Carnegie Mellon University
Software Engineering Institute

INNOVATIONS

History of Innovation at the SEI

SECOND EDITION

40 *years*

ADVANCING SOFTWARE *for* NATIONAL SECURITY

INVOTIS

Advancing Software for National Security

FOR THE PAST 40 YEARS, the Software Engineering Institute (SEI) has worked beyond the leading edge to ensure that the U.S. Department of Defense (DoD) is prepared to respond with new software capabilities to warfighter needs before they appear on the horizon. The SEI, established by the DoD at Carnegie Mellon University (CMU) in 1984, has advanced software as a strategic advantage for national security since day one and has become essential in the DoD software ecosystem by leading the development of software architecture practices; inventing and prototyping new technologies; helping DoD programs adopt modern software practices; sharing knowledge across government, industry, and academia; and bringing about measurable improvement in DoD software acquisition, software development, system operation, and sustainment.

Building on the academic foundation provided by CMU, the SEI quickly became a research leader. Its technical work often produces proven engineering solutions that benefit defense and national security systems years later. For example, SEI pioneering work in software architecture in the 1990s led to the accepted understanding today that architecture determines the quality, performance, and longevity of a software system. In the last decade, the SEI has advanced artificial intelligence (AI) from bespoke solutions and isolated algorithms toward an AI Engineering discipline and an AI system development lifecycle.

Yesterday and today, the SEI shows that it both anticipates and responds to DoD challenges efficiently, effectively, and thoroughly through excellence in research, development, and deployment (RD&D) and technology transition practices. Tomorrow, when the DoD seeks answers to incorporate quantum computing, greater automation, more and better approaches for cybersecurity, and to address a host of other over-the-horizon needs, the SEI will be there, too, continuing to ensure that software is a strategic advantage for the DoD.

Contents

AI for Autonomy Lab

The Department of Defense (DoD) has long recognized the potential benefits of using autonomous systems for mission success. Over the last 10 years, there have been advances in artificial intelligence (AI) and machine learning (ML) technology for autonomous systems. The DoD has intensified its autonomy research and development programs in the Army, Navy, and Air Force. Our adversaries have also pursued the promise of autonomous technology. The DoD's interest has continued to grow, and in 2023, it announced Directive 3000.09, *Autonomy in Weapon Systems,* a commitment to military uses of autonomous systems, AI, and ML.

To support this growing interest and commitment, the SEI formally established the AI for Autonomy Lab in late 2023 and, in 2024, began focusing on sponsored work and expanding its relationships with DoD agencies, academic research groups, federally funded research and development centers (FFRDCs), university-affiliated research centers, and vendors of AI and autonomy solutions. This lab was established to enable expert researchers to study and demonstrate how AI and ML technologies can be used to improve the performance of autonomous systems while meeting warfighter needs for trustworthiness and accountability. The AI for Autonomy Lab

2024

partners with leading-edge organizations, including the Carnegie Mellon University Robotics Institute's AirLab and National Robotics Engineering Center.

The AI for Autonomy Lab team includes experts in AI and ML; robotics; software engineering; test and evaluation; national security; cybersecurity; and vehicles that operate across land, sea, air, and space. The lab works with DoD stakeholders and partners to apply AI and ML to autonomy challenges such as planning and control, simulation methods, and evaluation practices—challenges that the DoD must meet to successfully deploy and operate these solutions in the field.

Looking ahead, the lab continues to grow its relationships with DoD departments and agencies, academic research groups, FFRDCs, university-affiliated research centers, defense industrial base contractors, and vendors of AI, ML, and autonomy solutions. These relationships enable the lab to have a broad impact on autonomous systems, improving their performance, capability, reliability, effectiveness, and suitability.

> In 2023, the SEI formally established the AI for Autonomy Lab to enable expert researchers to study and demonstrate how AI and ML technologies can be used responsibly in autonomous systems.

Photo: U.S. Air Force

Artificial Intelligence Security Incident Response Team (AISIRT)

In November 2023, the Software Engineering Institute (SEI) developed the first Artificial Intelligence Security Incident Response Team (AISIRT) to respond to the risks associated with artificial intelligence (AI) that can pose a threat to national security. The AISIRT identifies, analyzes, and responds to the threats, vulnerabilities, and incidents that emerge from the ongoing advances in AI and machine learning (ML) and supports the Department of Defense (DoD) and other federal agencies in effectively and securely developing, adopting, and using AI.

The SEI, as a national leader in cybersecurity and coordinated vulnerability disclosure (CVD), leverages its decades of expertise to strengthen both the work and operations of the AISIRT.

Through the AISIRT, the SEI has—amongst other things—worked with the developers of a large language model (LLM) to prevent time-based jailbreak attacks after a jailbreak vulnerability was reported; helped address a GPU API vulnerability related to GPU memory management; and collaborated with a vendor to implement security measures to mitigate vulnerabilities related to remote code execution via prompt injection.

Learn more about the AISIRT at insights.sei.cmu.edu/projects/aisirt-ensures-the-safety-of-ai-systems

Photo: U.S. Space Command

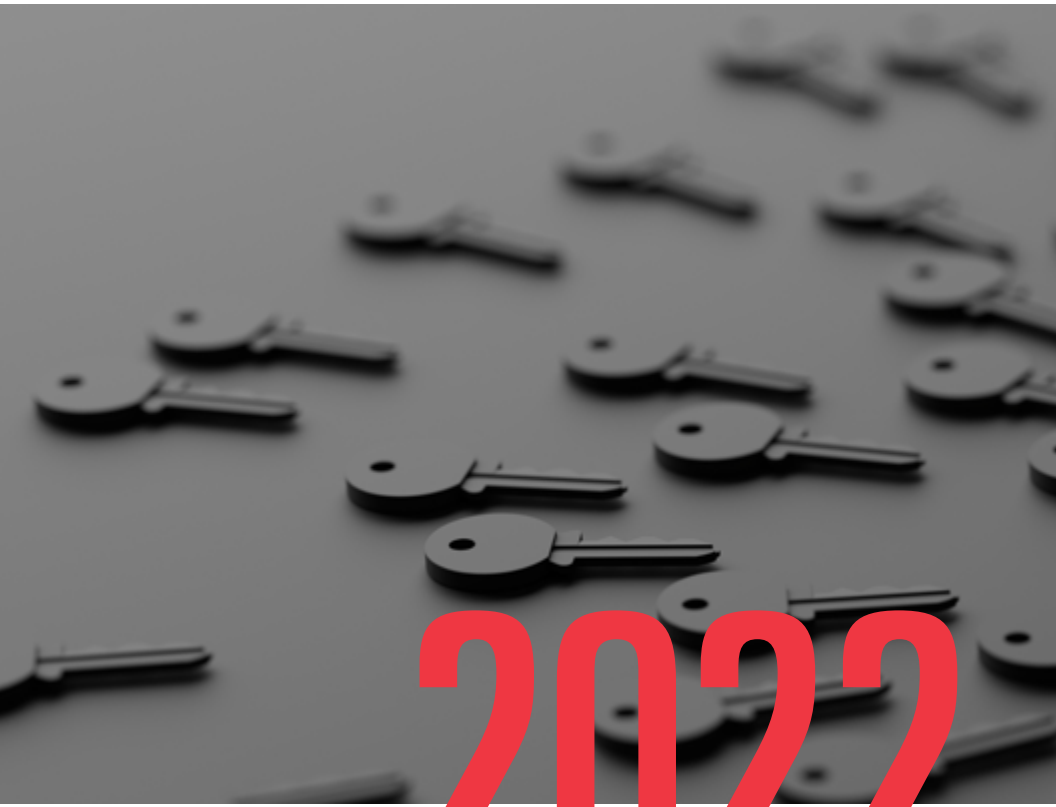
2023

DevSecOps Platform-Independent Model

Due to their unique software system capability requirements, many organizations in highly regulated environments face challenges implementing DevSecOps while ensuring that adversaries cannot abuse weaknesses in the pipeline. Enter the SEI's DevSecOps Platform-Independent Model (PIM), which uses a model-based systems engineering (MBSE) approach to formalize the requirements, capabilities, and maturity of DevSecOps and provide relevant guidance.

This first-of-its-kind model gives software development organizations the structure and articulation needed for creating, maintaining, securing, and improving their DevSecOps posture.

The SEI PIM helps organizations visualize their pipeline infrastructure, decide how to structure the planning process, and ensure that the pipeline and its associated products are implemented in a secure, safe,



2022

and sustainable way. By highlighting their unique strengths and weaknesses, the SEI PIM provides a framework for organizations to recognize appropriate DevSecOps elements in their software development lifecycle, empowering them to choose a customized path to achieve their goals.

The PIM helps organizations improve cybersecurity, providing analysts with a minimum set of MBSE tools to assist with threat identification, analysis, documentation, and subsequent mitigations. The PIM has become a foundational component of SEI DevSecOps-based activities, serving as a reference model for assessments and creating software development documentation.

