TSP and Security

PSP/TSP Community of Practice
Breakout Group

December 14-15, 2016

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Topics

Problem

Build Security In Maturity Model (BSIMM)

Composing Effective Security Assurance Workflows (CESAW)

Architecture Analysis Design Language (AADL)

Motivation

References

Civilization's Software Challenge

The volume of software in the world is increasing exponentially.

Approx. 1% to 5% of these defective lines of code are potential safety faults or security vulnerabilities.

There is increasing concern for the mounting technical debt and risk that this represents.

As software users, managers, or developers we would like to have some indication of the level of trust or confidence that the software we are developing or using functions as intended and is free of vulnerabilities.

Software Assurance: 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. **[DoDI 5200.44]**

The Chasm between Development and Operations

Two "communities of practice" with few abstractions, vocabulary, methods, measures, and tools in common

	Communities	
	Software Development	***Operations***
Motivation	Deliver complete product on time	The mission
Language used for "what can disrupt the work"	Defects	Faults, Vuls
Costs of disruption	Rework, opportunity cost, reputation	Mission failure, cost of patching, diversion of attention and resources
Means to address disruption	Process (prevention, testing, etc.)	Workarounds, patches

The TSP COP must find a way to talk about TSP with Operations based on their motivation and using their vocabulary.

But We Both Have Data …

Examples of data collected by the SEI. NIST and MITRE collect even more data on vulnerabilies and weaknesses.

TSP data

- **approximately 10,000 defect records**
- **for each defect**
  - **find and fix effort**
  - **phase injected**
  - **phase removed**
  - **ODC type**
  - **an explicit description of the change made**
  - **if it was injected while fixing another defect**
- **development effort for design**
- **effort for design review**

CERT data

- **over 40,000 vuls cases in a number of Lotus Notes databases**
- **vuls are not categorized or classified in any particular way**
- **for each vul**
  - **descriptive title**
  - **tracking number**
  - **date when reported**
  - **quick first order estimate of how many systems affected (impact)**
  - **list of vendors affected**
  - **CVSS ratings**
  - **URLs with additional info about vul, particularly those that are publicly known**

Composing Software Assurance

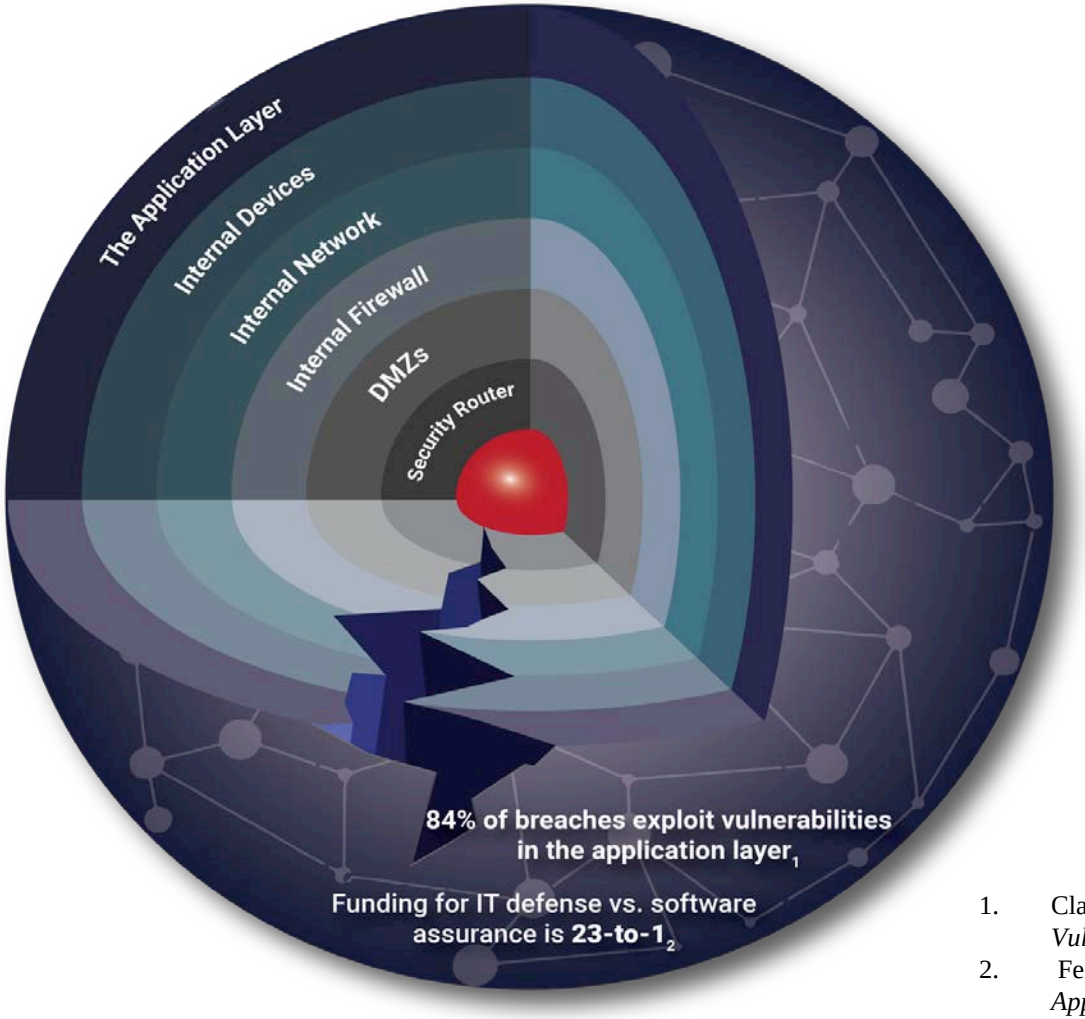
There "*is inadequate ground truth information to help [DoD] make decisions*", for example the true cost, schedule impact, and effectives of integrating these tools into an SDLC.

