Software Engineering Institute | Carnegie Mellon University

Architecture Analysis & Design Language and System Verification

Software Solutions Division

When a system fails, engineers too often focus on the physical components, but pay scant attention to the software. In software-reliant systems, deemphasizing the importance of software failures can lead to disaster. The Architecture Analysis & Design Language (AADL) standard can help alleviate mismatched assumptions between the hardware, software, and their interactions that can cause system failures.

There are many documented examples of problems in software-reliant systems:

- In December 2012, Ford Motor Company announced that it would be making software updates to the cooling systems of its hybrid 2013 Ford Escape and Ford Fusion because problems with the original cooling system design caused the vehicle to catch fire while the engine was running.
- A recent study of a popular automatic external defibrillator found security flaws in the embedded software and the commercial, off-the-shelf update mechanism.
- A cluster of ships attached to the Ariane 5 rocket was lost when the rocket failed to achieve orbit on its maiden voyage. The failure was attributed to an error in software design and resulted in a loss of more than $370 million.

These incidents demonstrate the need to make software safer and more reliable.

Addressing Software (as Well as Hardware)

Mismatched assumptions between hardware, software, and their interactions often result in system problems that are caught too late, which is an expensive and potentially dangerous situation to developers and users of mission- and safety-critical technologies. To address this problem, the Society of Automotive Engineers (SAE) released the aerospace standard AS5506, named the AADL. The AADL standard, which was authored by my colleague, Peter Feiler, defines a modeling notation based on a textual and graphic representation that is used by development organizations to conduct lightweight, rigorous—yet comparatively inexpensive—analyses of critical real-time factors, such as performance, dependability, security, and data integrity. AADL models capture both software and hardware components, as well as their interactions, including the association of a software process on a processor or the deployment of connection on a bus.

The AADL standard includes abstractions of software, computational hardware, and system components for

- specifying real-time, embedded, and high-dependability systems with their software/hardware concerns and specific requirements (such as scheduling, bus latency, or jitter) systems
- validating systems and ensuring that stakeholders' requirements can be achieved

Military and Civil Applications

While initial AADL efforts were experimental, both military and civil organizations have been using the AADL standard for nearly a decade. An early adopter was the U.S. Army Aviation and Missile Research, Development, and Engineering Center, the organization that provides R&D and engineering technology for aviation and missile platforms across the program lifecycle. Using AADL, AMRDEC has performed quantitative analysis of a system architecture—both software and hardware—through virtual integration to determine if the system could meet safety and other requirements before building it.

Another early adopter was the European Space Agency, which leads the ASSERT project for the design and implementation of safety-critical systems. This project has relied on AADL since its inception, and project members use it to model, validate, and produce software. From this early use of the system, other projects and communities have shown a strong interest in adopting and applying the language.

A Global Proof of Concept

In 2008, the Aerospace Vehicle Systems Institute (AVSI), a global cooperative of aerospace companies, government organizations, and academic institutions, launched an international, industry-wide program called System Architecture Virtual Integration (SAVI) to reduce cost/cycle time and rework risk by using early and frequent virtual integrations. In addition to the SEI, major players of the SAVI project include Boeing, Airbus, Lockheed Martin, BAE Systems, Rockwell Collins, GE Aviation, the Federal Aviation Administration, the U.S. Department of Defense, Honeywell, Goodrich, United Technologies, and NASA.

Architecture Analysis & Design Language and System Verification

SAVI was created in response to the realization by aircraft manufacturers that the source lines of code (SLOC) in software-reliant systems were increasing exponentially. For the past 20 years, the size of aircraft software, measured in SLOC, has doubled every 4 years. For the next decade, the projected 27.5 million lines of code required could cost more than $10 billion. In addition to the increased software cost, the latest generation of aircraft, such as the Boeing 787s, will be highly interconnected and is projected to produce half of terabyte of data per flight.

Coupled with increasing complexity is a growing reliance on embedded software to meet requirements for key features, such as greater range and comfort on fewer, more economical flights. SAVI intends to pilot new technology and acquisition processes based on architectural models, rather than paper documentation, with multiple dimensions of analysis used throughout the lifecycle.