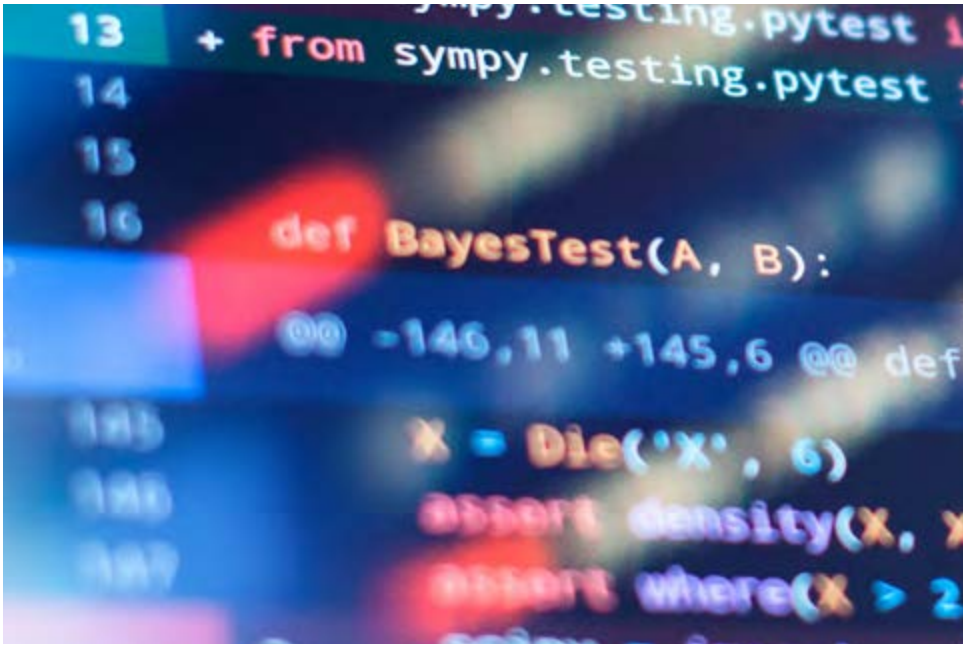
> The SEI PIM is free on the SEI's GitHub site at cmu-sei.github.io/DevSecOps-Model/

SCAIFE

Secure Code Analysis for Continuous Integration

Producing secure software free from serious flaws is a top priority for organizations in defense, government, and industry, and the SEI has a long history of engineering solutions that address this challenge. One way analysts and developers identify flaws in software is through the use of static analysis tools. These tools output alerts that identify potential flaws in code. However, manually adjudicating static analysis results as true positive or false positive can be time consuming and requires expert knowledge. This work also requires a consistent adjudication process. To help relieve developers and analysts of this burden, the SEI created the Source Code Analysis Integrated Framework Environment (SCAIFE), a research prototype for a modular architecture that supports static analysis classification and prioritization. SCAIFE is designed to enable a wide variety of tools, systems, and users to use artificial intelligence (AI) classifiers for static-analysis results (meta-alerts) at relatively low cost and effort.

In 2021, the SEI extended SCAIFE to work with continuous integration (CI) systems. CI is a method of software development in which the working copies from all developers on a software project are automatically merged and shared frequently. CI usually includes an automated suite of tests, which sometimes includes static analysis testing. The CI process helps prevent the code of any one developer from straying too far afield



and averts the potential for a catastrophic merge conflict and merges that introduce bug regressions and/or test failures. Consequently, CI focuses developers to produce stable code that works in an automated build of the software.

SCAIFE for CI works with a range of variations of CI, and it can be geared to the level of testing automation in use in a given development environment. It can also automatically update its static analysis projects and transfer adjudications appropriately on code updated at various frequencies, code produced through several developer threads and automatically merged on a shared server, and/or code updated daily by developers when they commit their code to a code repository server. SCAIFE for CI can also support static analysis testing in development environments that don't yet use a CI server but have generated different versions of the codebase and static analysis to output for these versions. It does so with a graphical user interface process that automates the appropriate transfer of adjudications from one code version project to the other.

> The SEI has made the SCAIFE API definition (using OpenAPI v3, in YAML) available via GitHub at github.com/cmu-sei/SCAIFE-API

2021

2020

Crucible and GHOSTS

Enabling Realistic Cyber Simulations

To empower developers of cyber simulations, the SEI offers some useful tools: Crucible for creating simulated virtual environments and GHOSTS for creating non-player characters (NPCs) within these environments.

Cyber simulations today are frequently developed manually with proprietary systems. Manual methods are time intensive and can introduce human error. Proprietary software creates a dependency on a single vendor, which can result in vendor lock-in and higher costs. In response, the SEI's open source Crucible cyber simulation framework uses open standards and modular application program interfaces (APIs) to deliver low-cost, dynamic virtual environments that maximize interoperability and scalability. Virtual environments in Crucible are deployed using an infrastructure-as-code approach, which enables reuse and iteration. Crucible simulates environments for training labs, team-based exercises, operational tests, and rehearsals. It automates the workflow for creating, deploying, facilitating, and assessing simulations. Developers can reuse existing templates for topologies, scenarios, assessments, and user interfaces.

In cyber simulations, human participants are called players, while non-human participants are called non-player characters (NPCs). In these simulations, players and NPCs interact in realistic contexts and situations. GHOSTS automates and orchestrates NPCs, whose activities produce realistic network traffic. NPCs can range from friendly to hostile. GHOSTS employs machine learning algorithms to combine NPC personas, preferences, and events to influence NPC decisions. The behaviors of GHOSTS NPCs are hard to distinguish from the behaviors of humans and thus defenders struggle to filter out NPC traffic.

Cybersecurity content developers can use Crucible and GHOSTS with their existing tools to create realistic simulations, reduce knowledge gaps within their organizations, evaluate cyber-mission readiness, and cultivate expert cyber teams.

Foundry

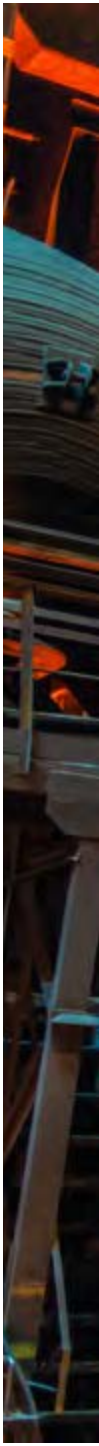
A Training Asset Management Portal

Before 2018, cyber training and exercises were distributed using separate, stove-piped contracts and platforms, which made it difficult to leverage high-quality cyber training and exercises broadly across the DoD. The SEI researched modern platform theory and design to develop Foundry, a next-generation cyber-training asset-management portal. Foundry connects cyber training content users, sponsors, and developers in a shared environment where available content is registered for users to consume, rate, and add to playlists.

Foundry implements the latest industry and DoD standards for interoperability, allowing it to bring together independent, pluggable components. Because of this modular design, training providers can focus on developing innovative content without worrying about hardware stacks or other technical infrastructure. Using Foundry, organizations can integrate different training provider platforms and tools without having to make a long-term commitment to a single provider. This approach encourages innovation and competition among training providers on the platform and motivates the production of increasingly effective cyber training.

Foundry logs and monitors learner ratings and reviews to track the best training available on various topics. Foundry can use those ratings and reviews to better match learners and their teams to the most appropriate training for their roles and responsibilities. Using Foundry, cyber training developers can select from many different content providers to assemble the most realistic, effective training possible, while considering user ratings and reviews to help assess training quality and effectiveness.

Under USCYBERCOM sponsorship, a prototype instance of Foundry was made available in 2018 to anyone issued a DoD Common Access Card (CAC)—seamlessly integrating cyber training content from multiple legacy and next-generation content provider platforms.



2019

2018

Defining the Practice of Managing Technical Debt

From Research to Community

As software systems mature, earlier design or code decisions made in the context of budget or schedule constraints increasingly impede evolution and innovation. This phenomenon is called technical debt, and for a decade, the SEI has been at the forefront of shaping a definition of technical debt, forming and executing a research agenda applicable to government and industry, and cultivating a community of practice.

Since 2010, the SEI has challenged the software engineering research community to find ways to manage technical debt by convening the annual Managing Technical Debt Workshop series. In 2018, the workshops evolved into an annual Conference on Technical Debt co-located with the ACM/IEEE International Conference on Software Engineering. Those events have produced more than 100 publications in the Association for Computer Machinery (ACM)/Institute of Electrical and Electronics Engineers (IEEE) digital libraries and spawned more than 2,000 citations in other published papers.

Through relationships with researchers, practitioners, and industry tool vendors in the software engineering community, SEI researchers have gathered data on technical debt in large-scale systems. They have identified metrics that can be extracted from the code and module structures of software systems. And by combining techniques from machine learning and code analysis, they have provided software engineers visibility into technical debt from strategic and architectural perspectives.

The book *Managing Technical Debt* in the SEI Series in Software Engineering blends this research into a cohesive approach that developers can use to deliberately manage technical debt in their systems. In 2019, the SEI transitioned conference management to this community that continues to share research results, data, and lessons learned about their projects to advance the practice of managing technical debt.

Helping Analysts Automate Reverse Engineering

Increasingly, malware is object oriented and written in C++. Object-oriented malware presents considerable challenges to engineers because it rarely has source code available and must be reverse engineered.

Reverse engineering is challenging and time consuming, and traditionally requires skilled and experienced analysts. C++ data structures are especially difficult to reverse engineer because they maintain state across multiple functions, they include sophisticated mechanics, and they can be arranged in arbitrarily complex relationships.

The SEI's Pharos Binary Static Analysis Framework, built on the ROSE compiler infrastructure developed by Lawrence Livermore National Laboratory, includes tools that automate common reverse engineering tasks.

In 2017, the SEI released OOAnalyzer as part of the Pharos suite. OOAnalyzer automatically recognizes common patterns that indicate C++-style objects in assembly code. It exports these patterns as JSON, which, in turn, is read into IDA Pro by the OOAnalyzer.py plugin (part of the Pharos tool suite). The plugin helps malware analysts understand the program's design and functionality.

CallAnalyzer, another tool in the Pharos suite, statically reasons about the contents of memory at each function call. This reasoning provides reverse engineers with concrete information that identifies the program state and the values passed to each function.

ApiAnalyzer, a pattern-matching tool in the suite, allows analysts to find program behaviors based on program API usage. It enables reverse engineers and malware analysts to specify and then search for many potentially malicious API function call patterns.

To help malware analysts perform quick surface analyses, Fn2Hash and Fn2Yara generate function hashes and YARA signatures for each function in a binary.

The SEI continually updates the Pharos framework, adding new tools to the suite. These tools, combined in the Pharos Binary Static Analysis Framework, assist reverse engineers and malware analysts in gaining insight into software binaries and help them combat the intrusion of object-oriented malware.