Problem:

DoD faces an unfunded Congressional mandate to secure software from attack. This has been implemented by requiring the use of software security assurance (SwA) tools on "all covered systems".

Without empirically validated "ground truth" about cost and effectiveness.

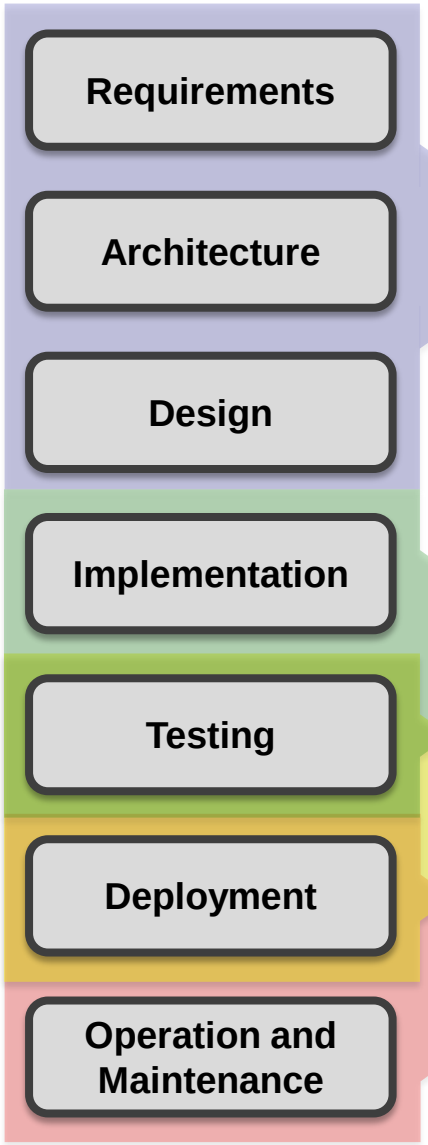
The CORE of Cybersecurity is the Application (Software) Layer



- **The Application Layer is the new perimeter**
- **SwA must be Engineered into the Lifecycle of Applications**

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. G00269825

Spanning the Lifecycle to Prevent/Mitigate Vulnerabilities



Architecture/Design Initiatives

- AADL
- Design fewer new vulnerabilities

Secure Coding Initiative

- Implement fewer new vulnerabilities

Vulnerability Discovery

- Find vulnerabilities before being deployed

Vulnerability Response

- Reduce deployed vulnerabilities

Call to action, COP

Include BSIMM practices into TSP Secure

The Software Security Framework (SSF)			
Governance	**Intelligence**	**SSDL Touchpoints**	**Deployment**
Strategy and Metrics	Attack Models	Architecture Analysis	Penetration Testing
Compliance and Policy	Security Features and Design	Code Review	Software Environment
Training	Standards and Requirements	Security Testing	Configuration Management and Vulnerability Management

