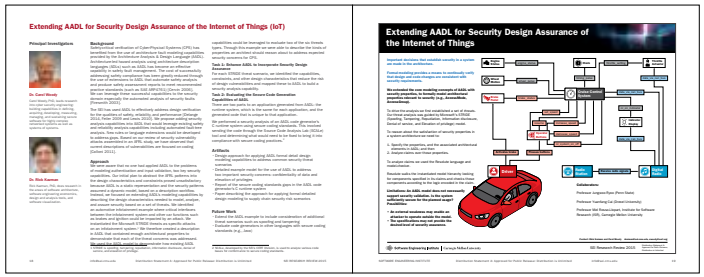
SEI Research Review 2015

Project Summaries and Posters

October 7–9, 2015

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.



This booklet contains descriptions of SEI research projects and images of posters related to the research. In each of the sections, you will find a project description on a left-hand (even-numbered) page and a poster image facing it on a (odd-numbered) right-hand page.

The Annual SEI Software and Cybersecurity Research Review

Dr. Kevin Fall

Welcome to Carnegie Mellon University (CMU) and the Software Engineering Institute (SEI). We are excited to provide you with this booklet describing our recent line-funded research into software and cyber challenges confronting people engaged in the acquisition, design, development, operation, and sustainment of systems that are increasingly reliant on software.

The SEI is a federally funded research and development center (FFRDC) sponsored by the U.S. Department of Defense (DoD). The SEI invests its federally appropriated line funding in research and development (R&D) activities that produce artifacts useful in addressing challenges in DoD and other government priority areas. Artifacts include prototypes, algorithms, standards, models, automated techniques, and software tools.

The SEI operates in the role of a value-added broker of R&D for DoD and other government users. The SEI adds value by managing its R&D portfolio and working with members of the software ecosystem in government, academia, and industry to customize, develop, and adapt software and cybersecurity technologies and related methods for the measurable benefit of the U.S. government. To act effectively in its role, the SEI maintains capabilities in several key areas: the government lifecycle for software systems, software technologies, and cybersecurity tools and methods.

With the access afforded by its DoD affiliation and conflict-free status as an FFRDC, SEI has a nearly unique ability to undertake technical work ranging from fundamental research targeting widespread publication to support of sensitive government programs. SEI technical work—line-funded research and sponsored engagements—also helps to accomplish strategic goals such as building organic software and cyber capabilities in the FFRDC's government users, improving our own overall capabilities, and achieving synergy with other parts of CMU.

Dr. Kevin Fall
Deputy Director, Research, and CTO
Carnegie Mellon University Software Engineering Institute

Contents

Acquisition and Management

Agile in Government: Validating Success Enablers and Inhibitors

Principal Investigators

Mary Ann Lapham

Mary Ann Lapham works to improve the acquisition of software-reliant systems through research and applying technologies. For DoD programs such as 3DELRR, AEHF, CCS-C, GBS, TSAT, and others, this means working with the Program Office to assist and advise on software issues at the system and/or segment level.

SuZ Miller

SuZ Miller performs research and builds work products related to Agile Adoption in Regulated Settings (such as those in the U.S. Government). Miller also works with enterprise-level customers that are contemplating or that have embarked on the adoption of Agile or Lean methods in regulated settings.

Background

The Office of the Secretary of Defense and other organizations have released in the public domain several documents indicating that Agile[1] is not a passing fad in DoD; rather Agile is here to stay [Bellomo 2012]. Even with multiple government documents urging the use of Agile-like approaches, there is a lack of community guidance for acquisition organizations on how to employ Agile or oversee contractors that are using Agile.

Since 2009, we have collected anecdotal and structured qualitative data on issues, barriers, enablers, success stories, and failure modes associated with the adoption of development approaches that use Agile and Lean principles in government settings, particularly the DoD. Overall, three major results have ensued:

- adaptation of a general adoption risks model for technologies, including practice-based technologies, to address the emerging Agile barriers/enablers that appear to be unique to regulated settings
- 12 guideline documents for those who are either mandated to use Agile methods or desire to use them within DoD, addressing "deep-dive" topics that seemed to have caused practitioners particular problems
- convening and growth of an SEI Agile Collaboration Group (ACG) with over 170 individuals representing all military services, several prominent federal agencies, defense contractors, and vendors

Approach

Our current research question: "What are the key barriers and enablers to (1) successfully adopting Agile principles and practices and (2) applying them to achieve positive project results, all within the constraints of DoD/federal acquisition guidance?" These questions move us toward transition of knowledge about productive implementation strategies and acquisition workforce development, through three task areas:

1. **Agile Project Success Indicators:** We are adapting Kaplan's well-known balanced scorecard to guide acquisition programs intending to use or using Agile-principles-based approaches to set appropriate success measures and monitor against those measures throughout the lifecycle. Our prototype will address the question "How do I know if my Agile program is successful?" from the viewpoint of multiple stakeholders and will provide questions that need to be addressed.
2. **Agile Readiness & Fit Analysis (RFA):** We refined the RFA model based on the execution of the research plan begun in FY2014, to give users confidence concerning Agile adoption risks and mitigations—for the acquirer or contractor.
3. **Guideline documents:** We are creating two guideline documents with "deep dives" in areas that our ACG has requested: Technical Reviews and Agile (updated) and Application of Commercial Scaling Frameworks to Agile Settings in Government. Commercial scaling frameworks will include, at minimum, Disciplined Agile Delivery, Dynamic Systems Development Method, and Scaled Agile Framework. The degree of discourse on applicability of these frameworks to government acquisition settings is highly variable.