A key tenet of virtual integration is the use of an annotated architecture model as the single source for analysis. For the SAVI proof of concept, an aircraft system architecture was modeled using the AADL standard. This process allowed AVSI to capture integration faults earlier in the development process and meet increasing demands for safety and economy.

The National Institute of Standards & Technology estimated in the report *The Economic Impacts of Inadequate Infrastructure for Software Testing* that more than 70 percent of all system defects are introduced prior to code development, while only a small fraction are detected during that time. The remaining errors are detected in the testing phase when they are most expensive to fix. The ability to detect errors and defects prior to implementation efforts therefore reduced development costs and efforts, while enhancing overall system robustness and reliability.

The SEI technical report *System Architecture Virtual Integration: An Industrial Case Study* details the SAVI proof of concept and results, which include

- multi-tier modeling and analysis of an aircraft and its subsystems
- support for the needs of both system engineers and embedded software system engineers
- propagation of changes to the model across multiple analysis dimensions
- maintenance of multiple model representations in a model repository
- auto-generation of analytical models
- interfacing of multiple tools
- distributed team development via a model repository

New Directions for AADL

We are developing a workbench that demonstrates measurable reduction in cost of verifying system implementations against requirements through incremental lifecycle assurance by

- assuring hazard and derived requirement coverage during architecture design iterations
- reducing verification-related rework as measured by increased early defect detection
- providing a measure of confidence throughout the lifecycle by tracking requirement quality and verification results and auto-generating assurance-case artifacts

Additional Resources

Webinars

Architecture Analysis with AADL. Introduces AADL, the architecture modeling language used to specify safety-critical systems. https://www.webcaster4.com/Webcast/Page/139/5357

An Architecture-centric Virtual Integration Strategy to Safety-Critical System Verification. Elaborates on incremental life cycle assurance for critical systems. https://www.csiac.org/event/architecture-centric-virtual-integration-strategy-safety-critical-system-verification

Wiki and Source Code

With our user community, the SEI maintains a wiki with all of our research on AADL that is accessible to the public. http://www.aadl.info/aadl/currentsite/

To access a site that maintains source code for AADL, where people can come and freely contribute to the OSATE tool set that is supporting AADL, please visit https://github.com/osate

Book

Model-Based Engineering with AADL: An Introduction to the SAE Architecture Analysis & Design Language, by Peter Feiler and David Gluch. http://resources.sei.cmu.edu/library/asset-view.cfm?assetID=30284

Reports

System Architecture Virtual Integration: An Industrial Case Study details the SAVI proof of concept and results. http://resources.sei.cmu.edu/library/asset-view.cfm?assetID=9145

Four Pillars for Improving the Quality of Safety-Critical Software-Reliant Systems presents an integrate-then-build practice to improve quality through early defect discovery and incremental verification. http://resources.sei.cmu.edu/library/asset-view.cfm?assetid=47791