To learn more about this work, visit the Pharos page in the SEI's digital library at insights.sei.cmu.edu/library/pharos/

2017

2017

Automating the Repair of Software Flaws

Codebases typically contain billions of lines of code that contain errors that can lead to costly security vulnerabilities. These errors are typically too numerous to vet manually, and finding and fixing coding errors manually is a time-consuming and error-prone process. This process is expensive; researchers from the SEI's CERT Division report that the average cost to manually fix one defect is $14,000.

Static code analysis tools can help find these errors, but these tools are typically used late in the development process and generate a huge number of error warnings. Even after excluding false positives, the volume of actual coding errors can overwhelm developers. Consequently, only a small percentage of the vulnerabilities identified are eliminated.

In 2016, CERT researchers developed tools to automatically detect and repair two common software-coding errors: integer overflows that lead to buffer overflow, and reads of stale and potentially sensitive memory. These CERT-developed tools infer the specification the developer intends—a strongly supported guess based on the pattern—and make a repair to satisfy the inferred specification.

The tool for integer overflows performs an additional check for error conditions where the overflow can lead to a memory violation. For software that is not safety critical, if the tool cannot fully repair an overflow, it simply inserts code that checks for an overflow and ends the process if it is detected.

The tool for invalid memory reads dynamically detects when the occurrence of a memory read falls outside the valid portion of a buffer. The tool addresses the problem of security vulnerabilities caused by such reads, which can leak sensitive information.

These tools help developers reduce the number of vulnerabilities in a codebase, freeing them to focus on fixing the remaining coding errors, developing secure code, and achieving their organization's software assurance goals.

Contributing to Developing and Implementing the DoD Vulnerability Discovery Program

The security research community regularly makes valuable contributions to the security of organizations and the broader Internet. Since maintaining the security of networks is a high priority at the U.S. Department of Defense (DoD), it recognizes that fostering a close relationship with the community helps improve its security. In 2016, the DoD identified a need for a transparent and modernized vulnerability disclosure program and asked the SEI's CERT Division to help develop and implement such a program.

Since 2002, the SEI has gathered, investigated, and published research about vulnerabilities, as well as curating the vulnerability notes database. This research provided the backdrop for its work with the Defense Cyber Crime Center (DC3) to develop the DoD Vulnerability Disclosure Program (VDP), based on the Hack the Pentagon and Hack the Army bug bounty pilots.

Photo: U.S. Army

During the first phase of the program, the SEI helped design processes and handle reports from researchers—validating vulnerabilities, passing them to the DC3 for mitigation, and validating the applied fixes. The SEI developed the CONOPS (Concept of Operations) and TTPs (tactics, techniques, and procedures) of the DoD VDP. The SEI also provided initial operating capability for DC3 until the planned hand-off in early 2018, after which, the SEI would provide only policy, process, and technical support.

The VDP is the DoD's legal avenue for researchers to find and disclose vulnerabilities in DoD public-facing systems. The program was the first of its kind for the DoD. Its clear guidance not only helps security researchers know how to test and disclose vulnerabilities in DoD websites, but it also commits the DoD to working transparently with the research community.

2016

Enhancing Computing Power at the Edge

As part of its mission to transition the technologies it develops into use, in 2015 the SEI made its implementation of tactical cloudlets, KD-Cloudlet, freely available in its open source code repository on GitHub.

To support their missions, military and emergency personnel operating in crisis and hostile environments increasingly use mobile applications. Most of these applications perform computation-intensive tasks, such as speech and image recognition, natural language processing, and situational awareness enhancement. These tasks take a heavy toll on a mobile device's battery power and computing resources. Unfortunately, battlefield and disaster environments are not only at the edge of the network infrastructure but are also resource constrained.

Cyber-foraging augments the capabilities of resource-limited mobile devices by leveraging computing resources in the surrounding environment. Cloudlet-based

2015

cyber-foraging relies on discoverable, generic, forward-deployed servers located in single-hop proximity of mobile devices.

Using KD-Cloudlet, developers can turn any system running Linux—from a laptop to a more powerful server—into a discoverable source that can be used by nearby mobile devices for computation offload and data staging.

The KD-Cloudlet tool's release springs from several years of SEI research into the use of cloud computing at the tactical edge. The research into the needs and constraints of tactical environments drove the development of the tactical cloudlets. SEI researchers collaborate in this ongoing research with the creator of the cyber-foraging and cloudlet concepts, Dr. Mahadev Satyanarayanan of CMU.

Photo: U.S. Army

Creating a New Language to Verify Complex Systems

Distributed, adaptive real-time (DART) systems (e.g., UAS) are key to DoD capability. These systems are safety critical, resource constrained, and sensor rich, and they adapt autonomously to their physical environments.

In general, formally verifying a DART system is intractable. Coordination, adaptation, and uncertainty pose key challenges to assuring their safety- and mission-critical behavior. The typical approach to verifying DART systems is to test rigorously and exhaustively. Testing, however, is usually performed later in development and cannot account for all reactions of an essentially autonomous system.

One innovative approach that SEI researchers are using involves creating a new programming language for DART systems called the DART Modeling and Programming Language (DMPL). DMPL is a C-like language that can express distributed, real-time systems. The semantics of this language are precise; it supports formal assertions usable for model checking and probabilistic model checking. In addition, in DMPL physical and logical concurrency can be expressed in sufficient detail to perform timing analysis.

The SEI's investigation into verifying DART systems will also produce other tools for mixed criticality scheduling and model checking. In addition, work to verify DART systems continues longer term SEI research into mixed-criticality and real-time scheduling, model checking, and high-confidence cyber-physical systems (HCCPS).

2015

Photo: U.S. Navy

2015

Integrating Early to Prevent Costly Problems

In a 2014–15 shadow exercise, the SEI rapidly detected potential integration issues early in Joint Multi-Role (JMR) development that traditional approaches missed, using its Architecture-Centric Virtual Integration Practice (ACVIP). The findings led to ACVIP adoption by JMR contractors and its inclusion in RFPs for new projects.

The roots of ACVIP are in SEI research into virtual integration that began in 1998. Unlike the traditional development approach of design–build components–integrate–test, the virtual integration approach employs architectural modeling to make sure the components work together before building components in conformance to the model.

DoD line funding enabled the SEI to lead the technical development of the SAE Architecture Analysis and Design Language standard (established in 2004) for the specification, analysis, automated integration, and code generation of real-time, performance-critical, distributed computer systems. Line funding, together with sponsorship by the Army and others, enabled the SEI to produce the Open Source AADL Tool Environment (OSATE) workbench for implementing virtual integration.

In 2008, the international Aerospace Vehicle Systems Institute (AVSI), whose membership includes defense industry organizations, chose AADL and OSATE for its System Architecture Virtual Integration (SAVI) initiative, based on evidence that the technologies offer a means to achieve an integrate-then-build approach to evolving complex systems.

Taming Uncertainty in Software Cost Estimation

In 2014, SEI researchers used their Quantifying Uncertainty in Early Lifecycle Cost Estimation (QUELCE) method in a workshop with a live major defense acquisition program (MDAP). This milestone along the way to transitioning innovation into the acquisition lifecycle is the result of focused research and development.

DoD acquisition regulations call for early (pre-Milestone A) estimates that stretch across the entire program lifecycle, including operations and support. These early cost estimates rely heavily on expert judgments about cost factors. In addition, the ways in which cost factors may change through the lifecycle receive little attention.

Based on research initiated in 2011, the QUELCE approach provides an explicit, quantified consideration of the uncertainty of change drivers. In doing so, QUELCE enables calculation (and recalculation) of the cost impacts caused by changes that may occur during the program lifecycle. The result is that this approach enhances decision making through transparency about the expert assumptions that underlie the cost estimate.


Risk management
Any Risk?
Yes
Probability
High
Impact
Low
Remedy action plan
Root cause analysis
Procedures
High
Risk avoidance
Deal with
Eliminate cause
Arrange risk assessment meeting

2014

Enabling a Stronger Cyber Workforce

For more than 15 years, the SEI has been investing in developing platforms and courseware for DoD and government cyber warrior readiness.

The SEI's CERT Division developed an initial Virtual Training Environment (VTE) platform using line funding in 2001. By 2005, the VTE was being used to address DoD training and capability-building challenges related to information security. In 2012, the VTE was redesigned to meet cyber workforce training requirements and transitioned as FedVTE to serve tens of thousands of government and military users. The SEI estimates that FedVTE has saved the government over $70 million dollars by providing the equivalent of 24,000 five-day training courses.

2014

The CERT Division followed VTE with a web-based system, the CERT Exercise Network (XNET). The USCYBERCOM Exercise Network is a customized instantiation of XNET. In 2012, the SEI introduced the Simulation, Training, and Exercise Platform (STEP), a flexible, multimedia, e-learning environment that students can access anywhere, anytime. STEP has formed the backbone infrastructure for USCYBERCOM's Cyber Flag and Cyber Guard joint exercises since their inception.

Most recently, in 2015, CERT researchers prototyped an Automated Cyber Readiness Evaluator platform to provide a scalable, objective assessment that validates the technical knowledge and skills of the government's cyber workforce.

Photo: Georgia Army National Guard

Attacking Software Vulnerabilities

In 2014, the SEI's CERT Division introduced the Tapioca tool to check Android apps for vulnerabilities. In the first year of use, Tapioca was used to check more than one million Android apps.

The release of the open source Tapioca tool, a network-layer man-in-the-middle proxy virtual machine, is one bit of evidence of the CERT Division's continuing commitment to proactive vulnerability discovery. The CERT Division vulnerability analysis team maintains over 1,400 vendor contacts, creating vulnerability reports that eventually appear as entries in the National Vulnerability Database.

The SEI also works directly with US-CERT to publish Vulnerability Notes directly to the US-CERT website, where they are considered the authoritative statement from the government regarding a given vulnerability. In addition, the SEI's CERT Division is the only organization that has proven to be able to, repeatedly and successfully, coordinate responses to a vulnerability across industry, the DoD, and the federal government.