- Four domains
- Twelve practices

Problem

Monkeys eat bananas

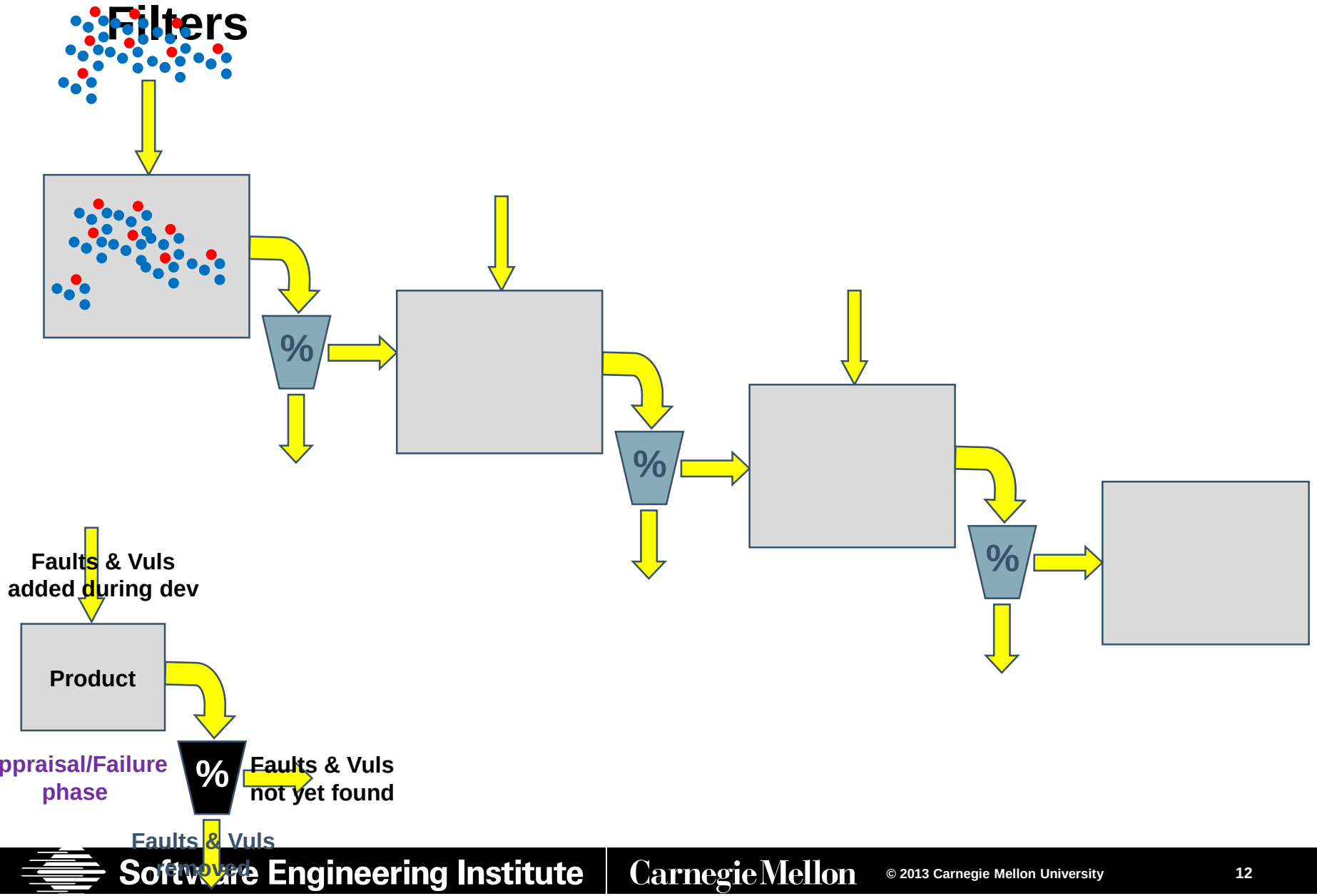
- BSIMM is not about good or bad ways to eat bananas or banana best practices
- BSIMM is about observations