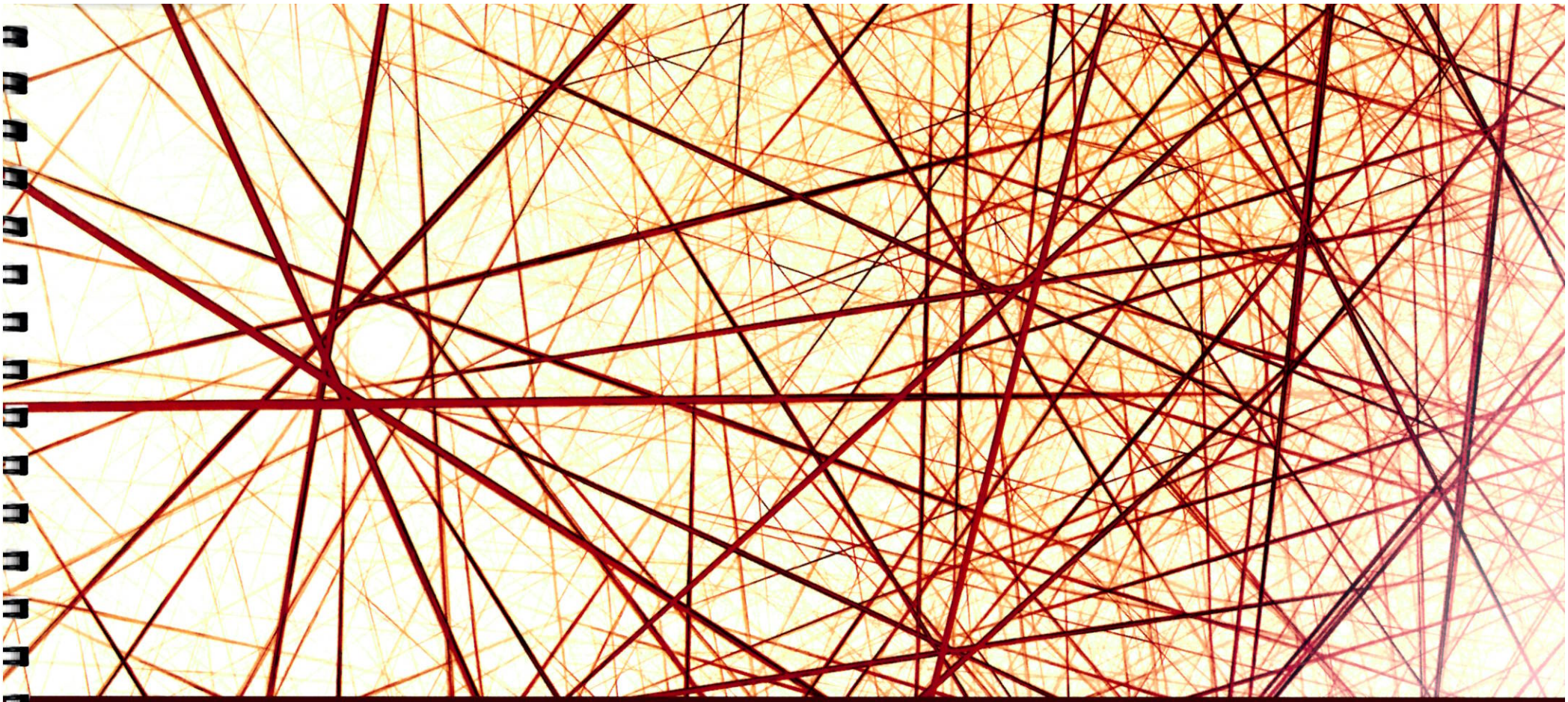
For Course Registration

http://www.sei.cmu.edu/training/p72.cfm

For More Information

Contact Customer Relations
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15313-2612
Phone: 412-268-5800
info@sei.cmu.edu