2014

Illustration: Rob Bulmahn

Photo: U.S. Marine Corps

2014

Building Capability to Defend Against Malware

Malicious code, or malware, is a piece of software that runs without the user's explicit consent and maybe without the user's knowledge. Historically, malware has caused nuisance-type results, such as delivering unwanted content. In the last decade or so, more malware has focused on committing crime, such as stealing an identity or taking control of a computer.

For malware analysts, a significant challenge derives from the fact that malware rarely has source code available. Analysts must grapple with sophisticated data structures exclusively at the machine code level.

To help analyze malware, CERT researchers at the SEI are developing a suite of binary static program analysis tools based on a framework called Pharos. This framework is built on top of the Lawrence Livermore National Laboratory (LLNL) ROSE compiler infrastructure. The Pharos tool suite includes many extensions to the binary analysis features of ROSE that the SEI has jointly developed with LLNL. The Pharos tools use static analysis techniques, such as control flow analysis and dataflow analysis, to reason about the behavior of data structures in binary files.

In 2014, the SEI's CERT Division completed research to eliminate bottlenecks in the process of deriving actionable insights by automating tasks and providing more semantically rich abstractions used by a malware analyst.

Assessing Cyber Risk Readiness

One lesson of the past 20 years is that organizations cannot expect to prevent every cyber attack. Instead, they must be ready to continue operations and meet their missions when disruption occurs.

The SEI's CERT Division tools for cyber risk and resilience promote a structured approach to managing security risks, business continuity, and information technology operations in the context of business objectives.

Created by the CERT Division of the SEI for the U.S. Department of Homeland Security (DHS) in 2011, the Cyber Resilience Review (CRR) is a no-cost, voluntary, non-technical assessment to evaluate an organization's operational resilience and cybersecurity practices. The CRR assesses enterprise programs and practices across a range of 10 domains based on the CERT Resilience Management Model (CERT-RMM), including asset management, vulnerability management, incident management, risk management, and situational awareness. In 2014, DHS released a CRR self-assessment guide to allow organizations to conduct a CRR without outside facilitation. In 2015

alone, the CERT Division conducted 48 CRRs in 10 critical infrastructure sectors.

In 2012, the SEI's CERT Division developed the Risk and Vulnerability Assessment (RVA) to aggregate vulnerability data in support of informed decisions about the security and safety of information systems. An RVA combines national-level threat and vulnerability information with assessment data to provide specific risk analysis reporting and remediation steps. An RVA provides information on network mapping, penetration testing, wireless networks, databases, and other areas. During 2015, the CERT Division worked with DHS to conduct 46 RVAs.

In 2015, the SEI's CERT Division and DHS launched the External Dependencies Management (EDM) Assessment. This in-person, DHS-facilitated evaluation measures how well an organization can handle cyber disruptions in key services provided by third parties. Any external dependency presents a risk, from service agreements for cloud computing to business relationships that depend on a third party's computing infrastructure and security.

2011

2009

Certifying the Software Architect Role

In 2009, the U.S. Army mandated that all Project Executive Offices (PEOs) appoint a chief software architect (CSWA) to be responsible for oversight and management of software development within each PEO. The memo specified that the CSWA must earn a Software Architecture Professional Certificate from the SEI (or equivalent). The decision was based on an understanding of SEI work in software architecture and its impact, in particular an impact study of the use of SEI architecture evaluation techniques in the Army (see Army Requires PEOs to Appoint Chief Software Architect, **2009 Year in Review**).

The Army's mandate reflected appreciation for the value of more than 15 years of SEI innovation and leadership in software architecture definition, evaluation, analysis, and documentation. SEI work included the first software architecture book for practitioners, Software Architecture in Practice, winner of the prestigious JOLT award from Software Development magazine. Three other equally seminal books followed. All SEI software architecture books are cited often, have been updated in multiple editions, and have collectively sold more than 150,000 copies.

These books form the foundation of training courses and certificate programs, in which people from more than 900 organizations in industries such as defense, finance, health care, insurance, and energy have been trained by SEI experts. The SEI software architecture curriculum has been adopted by more than 80 colleges and universities around the world. The work also spawned the annual SEI Architecture Technology User Network Conference (SATURN).

Augmenting T&E with Assurance

SEI work on the use of assurance cases in the development of medical devices led directly to the FDA issuing draft guidance to manufacturers recommending the use of assurance cases and providing guidance for their use. As a result, infusion pump manufacturers are beginning to make use of assurance cases.

It is difficult to assure the safety, security, or reliability of net-centric systems of systems because of their size, complexity, and continuing evolution—and because they can exhibit undesired and unanticipated emergent behavior. (Emergent behavior is the actions of a system as a whole that are not simple combinations of the actions of the individual constituents of the system.)

Traditional software and systems engineering techniques, including conventional test and evaluation (T&E) approaches, cannot provide the justified confidence needed. The SEI is developing an assurance case methodology to augment testing and evaluation.

The assurance case provides a means to structure the reasoning that engineers use implicitly to gain confidence that systems will work as expected. It also becomes a key element in the documentation of the system and provides a map to more detailed information.

The concept of an assurance case was derived from the safety case, a construct that has been used successfully in Europe for over a decade to document safety for nuclear power plants, transportation systems, automotive systems, and avionics systems.

2009

Codifying Resilience Practice

In the aftermath of the 9/11 terror attacks, organizations began to seek answers to predictably and systematically control operational resilience through activities such as security and business continuity.

In October 2003, a group of 20 IT and security professionals from defense organizations, the financial services sector, IT, and security services met at the SEI to identify what could enable and accelerate IT operational and security process improvement. The bodies of knowledge identified included IT and information security governance, auditing, risk management, IT operations, security, project management, and process management.

Soon after, in March 2005, the SEI began work with the Business Continuity committee of the Financial Services

2009

Technology Consortium (FSTC), exploring the development of a reference model to help determine an organization's capability to manage operational resilience. Drawing on its experience with developing and evolving the widely used Capability Maturity Model Integration (CMMI) framework, the SEI developed the CERT Resilience Management Model (CERT-RMM), which has 26 process areas.

Since 2009, organizations in the DoD, the U.S. defense industrial base, U.S. federal civilian agencies, the financial services sector, and academia have been using the CERT-RMM to institutionalize improved processes for managing operational resilience and measure their benefit.

Strengthening Network Traffic Analysis

In 2007, the National Cyber Initiative made Einstein mandatory for all federal civilian agencies. The Department of Homeland Security (DHS) Einstein program helps protect federal computer networks and the delivery of essential government services.

First deployed in 2004, Einstein's capabilities for situational awareness are used throughout the federal government in part because of a casual conversation between SEI staff members and the DoD. That conversation led to the research and collaboration that produced a sophisticated suite of tools that can characterize network threats, assess the impact of security events, and identify vulnerable network infrastructure. Einstein integrates several distinct data collection/analysis systems and toolsets for network traffic analysis developed at the SEI CERT Division.

Initially, Einstein collected summary network traffic information at agency gateways and provided a high-level view of federal government network connections. The program has grown to provide an automated process for collecting, correlating, analyzing, and sharing computer security information across the federal government to improve our nation's situational awareness.

2007

Photo: U.S. Army

2004

Leading the Growth of an Architectural Modeling Standard

In 2004, the international industry standard SAE Architecture Analysis and Design Language (AADL) was published. Growing from DARPA-funded research into the MetaH and ACME architectural languages a decade or more before, the development of AADL was shepherded by the U.S. Army Aviation and Missile Research Division (AMRDEC) Software Engineering Directorate (SED) with technical leadership by the SEI.

Focusing its research for several years on architectural modeling and analysis for safety- and mission-critical systems, the SEI worked effectively across industry, government, and academic organizations to fashion the initial standard language and subsequent annexes. As technical lead for the standard, the SEI integrated several research technologies into the AADL standard, making it extensible, semantically well defined, and consistent.

Through its creation of the Open Source AADL Tool Environment (OSATE), the SEI has fostered pilot applications of AADL in a range of industrial pilot projects and the use of AADL and OSATE as a technology transition platform—as evidenced by their integration with formal analytical frameworks such as SysML and MARTE.