Artifacts

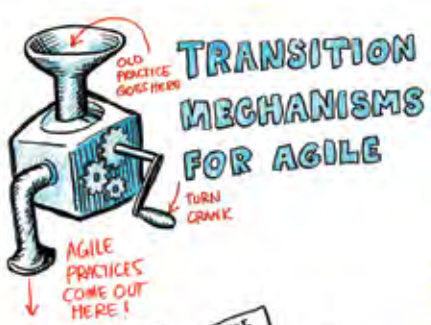
- RFA: refined model
- Agile Project Success technical note
- Technical Reviews and Agile technical note updated
- Agile Scaling Frameworks and DoD technical note

Future Work

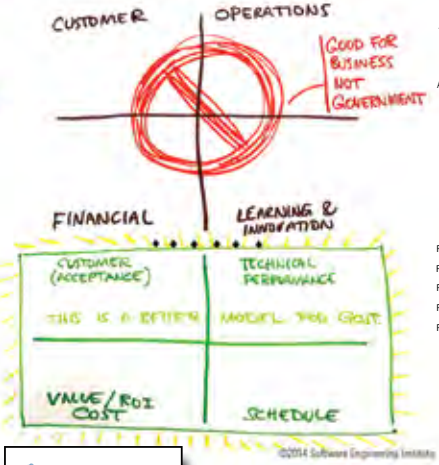
We propose a one-year follow-on to our current research to ensure the totality of our contribution is sound, robust, and has immediate impact.

1 By Agile, we mean a highly collaborative approach with just enough ceremony to produce high quality software within software-reliant system contexts.

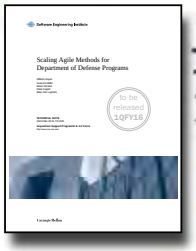
Agile in Government
The SEI Journey



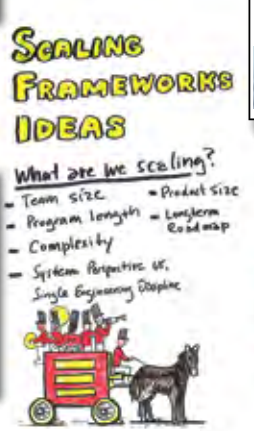
Requirements



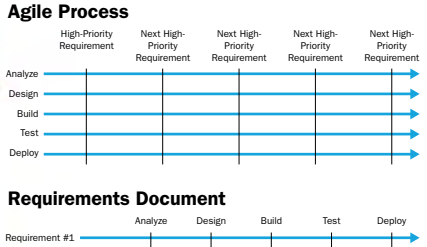
Measures of Success



Scale

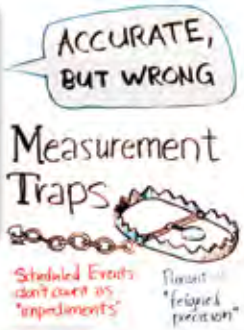


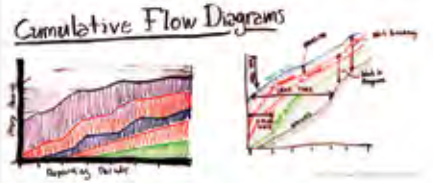
Contracts

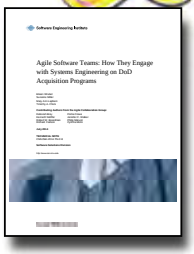


Parallel Worlds

Metrics







Systems Engineering

Contact: Mary Ann Lapham mlapham@sei.cmu.edu

Machine Learning to Support Big Data System Acquisition

Principal Investigator

John Klein

John Klein consults with commercial and government organizations to develop and evolve architectures that satisfy business and mission goals. Through these consulting engagements, John identifies common challenges and then conducts research to develop practical and repeatable solutions across the entire architecture lifecycle.

Background

The Defense Science Board report on the acquisition of information technology noted, "The subject matter competencies required for successful enterprise IT system acquisition are too often missing in government" [DSB 2009]. Without powerful decision support technologies for acquisition lifecycle practices, failures like the recent Integrated Electronic Health Record system and the 2000% cost increases in Defense Intelligence Agency systems noted in a 2014 GAO report will become common [GAO 2014].

Our FY2014 research resulted in QuABaseBD, a tool to improve competency in big data systems for acquisition and development practitioners. QuABaseBD represents a significant step toward improved acquisition of Big Data systems.[1] However, the current manual approach to populating QuABaseBD is not scalable, given the rapid evolution of technology. This project therefore overcomes this limitation by addressing the research question: Can machine-learning methods automate the population of a knowledge base to enhance the acquisition of Big Data IT systems?

Approach

To answer the research question, we propose to use an advanced machine learning method based on established theories [Romero 2010] to update QuABaseBD content automatically, in order to create a dynamic, up-to-date decision support knowledge base for IT systems acquisition. The project will build on QuABaseBD and on the Concept Graph Learning (CGL) machine-learning method[2] developed at the Carnegie Mellon University Language Technology Institute. The existing QuABaseBD semantic knowledge model provides the necessary linked information for CGL to learn a directed universal concept graph that represents the major concepts in Big Data systems.

CGL will use the learned graph to predict unobserved relations from new data—in our case the documentation pages for specific Big Data technologies that we wish to include or update in QuABaseBD. This will enable QuABaseBD to be updated rapidly to reflect the characteristics of new and evolving implementation technologies and will create a unique decision support capability for DoD acquisition of the next generation of scalable Big Data systems.

Our first step is to construct a canonical concept graph that represents the major concepts in Big Data architectures, using the CGL algorithm. The input to the CGL algorithm will be QuABaseBD knowledge base content containing semantic relations (links) between those elements of the knowledge base. QuABaseBD is built upon a formal semantic model. This semantic model makes it possible to query the knowledge base and generate suitable input for the CGL algorithm.

The CGL output will be a directed graph whose nodes are universal canonical concepts and whose links are relations among the concepts. The concepts nodes will include quality attributes, design tactics, tradeoffs, and supporting technology features; the links represent how these concepts are logically related.

Based on that canonical concept graph, CGL will input the documentation for at least three Big Data technologies that are not represented in the QuABaseBD knowledge base and will construct concept graphs for each. These technology-specific graphs will have mappings from their nodes to nodes in the canonical concept graph. We will post-process the CGL output to update QuABaseBD with the newly learned information. (The CGL method provides mapping from concept-level links to specific QuABaseBD content.)

Artifacts

- Automatically extensible knowledge base for acquiring and designing Big Data systems
- Refined CGL algorithm for building acquisition decision support knowledge bases

Future Work

- Trials with DoD programs acquiring Big Data technology
- Investigation of coupling QuABaseBD with a reasoning engine to provide semi-automated acquisition decision support

1 Learn more about QuABaseBD in the SEI Webinar "Software Architecture for Big Data Systems" that is available at http://www.sei.cmu.edu/webinars/view_webinar.cfm?webinarid=298346&gaWebinar=SoftwareArchitectureforBigDataSystems.