9/29/14

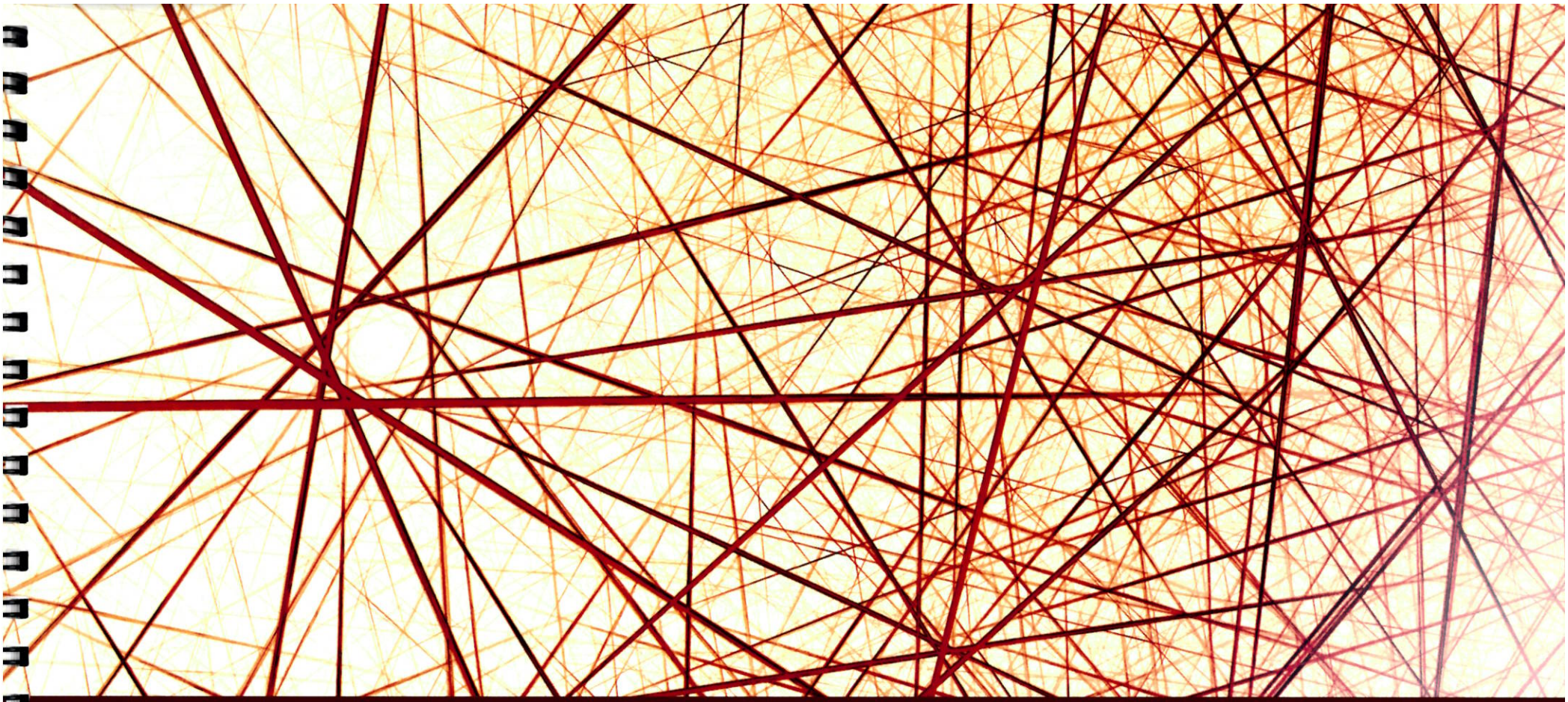


Fall 2014 SEI Research Review

October 28–30, 2014

Summaries of Projects, Presentations, Workshops, and Posters

Software Engineering Institute | Carnegie Mellon University



Fall 2014 SEI Research Review

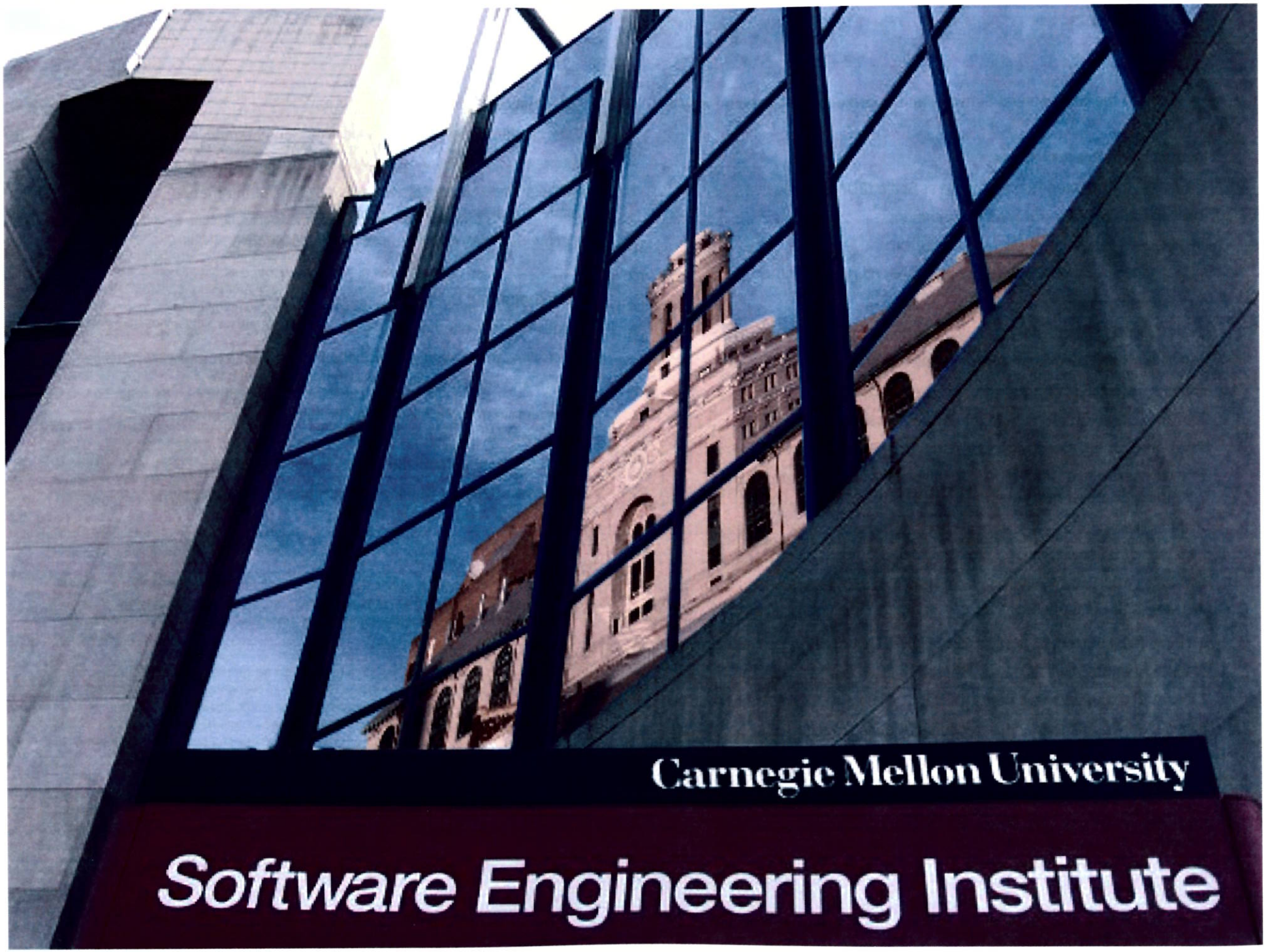
October 28–30, 2014

Summaries of Projects, Presentations, Workshops, and Posters

Software Engineering Institute | Carnegie Mellon University

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

CONTENTS


Carnegie Mellon University
Software Engineering Institute

Welcome to the SEI's Annual Software and Cybersecurity Research Review

Kevin Fall, PhD

Deputy Director,
Research, and CTO

Carnegie Mellon University
Software Engineering Institute

kfall@cmu.edu

Welcome to Carnegie Mellon University and the Software Engineering Institute (SEI). Our Research Review is intended to bring together the government, academic, and industrial communities with whom we work and interact to highlight our research activities.