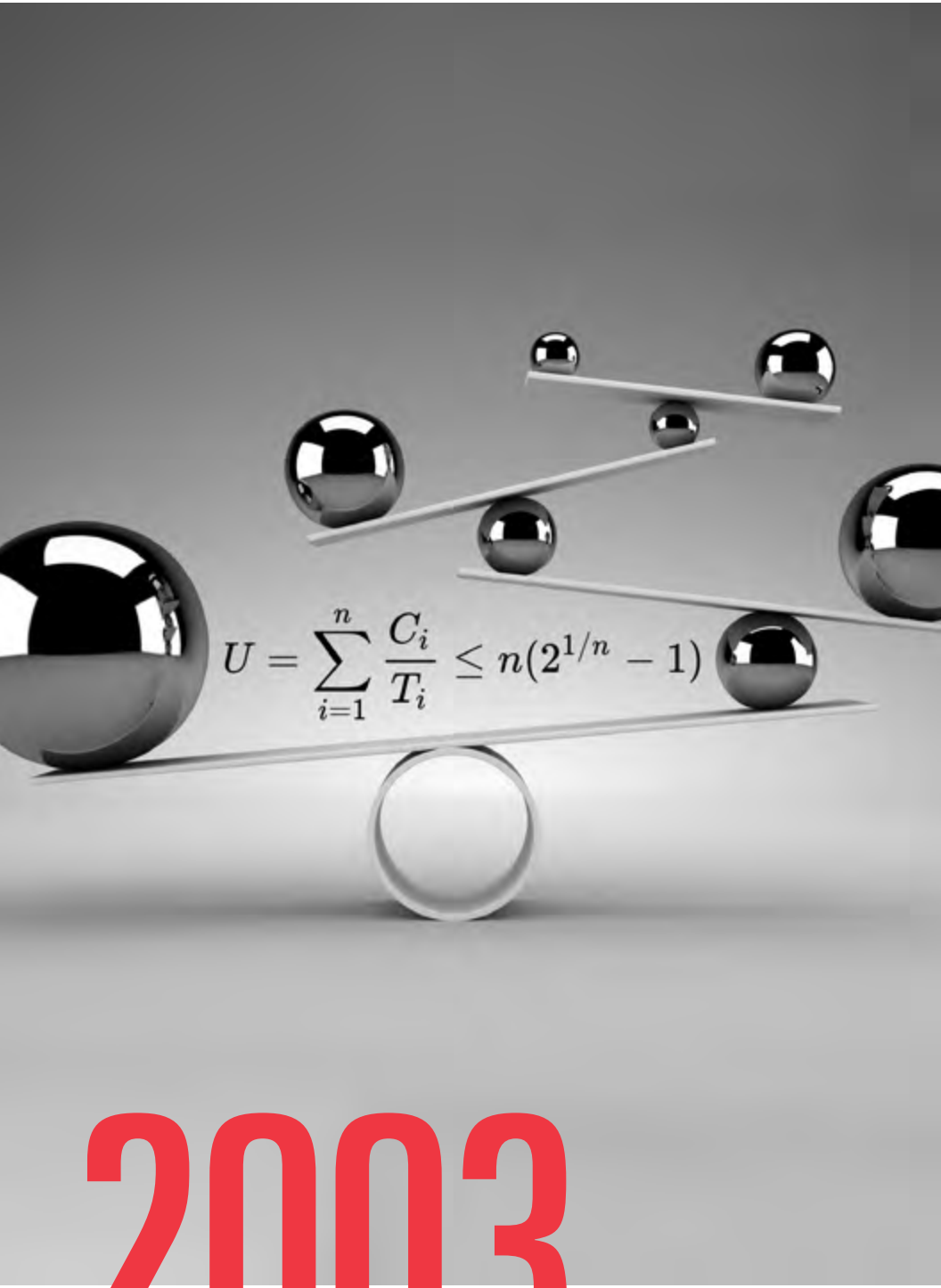
Defining Non-Functional System Qualities

Quality (or non-functional) attribute scenarios form a common language that users and software developers share, playing a significant role in requirements specification for an architecture and integration testing to see that requirements will be met. Questions of how secure, timely, reliable, and usable systems must be are now fundamental components of the processes used in all software development projects.

The idea that quality attributes influence the shape of the architecture and that the architecture is fundamental to the system emerged from SEI work in rate-monotonic analysis (RMA). From its work on RMA, the SEI gained the insight of considering system structure using an analytical framework. By analogy, SEI researchers realized that such a framework could be applied to quality attributes.

Developing systematic ways to relate the software quality attributes of a system to the system's architecture provides a sound basis for making objective decisions about design tradeoffs and enables engineers to make reasonably accurate predictions about a system's attributes that are free from bias and hidden assumptions.

SEI researchers tested and validated this insight into the primacy of quality attributes through conducting architecture evaluations. Whether they were evaluating a financial or an avionics system, conversant in the domains but not experts, they succeeded in finding risks by evaluating the systems from the point of view of different quality attributes. A lasting influence of the SEI work in the field of software architecture and software development can be seen in the pervasive attention paid to quality attributes and a general acknowledgment that requirements specifications need to include them.



2003

Standardizing More Secure Software

Software vulnerabilities expose the U.S. Department of Defense (DoD), other federal agencies, our nation's critical infrastructure, and businesses to attacks that could compromise their systems' integrity or expose or modify their critical information. Preventing the introduction of software vulnerabilities during software development is a proactive, efficient way to reduce risk before software is deployed.

Since forming its Secure Coding Initiative in 2003, the SEI's CERT Division has analyzed and cataloged thousands of software vulnerabilities and discovered that many share the same common errors. By engaging more than a thousand security researchers, language experts, and software developers, the CERT Division produced secure coding standards for common software development languages such as C and Java. These standards guide programmers to help them avoid coding errors that lead to vulnerabilities and provide them with example solutions.

2003

The U.S. military, other government agencies, and system developers from industry have adopted CERT Division secure coding standards, and Siemens and Computer Associates have licensed the SEI's training courses on secure coding in C and C++. Many others in military, government, and industry organizations have taken SEI courses, including the U.S. Navy, Cisco, Raytheon, Lockheed Martin, and Qualcomm.

In addition, courses based on the CERT Division standards for C and C++ are taught at major software engineering universities and colleges, such as Carnegie Mellon, Purdue, Stevens Institute, the University of Florida, and Santa Clara University.

Finally, through its security contributions to the ISO/IEC C-language specification, the SEI's CERT Division also influences developers of C language compilers, who conform their code to the ISO/IEC C-Standard and thus to countless software products written in the C language.

Learn more: insights.sei.cmu.edu/annual-reviews/2024-year-in-review/lasting-impact-the-cert-secure-coding-initiative/

Tailoring Risk Management Practice

In the 1990s, SEI risk research produced standards for software risk management, enabling program managers in all types of software-intensive programs to do a better job of identifying what could go wrong and mitigating the worst of those risks.

In 1996, the SEI published the *Continuous Risk Management Guidebook*, which brought together several concepts developed through its work with DoD agencies and Service branches in the preceding years. This approach had widespread influence. A Cutter Consortium's report a few years later, The State of Risk Management 2002, revealed that 21% of respondents to a survey about risk management techniques said that they used SEI standards for risk management. Only ISO ranked higher, with 36% of respondents.

In the decades since it was published the guidebook, the SEI has continued to conduct research and development in various aspects of risk management. In 1998, the SEI's CERT researchers began developing a new approach for managing cybersecurity risks within an organization, the Operationally Critical Threat, Asset, and Vulnerability Evaluation (OCTAVE). OCTAVE was transitioned and continues to be a widely used information security risk assessment method.

Other SEI-developed applications of risk management principles include the COTS Usage Risk Evaluation (CURE), the widely used Architecture Tradeoff Analysis Method (ATAM), and the Mission Risk Diagnostic (MRD), which assesses risk in interactively complex, socio-technical systems across the lifecycle and supply chain.

Much of the SEI's risk management work today is focused on software assurance. SEI researchers are developing the Security Engineering Risk Analysis (SERA) method, a systematic risk-based method for building security into software-reliant systems rather than deferring security to later lifecycle activities such as operations.

2002

Setting a Foundation for Software Architecture

Safety-critical components need to interact safely with less reliable and even unsafe components. For example, the flight control component in an autopilot is certified to DO-178B Level A (the highest level). However, it needs to accept guidance commands from a flight guidance system that is certified only to Level C. Nevertheless, avionics certification requires that Level A software must still function correctly in spite of the software failures in less-critical components.

The SEI developed an architecture template called the Simplex architecture, which supports overall safety when a system is composed of both reliable/safe components and less reliable/less safe components.

2001

In the Simplex architecture, a system is divided into two parts: a complex component that cannot be fully verified but is needed to provide an important service and a high-assurance control subsystem that is simple and fully verified. The Simplex architecture also ensures predictable and guaranteed timing behaviors in spite of failures of complex components and allows restarting or replacing complex components during operation. Notable applications of Simplex architecture principles include the F-22 and F-35 aircraft.

Photo: U.S. Army

Changing Software Contractor Selection Criteria

At the beginning of the 21st century, the Capability Maturity Model Integration (CMMI) framework team published CMMI appraisal requirements, ushering in a new era for appraisals. In partnership with government and industry, the SEI published the Standard CMMI Appraisal Method for Process Improvement (SCAMPI), along with the specification for two other appraisal classes.

Later, the SEI developed SCAMPI B and C as a 100% community-funded project. Factors that might influence an organization's choice of a SCAMPI (A, B, or C) include cost, schedule, accuracy, efficiency, and desired results. SCAMPI continues to have a wide range of uses, including internal process improvement and external capability determinations.

The SEI's contribution also includes creating the SCAMPI Lead Appraiser role through the certification of lead appraisers (500 as of 2013), based on the SCAMPI Lead Appraiser Body of Knowledge (SLA BOK).

SEI work on assessing/evaluating contractors led the DoD and other government acquisition organizations to change their criteria for selecting contractors. In awarding contracts, they consider how well the contractors' software development processes follow CMMI goals and practices.

2001

2000

Bringing Science to Insider Threat Mitigation

For nearly two decades, the SEI's CERT Division has focused on gathering and analyzing data about actual malicious insider acts—including espionage, IT sabotage, fraud, and theft of confidential information—and potential threats to U.S. critical infrastructures.

In 2001, the DoD Personnel Security Research Center (PERSEC) sponsored the first CERT Division research into the malicious actions of insiders. A few years later, the Department of Homeland Security (DHS) added its sponsorship to build a database of information on more than 150 actual insider threat cases. The database now contains more than 1,000 cases, which CERT researchers analyze from technical and behavioral perspectives.

Carnegie Mellon University's CyLab published the first edition of the Common Sense Guide for mitigating insider threats in 2005, based on CERT Division research. CyLab establishes public/private partnerships to develop new technologies for measurable, secure, available, trustworthy, and sustainable computing and communications systems. Subsequent editions of the Common Sense Guide to Mitigating Insider Threats were released in 2006, 2009, 2012, and 2016.

Applying analytical methods to its insider threat cases, the CERT Division produced additional guidance and tools for government programs to detect, mitigate, and prevent insider threats that include

- interactive training simulation and workshops—*beginning in 2007*
- the Insider Threat Vulnerability Assessment—*beginning in 2009*
- The CERT Guide to Insider Threats (Theft, Sabotage, and Fraud) —*first published in 2012*
- transition of linguistic analysis tools to DoD/Intelligence Community (IC) customers—*2015*
- certificate programs to build skills in preventing and handling insider threats—*2015*

Enabling Large-Scale Network Flow Analysis

Today, network analysts in the DoD and federal agencies use the SEI CERT Division's network situational awareness technologies to characterize network threats, assess the impact of security events, and identify vulnerable network infrastructures.