2 Yang, Y., Concept Graph Learning from Educational Data, under review

Machine Learning to Support Big Data System Acquisition

Problem:

In rapidly evolving technology domains, there is no efficient way to create decision support tools with up-to-date, comprehensive product information for comparison, evaluation, and selection.

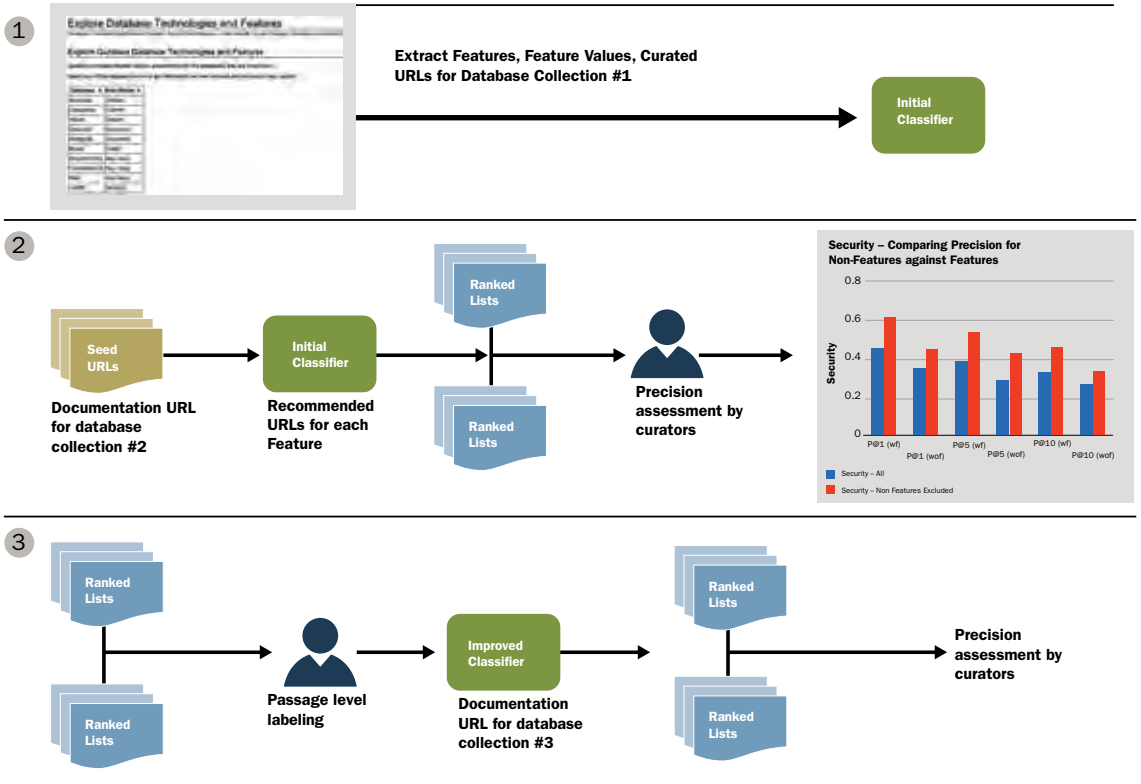
- Choose a modern technology stack (playbook.cio.gov)
- The subject matter competencies for successful enterprise IT system acquisition are often missing in government. (GAO)

Solution Approach

- Use machine learning to populate knowledge bases for decision support with latest information about rapidly evolving technology domains.
- Validate approach with big data technologies to show feasibility for broader application (e.g. SOA, analytics) in acquisition and IT system modernization

Results Towards Automating Curation

- Demonstrated ability to classify documents according to knowledge base feature taxonomy for positive feature values
- Classifier precision improved as training set was augmented



Accuracy of Curating Consistency and Security for 4 Different Distributed Databases.

Product Name	Consistency Feature Model	Security	Overall
Oracle	90.10	70.80	**80.40**
Berkeley	81.80	83.30	**82.50**
Couchbase	66.70	79.20	**71.90**
Redis	84.80	91.70	**87.70**
Mean	*80.16*	*81.25*	***80.65***

Research Team Leads
Ian Gorton, PhD, Prof. Yiming Yang, CMU LTI

Contact: John Klein jklein@sei.cmu.edu

Quantifying Uncertainty for Early Lifecycle Cost Estimation (QUELCE)

Principal Investigator

Robert Stoddard

Robert Stoddard promotes the use of advanced statistical, probabilistic, and simulation techniques for both software-intensive systems research and client support. His research interests span cybersecurity, secure coding, insider threat, engineering management, software reliability, software quality, software process improvement, and cost estimation.

Background

We propose to provide a sustainable and "living" approach to more capable DoD cost estimation by maturing and performing final validation on the QUELCE method and automating it through integration with a machine-learning mechanism. Our objective is to realize the full potential of QUELCE and render it transferable to the DoD in a way that is organic with the DoD acquisition lifecycle.

QUELCE has evolved into a 5-step method:

1. Anticipate applicable program execution change drivers leveraging a domain reference repository of historical DoD program experiences
2. Assess strength of cascading dependencies among change drivers
3. Represent cascading relationships in a Bayesian Belief Network (BBN) probabilistic model
4. Mathematically connect the BBN outputs with existing cost estimating relationships
5. Derive cost distributions using Monte Carlo simulation

Acknowledging the shortcomings of expert judgment is vital to the first two steps, we developed training to improve and baseline the effectiveness of deciding the probabilities. We developed a domain reference repository proof-of-concept to house structured data of change history (1,500+ highlighted change instances) from a set of over 250 DoD program artifacts. The repository helps the experts to anchor their judgment, thereby mitigating two challenges in eliciting expert judgment—namely over-optimism and over-confidence. The remaining expert judgment error is then modeled within the BBN.

Stakeholder feedback from the DoD cost-estimating community has been invaluable.[1] An early 2013 technical challenge workshop brought us additional DoD and Government Accounting Office (GAO) feedback followed by comments from the Defense Acquisition University (DAU). In 2014, we completed a validation QUELCE workshop for an active DoD program in addition to two separate commercial validation uses of QUELCE.