cigital

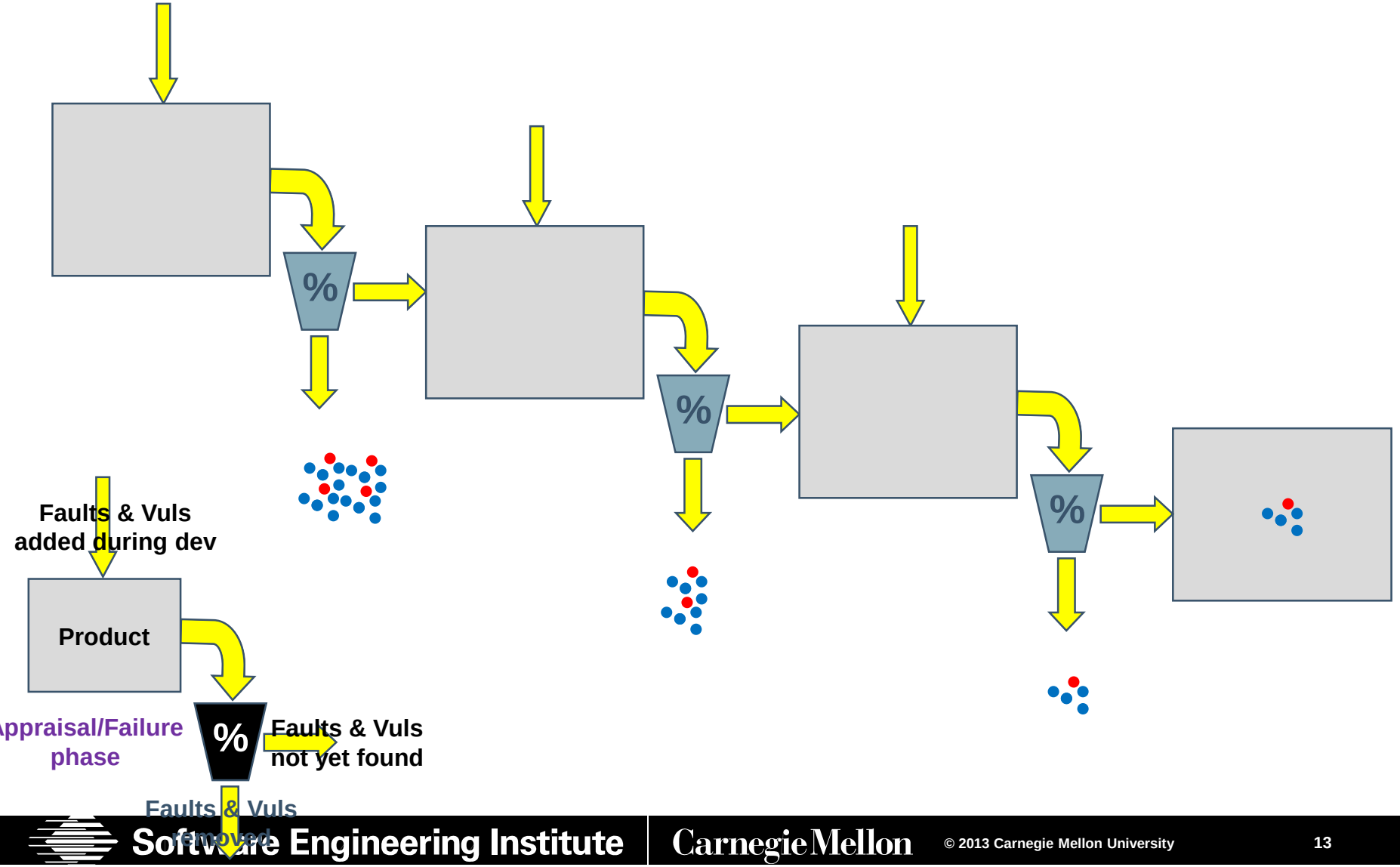
Build Security in Maturity Model (BSIMM)

