurance technical debt?

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

May Not Be Right Direction if Pushing Costs to Sustainment

Right Direction if Better Programs


Example: Reducing Technical Debt

Measures of accumulating software assurance technical debt?

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

May Not Be Right Direction if Pushing Costs to Sustainment

Right Direction if Better Programs

Example: Measuring Technical Debt - the Expanding Code Base - OSD Weapon Systems

Engineering-in Software Assurance Over the Lifecycle

Engineering Artifacts Provide Opportunities for More Integrated Measurement
## Working in the Infancy of the Software Engineering Discipline

Improving the workforce by developing software core competencies and career field

<table>
<thead>
<tr>
<th></th>
<th>Physical Science</th>
<th>Bioscience</th>
<th>Computer/Software/Cyber Science</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origins/History</strong></td>
<td>Begun in antiquity</td>
<td>Begun in antiquity</td>
<td>Mid-20th century</td>
</tr>
<tr>
<td><strong>Enduring Laws</strong></td>
<td>Laws are foundational to furthering exploration in the science</td>
<td>Laws are foundational to furthering exploration in the science</td>
<td>Only mathematical laws have proven foundational to computation</td>
</tr>
</tbody>
</table>
| **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
Example: 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 measurement approach will be needed…
- Decision makers will need insight and understanding about the meaning of the data
- 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 “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 analytics based framework that seeks to leverage traditional measurement, metrics, data, and information - if done right will provide meaningful understanding and insight
Final Thought: Measurement of Advanced Program Management, With Operational Participation,
Will determine if we create C3PO and Johnny 5 . . .

Source: SEI
…or the Borg

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