Approach

Our FY2015 research question was, "Can machine learning be successfully trained from a corpus of DoD program artifacts (e.g., pdf and Word documents of technical and programmatic nature, PowerPoint Gate Reviews, and Excel spreadsheets of program cost and schedule information), which have been manually coded by domain experts using a repeatable taxonomy of program execution change drivers?" We intend to motivate Service and DoD-level cost groups to use machine learning as a means to automatically update the domain reference repository with current change driver probabilities of occurrence.

Artifacts

- Adoption-ready program execution change driver taxonomy
- More representative and complete domain reference repository
- Machine learning automation to process future DoD program artifacts
- Templates for connecting the QUELCE BBN to common CERs
- Transition artifacts including tools, job aids, and training
- Several SEI-external scholarly publications

Future Work

- Classroom/eLearning materials surrounding QUELCE, including achievement of format requirements for use by DAU in their classroom/CLM curriculum
- 1–2 day workshop on expert judgment training and calibration testing using DoD MDAP and ACAT reference data points
- Fuller automation of QUELCE method
- Natural language processing and machine learning component of a QUELCE repository
- Integration of the QUELCE method with the Security Engineering Risk Analysis (SERA) method[2]

1 We benefited from feedback provided by members of the Cost Assessment and Program Evaluation (CAPE), the Air Force Cost Analysis Agency (AFCAA), the Naval Center for Cost Analysis, and the Office of the Deputy Assistant Secretary of the Army Cost Estimation (ODASA-CE).

2 For more information, please visit http://www.cert.org/cybersecurity-engineering/research/security-engineering-risk-analysis.cfm.

Quantifying Uncertainty in Early Lifecycle Cost Estimation (QUELCE)

QUELCE Workshop

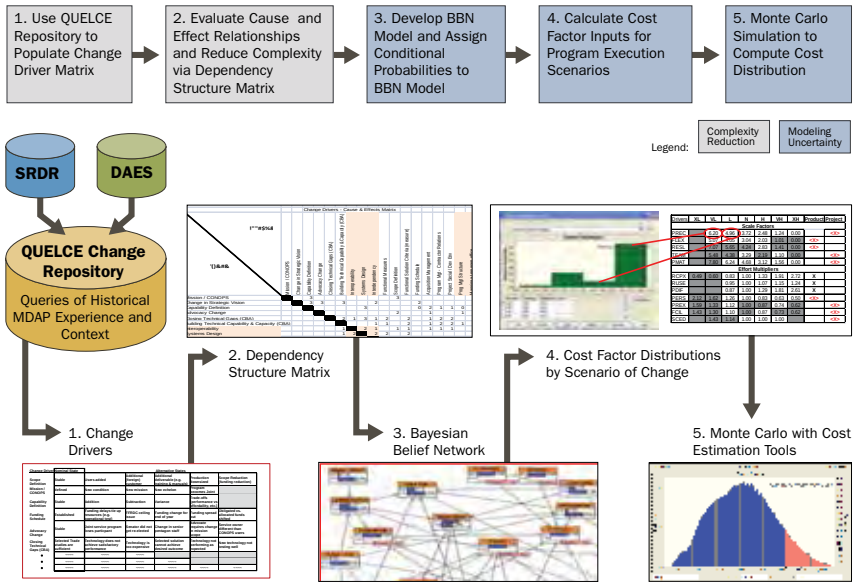
The Quantifying Uncertainty in Early Lifecycle Cost Estimation (QUELCE) workshop enables a client to convene a set of domain experts to formulate early lifecycle cost estimates expressed as cost distributions rather than single points. The QUELCE method involves a five-step process that begins with identifying potential future changes to nominal program execution that will influence program cost. This is followed by probabilistic modeling of the interrelationships of the program change drivers and Monte Carlo simulation of cost model inputs to create program cost estimate distributions. Because many of the inputs are based on subject-matter expert judgment, this workshop also involves a novel approach to calibrating expert judgment through a series of training exercises.

Data Requirements

- Pre-workshop access to existing planning artifacts, such as AoA and ICD/CDD
- Access to domain experts who can anticipate different reasons for cost changes during program execution

Time Frame

- SEI preparation of 1–2 weeks to review available documentation with two SEI staff members
- Two SEI staff members on site for 5–7 days to facilitate five 3-hour workshops with both technical and financial program office staff
- 5–7 days to prepare baseline estimate and suggested scenario-based estimates
- Typically, 3–5 days to assist program office staff with explaining estimates as needed



Expected Results

QUELCE produces a cost estimate that is represented as a distribution from which a decision maker can understand the level of risk associated with a particular cost value. It also produces an executable model that can be used to run alternative scenarios and that can be updated in the future for reestimation purposes. The model and information developed also provide good documentation of the basis of the estimate.

Publications

Quantifying Uncertainty in Early Lifecycle Cost Estimation (QUELCE)
www.sei.cmu.edu/library/abstracts/reports/11tr026.cfm

Quantifying Uncertainty in Expert Judgment: Initial Results
www.sei.cmu.edu/library/abstracts/reports/13tr001.cfm

Improving the Reliability of Expert Opinion within Early Lifecycle Cost Estimation
blog.sei.cmu.edu/post.cfm/improving-the-reliability-of-expert-opinion-within-early-lifecycle-cost-estimation

QUELCE Research

Objective

Quantify expert judgment of anticipated program execution uncertainties and enable more accurate inputs to existing cost models.

Description

Continuing research into the QUELCE method includes

1. Calibrating group judgments of the probabilities of change driver occurrence and co-occurrence
2. Expanding the QUELCE change-driver taxonomy to include detailed sustainment change drivers
3. Prototyping of supervised machine learning to enable the automatic processing of a future stream of DoD program artifacts. This will help create a "living" domain reference point repository benefiting ongoing DoD cost estimation.

Collaboration Opportunities

- Calibrating expert judgment in a group setting
  - Hubbard-style calibration to create more stability in elicited parameters
- Designing and mapping QUELCE BBN output nodes to cost model inputs
- Defining and classifying program change drivers
- Expanding the use of QUELCE in MDAP and PMO risk management programs to enhance the identification of future risks

Contact: Robert Stoddard, rws@sei.cmu.edu

Improving Software Sustainability through Data-driven Technical Debt Management

Principal Investigators

Dr. Ipek Ozkaya

Ipek Ozkaya, PhD, works to develop, apply, and communicate effective methods for software architecture and agile and iterative development to improve software development efficiency. At the SEI, she is the deputy lead for the Architecture Practices (AP) initiative. She also serves as the chair of the advisory board of the *IEEE Software* magazine and as an adjunct faculty member for the Master of Software Engineering Program at Carnegie Mellon University.