"data" is counts of practices of unknown value and cost

Faults & Vulnerabilities Removal Tanks & Filters



Faults & Vulnerabilities Removal Tanks & Filters



Composing Effective Security Assurance Solutions
CESAW

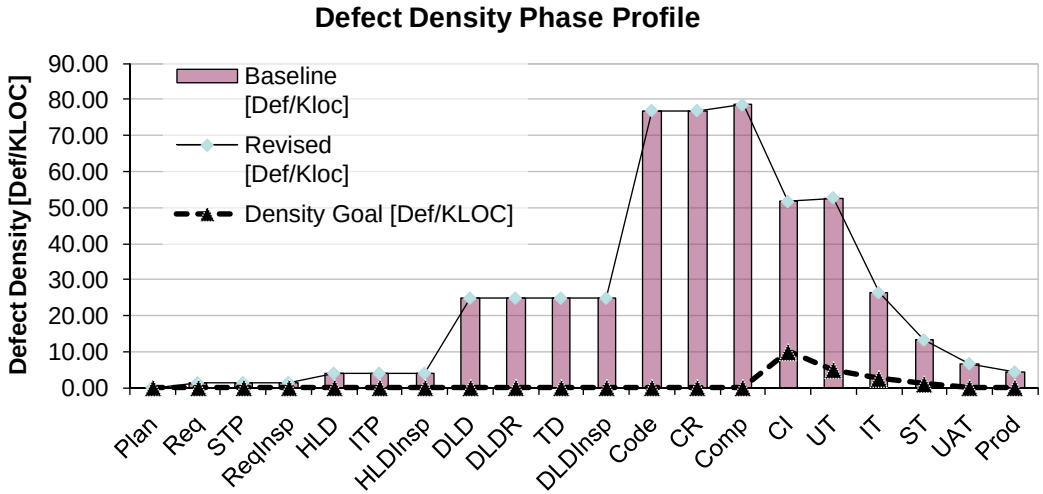
- Apply experience and toolkit from quality assurance to security assurance
- Develop a model that predicts the effectiveness and cost of selected SwA tools.
- Use the model and data to help DoD integrate SwA tools into SDLC processes

Tool kit includes

1. A metric framework
2. Quality planning
3. Design
4. Inspections
5. Process composition for additional techniques

CESAW, extend to model performance

Control Panel	Rate [LOC/hr]	Yield (per insp)	Yield (total)	# Insp	Effort [hr]
Design Review	200	50.0%	**0.0%**	0	0.0
Design Inspection	200	50.0%	**0.0%**	0	0.0
Code Review	200	50.0%	**0.0%**	0	0.0
Code Inspection	200	50.0%	**0.0%**	0	0.0



Compare your performance to a baseline.

Metrics Needed to “Engineer” a System

Defect Density (throughout lifetime)

Vulnerability Density (found at each stage)

Phase Injection Rate [defects/hr.] (derived)

Phase Effort Distribution (effort-hr.)

Phase Removal Yield [% removed] (effectiveness)

Defect “Find and Fix” time [hr./defect] (what was found)

Defect Type (categorize what was wrong)

Defect Injection/Removal Phases

Zero Defect Test time [hr.] (cost if no defects present)

Product Size [LOC] [FP] (for normalization and comparisons)

Development Rate (construction phase) [LOC/hr.]

Review/Inspection Rate [LOC/hr.] (cost of human appraisal)

Parameters Needed to “Engineer” a System

Phase Injection Rate [defects/hr]

Phase Effort Distribution [%] total time

Size [LOC]

Production Rate (construction phase) [LOC/hr]