In the early 1990s, the CERT Division developed Argus, one of the first software-based network flow analysis tools, to support incident response activity. In 2000, the Automated Incident Reporting to CERT (AirCERT) initiative released data conversion, sharing, and analysis tools (Analysis Console for Incident Data—ACID) and supported the development of Internet Engineering Task Force (IETF) standards to establish a data format for exchanging information on computer security incidents among response teams around the world.

The Einstein program, mandatory for all federal civilian agencies, integrates several distinct data collection and analysis systems and uses tool sets for network traffic analysis developed by the CERT Division. Through the years, the SEI's CERT Division has developed and released open source tools such as

- the System for Internet-Level Knowledge (SiLK) tool suite, which enables the DoD to conduct security analysis not driven by known-bad signatures
- Yet Another Flowmeter (YAF), which leverages additional data sources, including Domain Name System, Secure Socket Layer certificates, and application banners stored in the IP Flow Information eXport standard format

YAF, SiLK, and associated tools have been widely adopted. Telecommunication providers, government defense contractors, and many other high-tech companies use this technology to help protect their own networks and the networks of their clients.

2000

Evaluating System Architecture

One recurring theme in defense challenge problems is the need to predict problems before a system has been built. Maintenance and improvement costs represent more than half the total cost of a system, a percentage that has grown steadily since 1960. A problem for the DoD is to predict problems with modifiability before the system is constructed and before these problems occur.

The SEI pioneered the use of scenario-based methods in the evaluation of software architectures for modifiability and other qualities. The first SEI-developed architecture analysis method, the Software Architecture Analysis Method (SAAM), introduced the

1994

concept of a quality attribute scenario, giving specific modifications against which the system is to be tested. The SAAM led directly to the Architectural Tradeoff Analysis Method (ATAM), which evaluates a system for a collection of quality attributes.

Major defense contractors, such as Boeing and Raytheon, have architecture evaluation teams and architecture evaluation as a portion of their architect certification process. U.S. Army staff have reported that using scenario-based architecture evaluation methods reduced risk in schedule and cost, improved documentation, and resulted in a higher quality product.

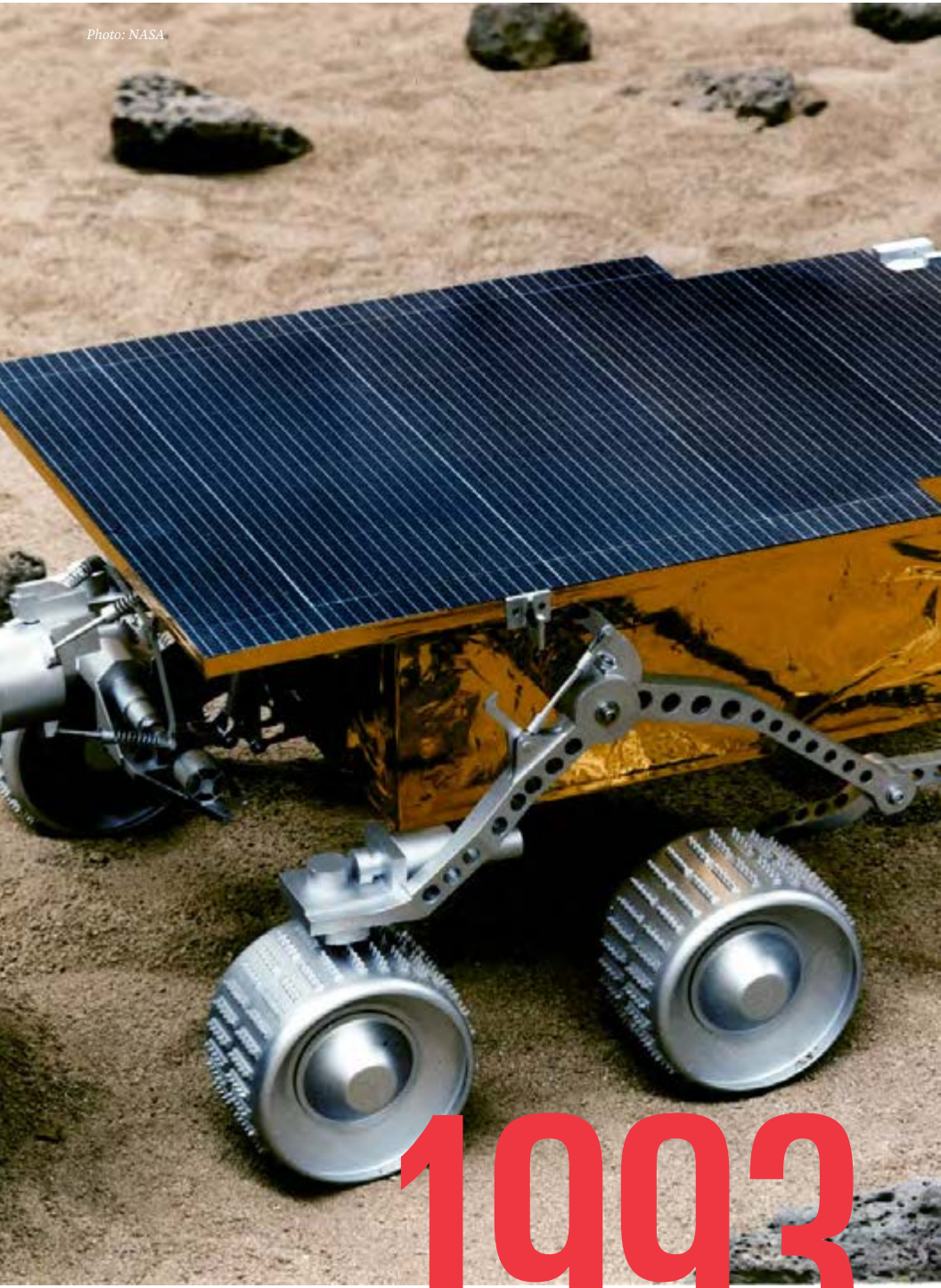


Photo: NASA

1993

Meeting Real-Time Scheduling Needs

Today, rate-monotonic analysis (RMA) is part of real-time computing textbooks and the only real-time scheduling technology approved by the FAA for Level A avionics software in networked control applications with distributed computers, sensors, and actuators.

The importance of RMA emerged when a software bug caused the computer on the Mars Pathfinder to reset and jeopardized the 1997 mission. Computer scientists patched the software to fix the bug using the rate-monotonic scheduling algorithm. Years before, the SEI was instrumental in the development of the rate-monotonic scheduling paradigm, and its technical staff played a crucial role in the development of the theory.

In 1993, the SEI published A Practitioner's Handbook for Real-Time Analysis: Guide to Rate Monotonic Analysis for Real-Time Systems, which contains quantitative methods that enable real-time system developers to understand, analyze, and predict the timing behavior of many real-time systems. In addition, the SEI created training workshop and consultation services for RMA early adopters.

The SEI (and others') work in RMA transformed real-time engineering practice. Also emerging from the SEI's work in RMA in the following years were two other ideas that underpin contemporary software system development practice: (1) quality attributes influence the shape of the architecture, and (2) the right architecture is fundamental to system success.

Transforming Software Quality Assessment

The SEI's publication of the Capability Maturity Model for Software (Software CMM) in 1991 changed the view in government and industry about software quality. The model consisted of best practices in key process areas, giving organizations an objective standard for software development.

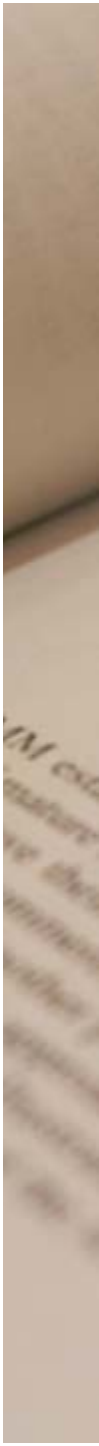
By 1986, the DoD and defense contractors recognized that some software engineering practices produced working software with greater consistency. Unfortunately, those practices were not documented or widely recognized.

Asked to conduct a study of "best practices," the SEI met with leading software professionals in the DoD, defense industry, commercial industry, and academia to develop a consensus on the practices that consistently lead to improved software development. To help organizations determine how well their work stacked up against these practices, the SEI produced a Maturity Questionnaire. Response to this questionnaire was overwhelmingly positive, from both the DoD and the defense industry.

After assisting several organizations with their assessments and subsequent improvement efforts, the SEI produced a guide for how organizations might manage that process. As the community began to adopt these ideas, some expressed a need for a more precise definition of the practices and the underlying model. As a result, the SEI published the Software CMM.

Many people contributed to the ideas in the Software CMM, and more than a few of those ideas preceded the SEI effort. It was the SEI's leadership that brought software community experts and practitioners together and its role as assimilator that filtered the ideas into a consistent framework and documents that became a worldwide de facto standard for software process improvement. The new structure for improvement, the capability maturity model, became a seminal information architecture that has been mimicked and adapted over time.

Eventually, the Capability Maturity Model Integration (CMMI) framework, managed with software community guidance by the SEI for more than a decade, evolved from the Software CMM.