Dr. Robert Nord

Robert Nord, PhD, develops and communicates effective methods and practices for software architecture. His research interests include scaling Agile development by incorporating architecture practices; architectural technical debt; software architecture design, description, and evaluation; and architecture-based development and evolution.

Background

Budget constraints and the need to accelerate capability delivery have resulted in the DoD's adoption of incremental approaches and a shift from new system acquisition to more cost effective system evolution and sustainment [McLendon 2014]. Accumulated design and implementation decisions made for expediency, in the absence of structural quality requirements or without due consideration for sustainment and evolution, often result in systems that become prohibitively expensive to maintain or extend [Bellomo 2013]. Such issues contribute to technical debt, a metaphor that resonates with many and is rapidly gaining traction in research and industry [Ernst 2015].

Our project is developing an integrated, automated suite of tools and techniques for detecting and visualizing technical debt to provide a comprehensive view of the technical debt that projects need to manage. We know that technical debt can arise from design or implementation decisions, but current tools focus primarily on the latter (e.g., complexity, cyclicity metrics) for quantifying debt without any underlying scientific basis or validation. Our prior research illustrated deficiencies in detecting significant types of technical debt when architectural abstractions are not considered [Nord 2012].

Moreover, the DoD and industry have started to invest heavily in model-driven and compositional software development approaches that existing technical debt identification techniques do not adequately address.

Our tooling approach will combine existing code-oriented analysis tools with techniques that uncover and analyze architectural abstractions to reveal structural causes of technical debt. This will provide a more accurate scope for technical debt that exists in the system and assist both ongoing development and difficult sustainment decisions such as how to balance system improvements, early delivery, and upgrades.

Approach

1. Codify known architectural sources of technical debt that are not addressed adequately by today's code-oriented tools (e.g., safety-critical testing partitioning, unbalanced modules, and dependency violations) [Nord 2014, Zazworka 2011, and Xiao 2014]. This will draw from our experience comparing tool results and architecture evaluation results for systems (e.g., government health-IT exchange system and collaborator examples), literature review, surveys, interviews, and SEI's data on architectural risks and quality attribute scenarios.
2. Identify architecture indicators through abstractions (e.g., interfaces, restrict compositional dependencies) and anti-patterns that are correlated with technical debt and that can be automatically identified by analyzing source code and other project artifacts. Implement validation techniques to identify violations [Nord 2013].
3. Integrate these architectural indicators with code indicators in an experimental workbench.
4. Conduct empirical studies over multiple releases of at least two systems to correlate the identified indicators with observable project measures such as cost to fix, cost to implement new features, and defects. Such statistical correlations have been found, but only a very few technical debt indicators have been tested [Nord 2014]. More analyses are needed to indicate, within a given program, whether such instances remain worthwhile investments or need to be resolved. In building statistical correlations, we will use techniques that allow for testing qualitative causal assumptions via the data that we collect (e.g., structural equation modeling).

Artifacts

At the end of the two years, this project envisions to deliver:

- Architectural technical debt management workbench
- An underlying extensible measurement model of architectural abstractions related to integration and composition (e.g., decomposition, compositionality, and partitioning) that other researchers can build on
- Demonstration on a real system, documented in a case study
- Provisional rules for detecting likely sources of technical debt, along with correlations to cost to fix, cost to implement a new feature, and defects
- Two peer-reviewed publications in leading conferences
- Organizational survey results of leading and lagging indicators of technical debt for community dissemination

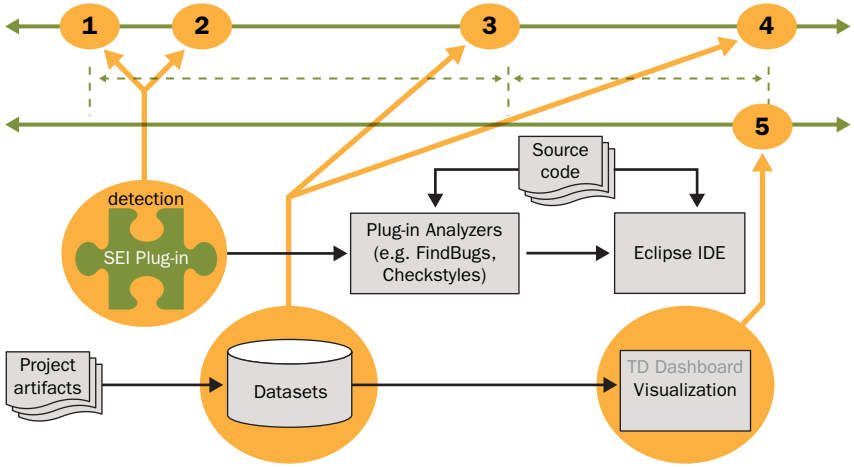
Future Work

- Extensions to the open source technical debt model and tooling to include other key quality attributes concerns (e.g., security and architectural technical debt management tooling)
- Relationship of technical debt management and testing
- Extensions to the data sets of rules for detecting likely sources of technical debt, along with correlations to cost to fix, cost to implement a new feature, and defects with other case studies

Improving Software Sustainability through Data-driven Technical Debt Management

Technical debt conceptualizes the tradeoff between the short-term benefits of rapid delivery and long-term value. In an effort to manage budget constraints, the DoD is increasingly searching for tool-supported approaches to manage technical debt. The goal of this project is to develop suite of tools and techniques for detecting and visualizing technical debt and provide exemplar data sets.

The technical debt metaphor is widely used to encapsulate numerous software quality problems. The metaphor is attractive to practitioners as it communicates to both technical and non-technical audiences that if quality problems are not addressed, things may get worse. However, it is unclear whether there are practices that move this metaphor beyond a mere communication mechanism. Existing studies of technical debt have largely focused on code metrics and small surveys of developers. Here, we report on our survey of 1,831 participants, primarily software engineers and architects working in long-lived, software-intensive projects from three large organizations, and follow-up interviews of seven software engineers. We analyzed our data using both non-parametric statistics and qualitative text analysis. We found that architectural decisions are the most important source of technical debt. Furthermore, while respondents believe the metaphor is itself important for communication, existing tools are not helpful in managing the details. We use our results to motivate a technical debt timeline to focus management and tooling approaches.