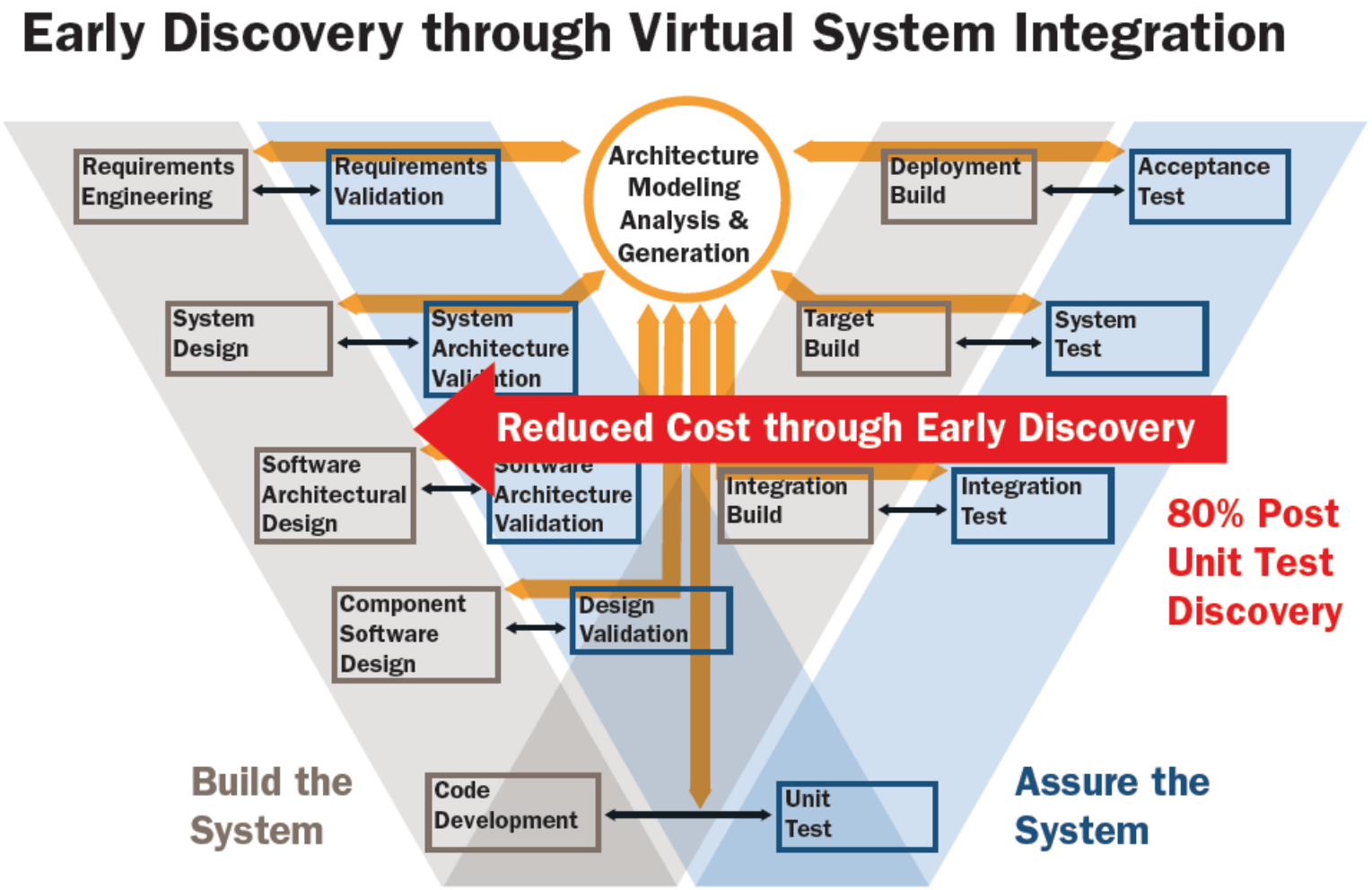
Phase Removal Yield [% removed]

Zero Defect Test time [hr]