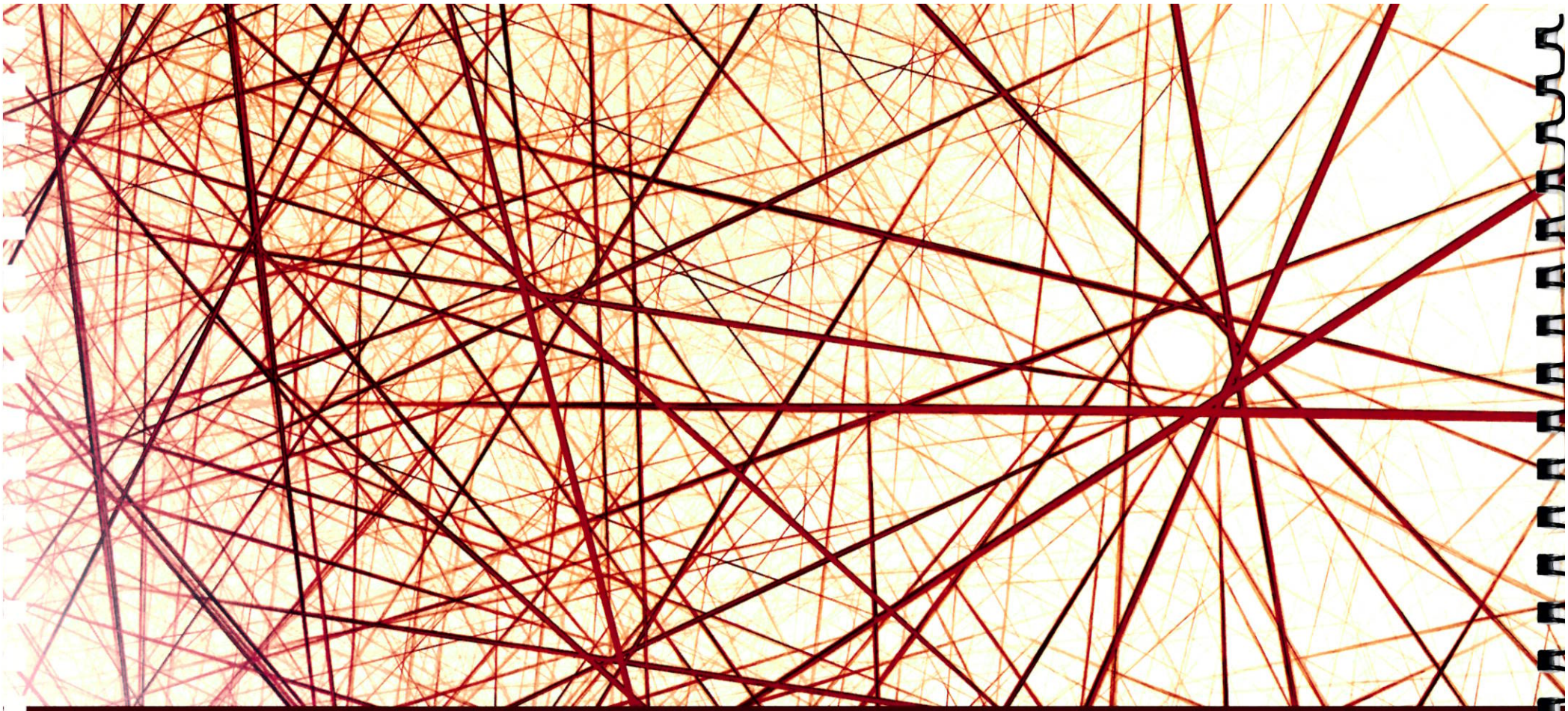
The SEI is a federally funded research and development center (FFRDC) sponsored by the U.S. Department of Defense. We endeavor to apply the best combination of thinking, technology, and methods to the most deserving government software-related problem sets, free from conflict of interest.

While other FFRDCs and research centers are also attentive to the government's problems, the SEI brings its unique combined capabilities in cybersecurity and software together with its university affiliation and industry access to bear on important and challenging software-related problems—in acquisition, development, testing, security, safety, operations, and sustainment.

To provide the capabilities it offers, the SEI's workforce maintains expertise in the following technical areas: software and systems engineering, cybersecurity and software assurance, computer science, applied mathematics, measurement and analysis, and acquisition of software-reliant systems. The SEI's work products include research reports, methods, software prototypes, and educational courses.

This booklet contains summaries of the research projects comprising the SEI's research portfolio, in addition to interactive workshops and other activities at the SEI. We encourage you to reach out to the authors, presenters, and other members of the SEI's staff for additional information, discussion, and future collaboration opportunities.

Thank you.



Software Engineering Institute
Carnegie Mellon University
4500 Fifth Avenue
Pittsburgh, PA 15213-2612

SEI Washington, DC
Suite 200
4301 Wilson Boulevard
Arlington, VA 22203

SEI Los Angeles, CA
2401 East El Segundo Boulevard
El Segundo, CA 90245