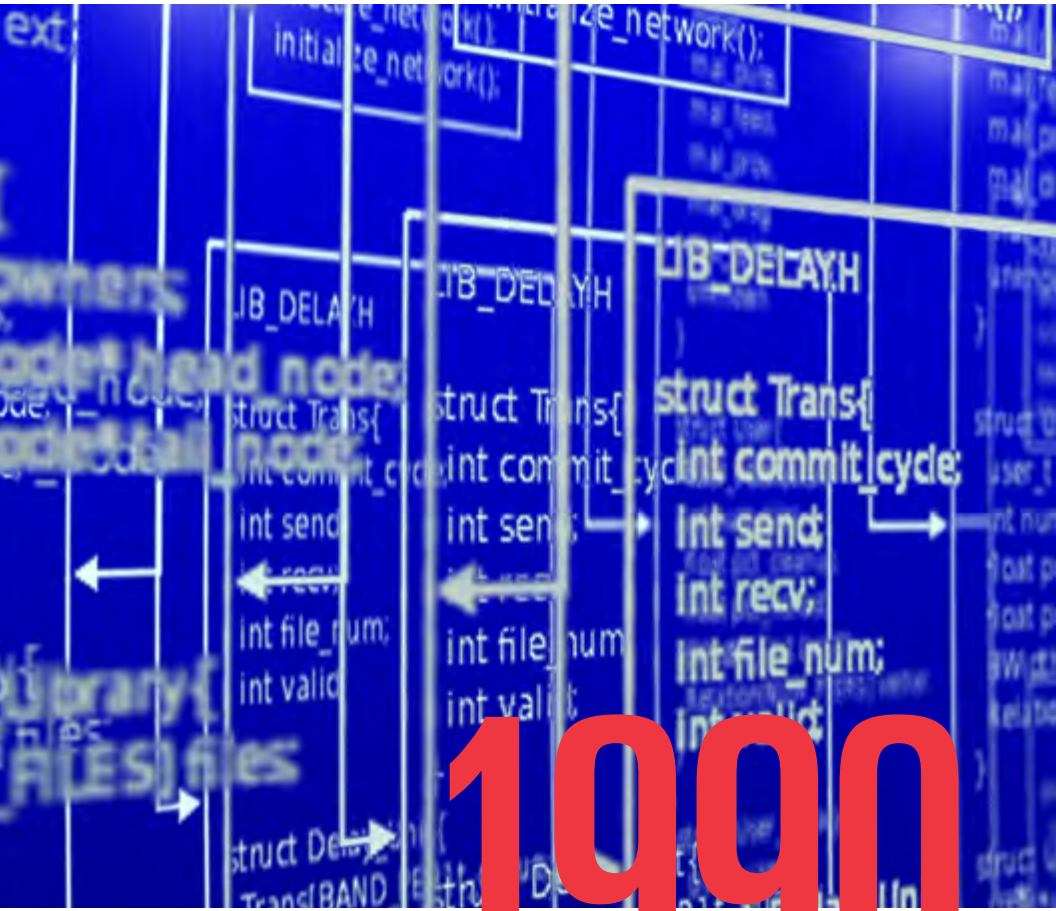


1991

Establishing a Basis for Software Reuse

Systematic software reuse is a strategy that can bring products to market or field more quickly, improve quality, and lower costs. Recently, this strategy has become more popular in the increasingly competitive development environment brought about by budgetary restrictions. For example, the DoD Systems Engineering FY 2014 Annual Report (issued in March 2015) notes that the CH-53K Heavy Lift Replacement Helicopter includes 7 million software lines of code, with 64 percent reuse.

Underlying today's efforts to reuse software is a 1990s technology called feature-oriented domain analysis (FODA). Developed by the SEI,



1990

FODA analyzes a problem domain across multiple similar systems to identify common and variable features. In developing FODA, the SEI demonstrated that managing variation was essential to systematic software reuse and that simply identifying common elements and features is insufficient.

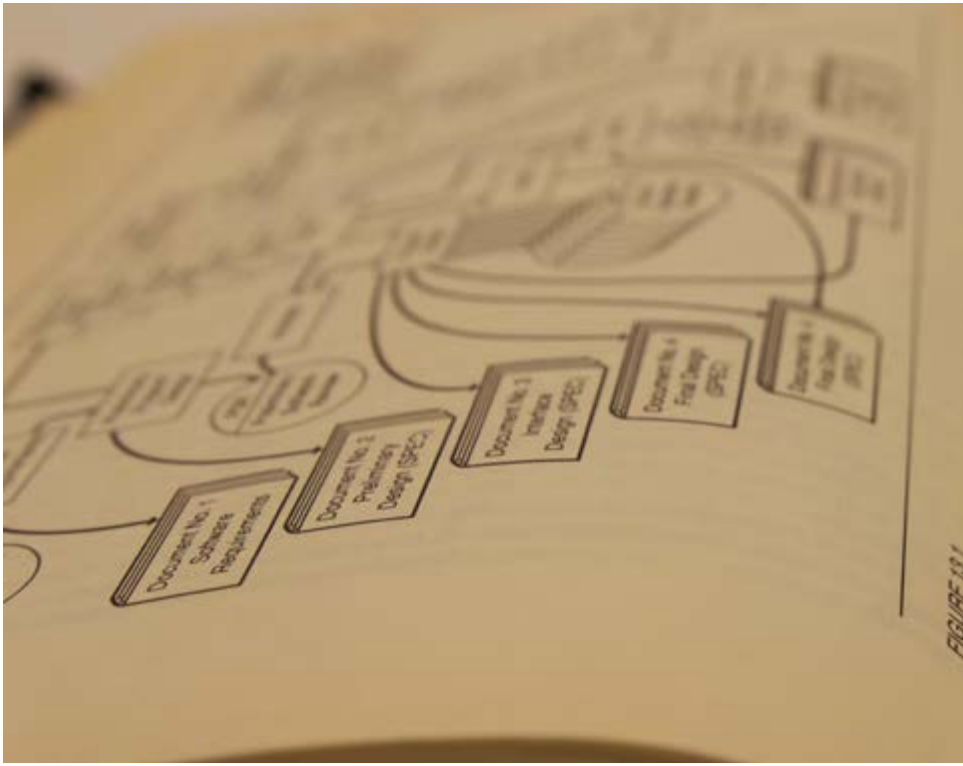
At the SEI, FODA later evolved into product line analysis, which extended the analysis of commonality and variability beyond features to quality attributes.

Building the Master of Software Engineering Curriculum

Today, there are more than 100 accredited software engineering schools in the United States, and about 1.5 million people work in fields related to software development. Nearly all university software engineering-related curricula trace their lineage to SEI-led efforts.

The SEI education effort provided needed leadership during the early years of curriculum development in software engineering education. In shaping a software engineering curriculum, the SEI engaged the academic community in creating the materials and amplified technology transition with government and industry by making materials available to allow other organizations to teach material it had developed.

In the winter of 1988, the SEI held a workshop of leading software engineering educators to design a recommended curriculum for a Master of Software Engineering degree. The SEI curriculum recommendations that grew from that



workshop were published at the annual Conference on Software Engineering Education and Training (CSEE&T), a series started by the SEI that continues today with its own independent steering committee and sponsorship.

The number of software engineering programs nearly doubled in the first three years after the publication of the guidelines. Most of those programs followed the recommended guidelines. Another outgrowth of the curriculum project was the development of materials called curriculum modules and educational materials, which helped to transition the MSE curriculum and support faculty members who wished to offer software engineering courses.

In subsequent years, the SEI worked with the Association of Computing Machinery (ACM), the Institute of Electrical and Electronics Engineers (IEEE), and others to influence the quantity and quality of undergraduate software education.

1989

1988

Pointing the Way Toward a Software Architecture Discipline

In studies dating back to 1978, data showed that the cost of development and modification of the user interface contributed over 50 percent of the total cost of software ownership. Attempts to reduce the cost of developing defense systems clearly had to include reduction in the cost of developing and maintaining the user interface.

The high cost of developing and modifying the user interface led to user interface management systems (UIMSs), a class of system intended to reduce this cost. Serpent was a UIMS that approached the problem of reducing the total ownership cost of the user interface by separating the user interface and functional portions of a system, allowing for modifications to the user interface with minimal impact on the remainder of the system.

Through its work on Serpent, the SEI contributed to a greater understanding by a generation of user interface researchers about the impact of software engineering architectural decisions on the ease of modifying the user interface. This work introduced an important concept to the discipline of software architecture that emerged in the 1990s.