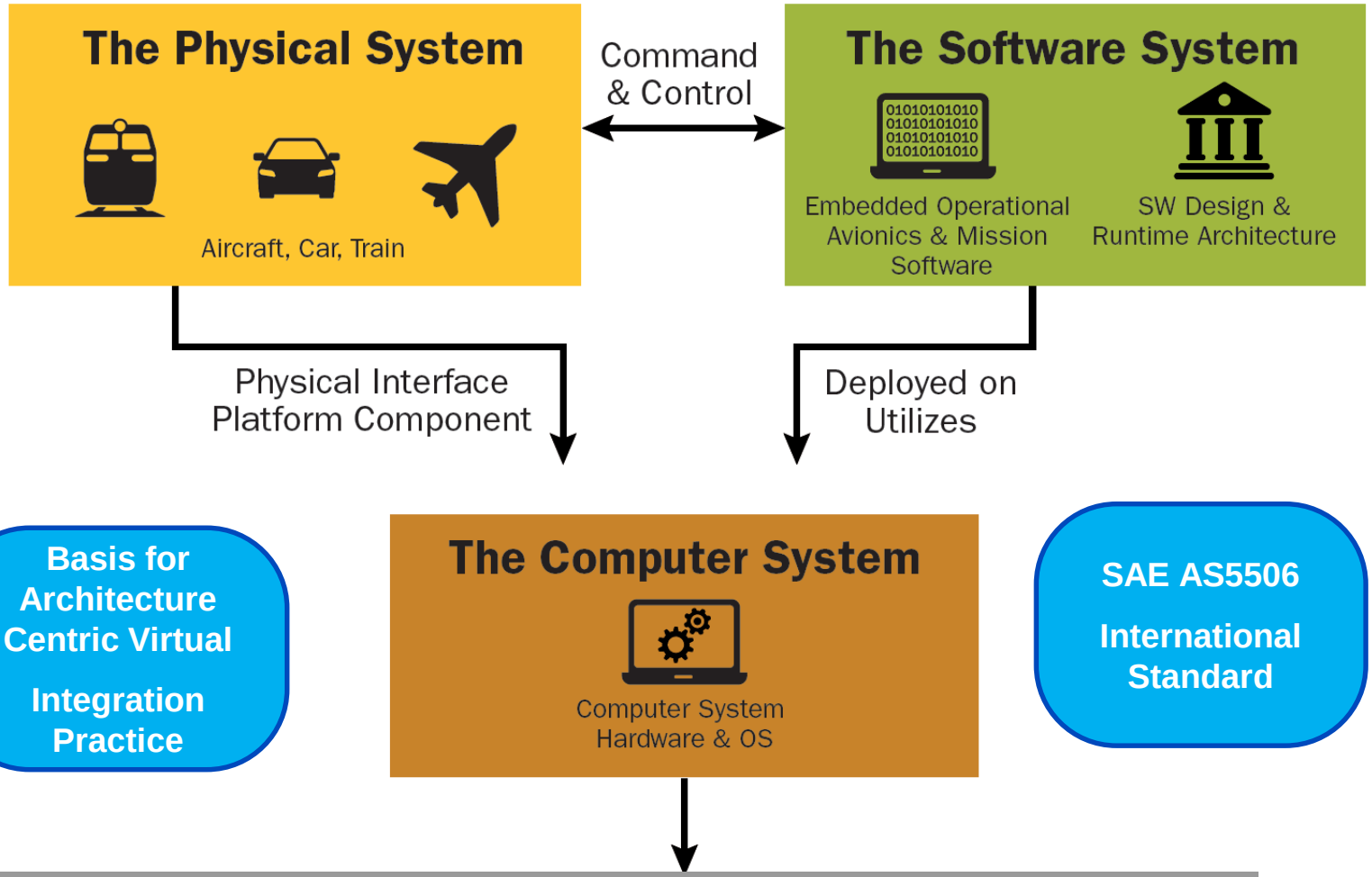
Phase “Find and Fix” time [hr/defect]

Review/Inspection Rate [LOC/hr]

Incremental Lifecycle Assurance



Architecture Analysis & Design Language (AADL) Enables Industry-Wide Virtual Integration and Assurance Approach



Motivation for SW Developers

- ✓ Non-disiplined software process rewards "heroism".

  … this motivation has to change
- ✓ Advocates of well engineered software process create wonderful product making no noise, **WE NEED to built up pride of doing right !!**

One simple strategy could be.

1) Any "defect" found in Unit Test or beyond is a fault (shoudn't happend, something was wrong)
2) Any "refinement" discover before Unit Test IS RIGHT, just following a "Well Engineered" Process to finally create the desired RIGHT artifact. Shouldn't be called a "defect", IT IS NOT, it is just good engineering thinking.

What is the difference??

- ✓ "refinement" IS RIGHT, we should "be pride", as an engineer, I'm going on track, my "ego" should not suffer.
- ✓ "defect" SOMETHING WAS WRONG, as an engineer I failed, it is a LEARNING OPPORTUNITY

A software ENGINEER should be proud of:

a) Exercise RIGOROUS test, …. **discover 0 defect**

b) When something was wrong, getting the most of that **LEARNING OPPORTUNITY**, I am moving in the right direcction to became a **good engineer**

Who do we talk to?

Example of non-TSP people we talk to

(from a view point of Academia in Japan)

Security professors:

- We have an increasing number of security departments, which might be teaching secure software development.
- However, they don’t (cannot) teach process matters.

Engineers/managers

- For systems not connected to the Internet before.
- …

References

SEI Vul Database: http://www.kb.cert.org/vuls/

National Vul Database: https://nvd.nist.gov

Common Vulnerability and Exposures Database (CVE): http://cve.mitre.org

Common Weakness Enumeration (CWE): http://cwe.mitre.org/index.html

Cybersecurity Framework: https://www.nist.gov/cyberframework

BSIMM: https://www.bsimm.com

SEI has blogs, webinars, cybersecurity minute, podcasts, conferences, etc. that can be utilized as communication vehicles for TSP COP.

Questions?