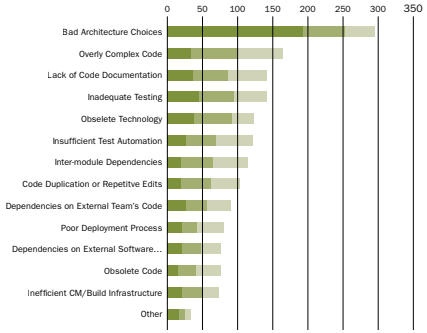


Technical debt timeline:
1: technical debt is incurred; 2: technical debt is recognized; 3: plan and re-architect; 4: technical debt is paid-off; 5: continuous monitoring

Our approach includes

1. Codify known architectural sources of technical debt that are not addressed adequately by today's code-oriented tools (e.g., safety-critical testing partitioning, unbalanced modules, dependency violations)
2. Identify architecture indicators through abstractions (e.g., interfaces, restrict compositional dependencies) and anti-patterns that are correlated with technical debt and can be automatically identified by analyzing source code and other project artifacts.
3. IIntegrate these architectural indicators with code indicators in an experimental prototype.
4 Conduct empirical studies over multiple releases of at least two systems to correlate the identified indicators with observable project measures such as cost to fix, cost to implement new features, and defects.

Most significant technical debt is architectural according to developers.



Neil A. Ernst, Stephany Bellomo, Ipek Ozkaya, Robert L. Nord, Ian Gorton: Measure it? Manage it? Ignore it? software practitioners and technical debt. SEC/SIGSOFT FSE 2015: 50-60 ACM SIGSOFT Distinguished Paper Award

Our findings are consistent with our approach. Architecture choices are key sources of technical debt. Architectural issues are difficult to deal with, since they were often caused many years previously. Monitoring and tracking drift from original design and rationale is vital, but tools do not capture the key areas of accumulating problems in technical debt. Technical debt is most useful when discussed in the context of executable system artifacts (such as code, automated test suites, build scripts). We identify areas of code that have multiple maintainability issues correlated with increased number of defects and changes, and we zoom in on those that have more accumulation compared to others.

- 75% of the respondents said that dealing with the consequences of technical debt has consumed a painful chunk of project resources.
- Tooling is a necessary component of any technical debt management strategy. But developers mostly rely on only issue trackers.

The SEI Architecture Practices team has been a pioneer in advancing the research agenda regarding technical debt. You can collaborate with us by

- **Contributing your technical debt examples**
- **Sharing observed gaps in tools to manage technical debt**
- **Collaborating on in-depth analysis of your project and sharing your data**

Contact: Ipek Ozkaya ozkaya@sei.cmu.edu

Acquisition and Management References

[Bellomo, 2012]
Bellomo, Stephany & Woody, Carol. *DoD Information Assurance and Agile: Challenges and Recommendations Gathered Through Interviews with Agile Program Managers and DoD Accreditation Reviewers.* CMU/SEI-2012-TN-024. Software Engineering Institute, Carnegie Mellon University, 2012. http://resources.sei.cmu.edu/library/asset-view.cfm?AssetID=34083

[Bellomo 2013]
Bellomo, S. et al. A study of enabling factors for rapid fielding: combined practices to balance speed and stability. Pages 982-991. In *Proceedings of the 25th International Conference on Software Engineering (ICSE 2013)*. San Francisco, CA (USA). May 2013. http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=6596173

[DSB 2009]
Defense Science Board (DSB). *Report of the Defense Science Board Task Force on Department of Defense Policies and Procedures for the Acquisition of Information Technology*. Office of the Secretary of Defense, 2009. http://www.acq.osd.mil/dsb/reports/ADA498375.pdf

[Ernst 2015]
Ernst, Neil A. et al. Measure It, Manage It, Ignore It: Software Practitioners and Technical Debt. Presentation at the *International Conference on Foundations of Software Engineering/European Software Engineering Conference (ESEC/FSE)*. Bergamo, Italy, 2015. http://www.slideshare.net/NeilErnst/measure-it-manage-it-ignore-it-software-practitioners-and-technical-debt

[GAO 2014]
U.S. Government Accountability Office (GAO). *Major Automated Information Systems: Selected Defense Programs Need to Implement Key Acquisition Practices*. GAO, 2014. http://www.gao.gov/assets/670/662045.pdf

[McLendon 2014]
Mclendon, M. et al. Addressing Software Sustainment Challenges for the DoD. *Crosstalk*. Volume 27. Number 1. Jan/Feb 2014. Pages 27-32. http://www.crosstalkonline.org/storage/issue-archives/2014/201401/201401-McLendon.pdf

[Nord 2012]
Nord, R. et al. In Search of a Metric for Managing Architectural Technical Debt. Pages 91-100. In *Proceedings of the 2012 Joint Working Conference on Software Architecture & 6th European Conference on Software Architecture* (*WICSA/ECSA 2012).* Helsinki, Finland. August 2012. http://dx.doi.org/10.1109/WICSA-ECSA.212.17

[Nord 2013]
Nord, R. et al. Variations on Using Propagation Cost to Measure Architecture Modifiability Properties. Pages 400-403. In *Proceedings of the 29th IEEE International Conference on Software Maintenance (ICSM 2013)*. September 2013. Eindhoven, The Netherlands. http://resources.sei.cmu.edu/library/asset-view.cfm?assetID=88709

[Nord 2014]
Nord, R. et al. Architectural Dependency Analysis to Understand Rework Costs for Safety-Critical Systems. Pages 185-194. In *Proceedings of the 36th International Conference on Software Engineering (ICSE 2014)*. Hyderabad, India. May-June 2014.