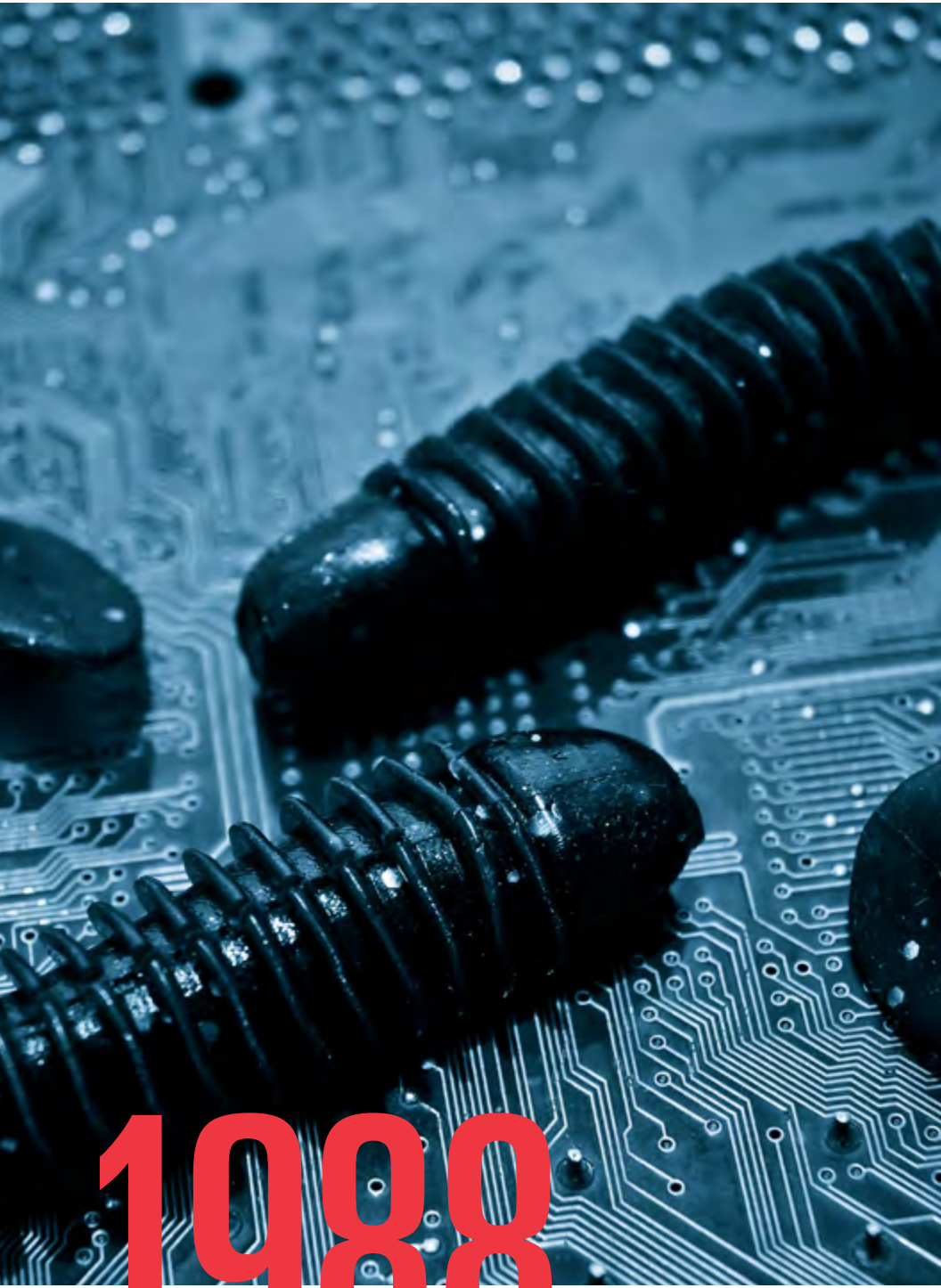
Fostering Growth in Professional Cyber Incident Management

The SEI's CERT Coordination Center (CERT/CC) was born from a newfound national concern about malicious attacks on communications networks. Graduate student Robert Morris jarred the network-connected world from ambivalence regarding cybersecurity on November 2, 1988, by releasing a worm that brought the nascent Internet to its knees.

In the aftermath of the Morris Worm attack, DARPA asked the SEI to establish a computer emergency response team, which has come to be known as the CERT/CC. As a neutral third party, the CERT/CC reports vulnerabilities to vendors without revealing the identity of the reporter. This role allows the CERT/CC to work with competing vendors whose products contain the same vulnerability, free of conflicts of interest.

Since its formation, the CERT/CC has facilitated the mitigation of vulnerabilities and disseminated information through the publication of Vulnerability Notes, which include summaries, technical details, remediation information, and lists of affected vendors. The CERT/CC maintains a knowledgebase that includes a publicly available Vulnerability Notes Database.

In addition, the CERT/CC has been instrumental in building a network of more than 50 national computer security incident response teams (CSIRTs) using tools and training that help managers, project leaders, CSIRT staff, and computer forensic professionals.



1988

References

[Alberts 2003] Alberts, Christopher & Dorofee, Audrey. *Managing Information Security Risks: The OCTAVE Approach*. Addison-Wesley Professional. 2003. ISBN 03211188630.

[AMIT 2001] Members of the Assessment Method Integrated Team. *Standard CMMI Appraisal Method for Process Improvement (SCAMPI), Version 1.1: Method Definition Document*. CMU/SEI-2001-HB-001. Software Engineering Institute, Carnegie Mellon University. 2001. sei.cmu.edu/library/scampi-v11-use-in-supplier-selection-and-contract-process-monitoring

[Ardis 1989] Ardis, M. & Ford, G. SEI Report on Graduate Software Engineering Education. *Proceedings of the Third SEI Conference on Software Engineering Education*. CSEE, Pittsburgh, PA. July 18-21, 1989. Published as *Springer Lecture Notes in Computer Science* 376, 1989.

[Chick 2022] Chick, Timothy et al. *DevSecOps Platform Independent Model (PIM)*. Software Engineering Institute, Carnegie Mellon University. 2022. cmu-sei.github.io/DevSecOps-Model

[CMMI 2001] CMMI Product Team. *Appraisal Requirements for CMMI, Version 1.1 (ARC, V1.1)*. CMU/SEI-2001-TR-034. Software Engineering Institute, Carnegie Mellon University. 2001. sei.cmu.edu/library/appraisal-requirements-for-cmmi-version-11-arc-v11

[DoD 2023] Department of Defense. *DoD Directive 3000.09: Autonomy in Weapon Systems*. Department of Defense. 2023. defense.gov/News/Releases/Release/Article/3278076/dod-announces-update-to-dod-directive-300009-autonomy-in-weapon-systems

[Dorofee 1996] Dorofee, Audrey et al. *Continuous Risk Management Guidebook*. Software Engineering Institute, Carnegie Mellon University. 1996. sei.cmu.edu/library/continuous-risk-management-guidebook

[Fowler 1990] Fowler, Priscilla & Rifkin, Stanley. *Software Engineering Process Group Guide*. CMU/SEI-90-TR-024. Software Engineering Institute, Carnegie Mellon University. 1990. sei.cmu.edu/library/software-engineering-process-group-guide

[Hayes 2002] Hayes, William et al. *Handbook for Conducting Standard CMMI Appraisal Method for Process Improvement (SCAMPI) B and C Appraisals, Version 1.1*. CMU/SEI-2005-HB-005. Software Engineering Institute, Carnegie Mellon University. 2002. kilthub.cmu.edu/articles/report/Handbook_for_Conducting_Standard_CMMI_Appraisal_Method_for_Process_Improvement_SCAMPI_B_and_C_Appraisals_Version_1_1/6574073

[Humphrey 1988] Humphrey, Watts et al. *A Method for Assessing the Software Engineering Capability of Contractors*. CMU/SEI-87-TR-023. Software Engineering Institute, Carnegie Mellon University. 1988. sei.cmu.edu/library/a-method-for-assessing-the-software-engineering-capability-of-contractors

[Inacio 2010] Inacio, Chris & Trammell, Brian. *YAF: Yet Another Flowmeter*. Software Engineering Institute, Carnegie Mellon University. 2010. sei.cmu.edu/library/yaf-yet-another-flowmeter

[Jones 2006] Jones, C. The Economics of Software Maintenance in the Twenty First Century. 2006. citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=b596667d929137deb6c24083b249c31f09773346

[Kang 1990] Kang, Kyo et al. *Feature-Oriented Domain Analysis (FODA) Feasibility Study*. CMU/SEI-90-TR-021. Software Engineering Institute, Carnegie Mellon University. 1990. sei.cmu.edu/library/feature-oriented-domain-analysis-foda-feasibility-study

[Marsan 2008] Marsan, Carolyn Duffy. Morris Worm Turns 20: Look What It's Done. *Network World*. October 30, 2008. networkworld.com/article/792726/morris-worm-turns-20-look-what-it-s-done.html

[Masters 2007] Masters, Steve et al. *SCAMPI Lead Appraiser Body of Knowledge (SLA BOK)*. CMU/SEI-2007-TR-019. Software Engineering Institute, Carnegie Mellon University. 2007. sei.cmu.edu/library/scampi-lead-appraiser-body-of-knowledge-sla-bok

[Nord 2009] Nord, Robert et al. *Impact of Army Architecture Evaluations*. CMU/SEI-2009-SR-007. Software Engineering Institute, Carnegie Mellon University. 2009. sei.cmu.edu/library/impact-of-army-architecture-evaluations

[Pharos 2017] Pharos Product Team. *Pharos*. Software Engineering Institute, Carnegie Mellon University. 2017. sei.cmu.edu/library/pharos

[RCTA 1992] RCTA, Inc. *Software Considerations in Airborne Systems and Equipment Certification*. DO-178B. December 1, 1992.

[SEI 2009] Software Engineering Institute. Army Requires PEOs to Appoint Chief Software Architect. *2009 Year in Review*. Software Engineering Institute, Carnegie Mellon University. 2010.

[SEI 2023] Software Engineering Institute. *AISIRT Advances National Security with Secure AI*. Software Engineering Institute, Carnegie Mellon University. 2023. sei.cmu.edu/projects/aisirt-ensures-the-safety-of-ai-systems/

[SEI 2025] Software Engineering Institute. Lasting Impact: The CERT Secure Coding Initiative. *2024 Year in Review*. Software Engineering Institute, Carnegie Mellon University. 2025. insights.sei.cmu.edu/annual-reviews/2024-year-in-review/lasting-impact-the-cert-secure-coding-initiative

[SCAIFE 2019] SCAIFE Product Team. *SCAIFE-API*. Software Engineering Institute, Carnegie Mellon University. 2019. github.com/cmu-sei/SCAIFE-API

[Sha 2001] Sha, L. Using Simplicity to Control Complexity. *IEEE Software*. Volume 18. Issue 4. July/August 2001. Pages 20–28.

[Sutton 1978] Sutton, Jimmy A. & Sprague, Ralph H., Jr. *A Study of Display Generation and Management in Interactive Business Applications*. Technical Report RJ2392. IBM Research. November 1978.

[Weinstock 2009] Weinstock, Charles & Goodenough, John. *Towards an Assurance Case Practice for Medical Devices*. CMU/SEI-2009-TN-018. Software Engineering Institute, Carnegie Mellon University. 2009. sei.cmu.edu/library/towards-an-assurance-case-practice-for-medical-devices

Copyright

INNOVATIONS

Contact Us

Carnegie Mellon University
Software Engineering Institute

412.268.5800

sei.cmu.edu

Locations

SEI Pittsburgh, PA
4500 Fifth Avenue Pittsburgh,
PA 15213

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