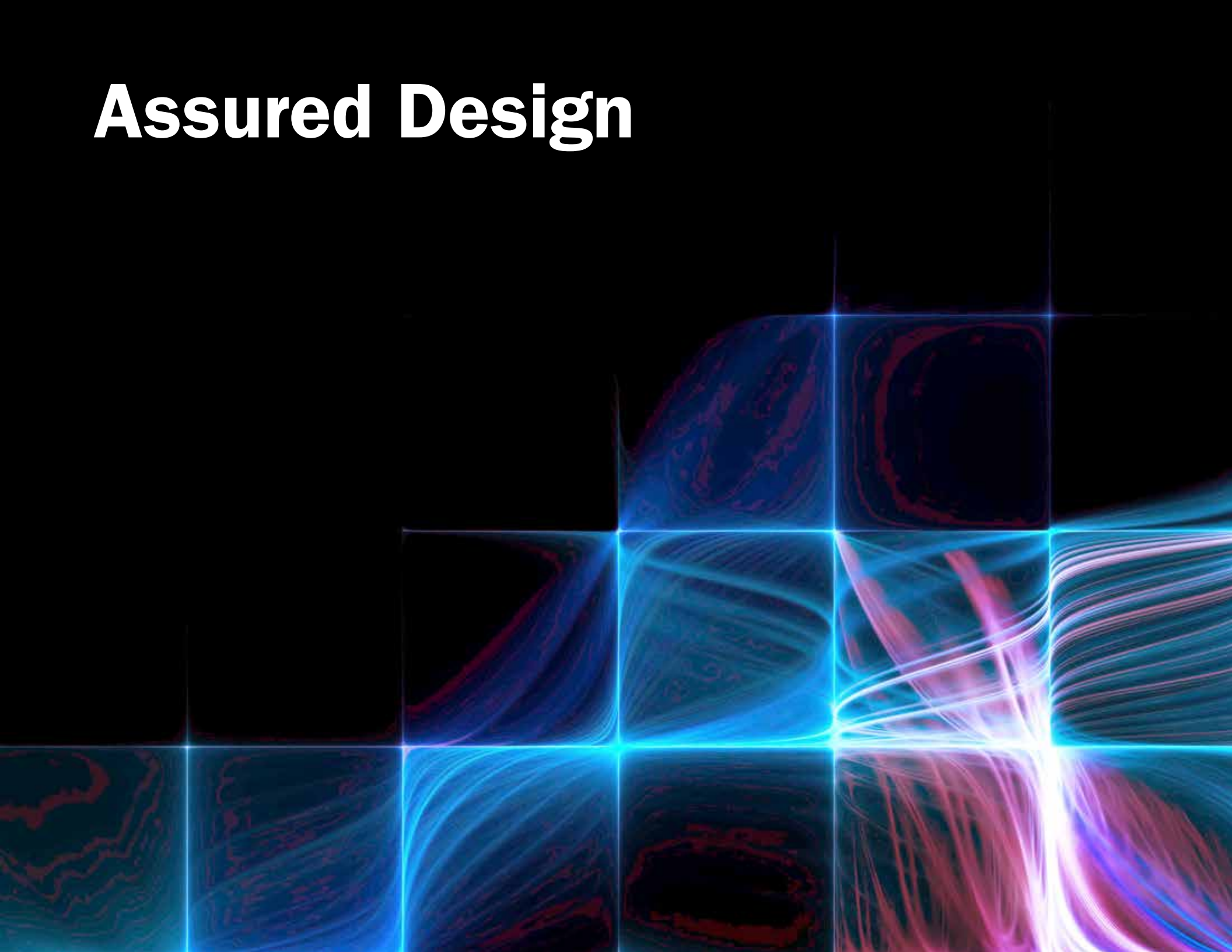
[Romero 2010]
Romero, C. & Ventura, S. Educational Data Mining: A Review of the State of the Art. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews.* Volume 40. Number 6. November 2010. Pages 601, 618.

[Xiao 2014]
Xiao, L. et al. Design Rule Spaces: A New Form of Architecture Insight. Pages 967-977. In *Proceedings of the 36th International Conference on Software Engineering (ICSE 2014)*. Hyderabad, India. May-June 2014. http://dl.acm.org/citation.cfm?id=2568241&CFID=544065544&CFTOKEN=61670460

[Zazworka 2011]
Zazworka, N. et al. Investigating the impact of design debt on software quality. Pages 17-23. In *Proceedings of the 2nd Workshop on Managing Technical Debt (MTD '11)*. May 2011. Waikiki, Honolulu, HA (USA). http://dx.doi.org/10.1145/1985362.1985366

URLs are valid as of the publication date of this document.

Assured Design



Effective Reduction of Avoidable Complexity in Embedded Software (ERACES)

Principal Investigators

Dr. Peter Feiler

Peter Feiler, PhD, is the technical lead and author of the SAE AS-2C Architecture Analysis & Design Language (AADL) standard. Version 2.1 of the standard was published in January 2012. He is currently leading the revision of the Error Model Annex standard for AADL. His research work focuses on safety-critical real-time systems, architecture languages software-reliant systems, and predictable system analysis & engineering

Dr. Julien Delange

Julien Delango, PhD, researches software and system architectures. Prior to joining the SEI, he worked as a software engineer at the European Space Agency.

Background

Embedded source code, even models used to auto-generate code, often lack abstraction, as exemplified by the Apache Flight Management Computer (FMC) with 7500 mostly Boolean variables (e.g., 112 Booleans for a history buffer of three commands) and a Rolls Royce Aero Engine Control (RR-AEC) Engine Control System (ECS) with 700 variables.[1,2] Much of the task execution ordering and data flow is hidden, making even simple mistakes hard to discover and leaving behind a costly testing and maintenance legacy. Source code analysis and refactoring have proven to be of limited value due to lack of abstraction and time-sensitivity of embedded software. Model-based development and auto-generation brings limited gain if models have a low level of abstraction.

Our project will develop a tool that applies data and architectural abstractions to existing models and will demonstrate that the tool measurably reduces avoidable complexity [NASA 2009]. Indicators of reduced avoidable complexity are

- fewer user-maintained variables—reducing state space explosion
- defects getting caught by compile-time type checking instead of tests—reducing avoidable rework cost
- fewer test cases and higher effectiveness of Statement Coverage (SC), Decision Coverage (DC), and Modified Condition Decision Coverage (MCDC4) required by DO-178C reducing verification effort [Chilenski 1994]

Approach

We are developing for public release two tools:

- Eclipse-based open source ERACES tool that analyzes software architecture and suggests architecture changes for removing avoidable complexity
- SCADE-based tool that analyzes SCADE models and produces complexity metrics, giving an indication of potential complexity

Our research contribution is to demonstrate measurable reduction of avoidable complexity in embedded software when using models. We have shown that using such abstractions reduces the number of tests and increases system maintainability, while maintaining generated code efficiency.

We have applied the ERACES tools to the RR-AEC ECS AADL and SCADE models of an engine used in Army and civilian aircraft. We have demonstrated that our tools can highlight avoidable complexity areas and suggest potential improvements to avoid them.

In parallel, we have performed a study to understand how and why users are introducing complexity in software models. We have done an experiment with software engineers from different backgrounds and experience levels to understand how they use the tool and why they use complex modeling patterns. The results ultimately help us to understand how to improve modeling language, methods, and tools. This study has been performed in collaboration with ANSYS. This firm is the tool vendor of SCADE, a modeling tool suite that produced certified code for avionics systems. SCADE has been used to design the software of the Airbus A350 and A380.

Artifacts

- ERACES prototypes for architecture abstraction detection methods
- Complexity metrics and complexity detection tool implemented in SCADE system
- Study and report on the use of modeling software and how to improve it
- Case study report of effectiveness of abstraction in reducing avoidable complexity

Future Work

- The initial ERACES tool prototypes will support only SCADE and AADL models and demonstrate the detection of avoidable complexity using a set of patterns. Broader use in DoD and industrial settings would require support of other modeling languages and extended modeling patterns.
- As customers migrate their embedded software to models with higher abstraction, there is an opportunity to deploy model checking to verify that, when using suggested abstraction, the system and execution semantics are not affected.

1 Value-Driven Incremental Development, FY2013/14 research, produced these case studies: RR-AEC ECS Power up Check, Connect evolution, and Apache architecture refactoring.

2 Described in a special, unpublished report on an architecture-centric analysis of the Apache FMCO.

Effective Reduction of Avoidable Complexity in Embedded Systems

Complexity Management in Software Models

Safety-Critical Systems are becoming extremely software-reliant. Software complexity can increase the total acquisition costs as much as 16%. The ERACES project aims to identify and remove complexity in software models, such as SCADE.

Why detect complexity in models?

Safety-critical systems development is shifting from traditional programming (e.g., Ada, C, assembly) to modeling languages (e.g., Simulink, SCADE). Model-Based Engineering (MBE) provides an accurate semantics for system analysis, validation, and automatic code production —reducing development and testing efforts. Parts of the Airbus A380 and A400M planes have been designed using models. Current costs savings estimates show that using models can help save as much as 57% on the development of an avionics system at the highest criticality level (DAL A).

Why complexity in models matters?

Software complexity increases not only development but the overall acquisition costs. As maintenance activity accounts for 70% of the lifecycle costs, reducing complexity of models is of primary importance. As MBE is a new development paradigm, we need new methods and metrics to identify complexity.

What has been done by the SEI?

During the ERACES project, the SEI team focused on these areas:

- Develop complexity metrics in models
- Understand the use of modeling tools
- Estimate the costs of software complexity

Complexity Metrics in Behavior Models

The SEI has been working on applying existing complexity metrics in software models. We selected complexity metrics that have a different focus from:

McCabe: focus on state space

Halstead: focus on operators and operands

Zage: focus on components connections

These metrics have been tested and implemented within the SCADE tool.

The SEI worked on new, model-specific complexity metrics that rely on the specific data-flow semantics of SCADE. This new complexity metrics reports for each flow the related number of operators, operands and outputs. Model designers use this information to update their design and reduce the system complexity with different strategies (e.g., refactoring components, change connections, change interfaces definition). By reducing the number of connections, designers decrease the number of tests (c.f. DO-178C) required to certify the software. These metrics have been implemented in the SCADE tool.

All ERACES plugins and tools are available on the SEI github forge under the BSD license.

Software Architecture Complexity

We identified software architecture patterns that incur unnecessary complexity. Software configuration and deployment policy (e.g., execution rate, communication queues dimensions) impacts system behavior and might have a significant impact (e.g., early/late values, missing values). We developed a method that identified inappropriate software architecture patterns using AADL that might incur complexity and suggest workaround and fixes. Our approach has been tested on a customer project and successfully detected an error related to missing values.

Understanding Complexity

The SEI started an experiment to understand the current vision of complexity in software models by practitioners. We also asked professionals and students to design a model from textual specifications. We found that many users, even experienced ones, have issues using models. When transitioning to a full MBE approach, training becomes the key to success.

Making an impact

The SEI initiated a collaboration with ANSYS, the developer of SCADE, to use the SEI complexity metrics tools and understand the impact of complexity in software models. ANSYS is currently working on integrating these metrics into their products to ultimately help system designers detect potential design issues and improve their models. The SEI has been invited to present the ERACES research project results at the SCADE User Conference to be held in Paris on October 2015.

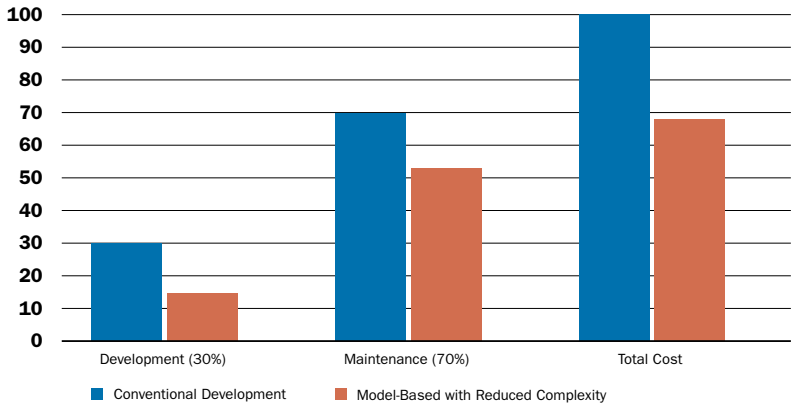
Costs of complexity

The SEI evaluates the cost of complexity when using models. MBE development approaches reduce development and testing efforts, especially for critical systems that require intense testing efforts. Our estimates shows a reduction of cost development of 50% for safety-critical systems when using a MBE development approach that includes a certified code generator.

Reducing software complexity can reduce acquisition costs by more than 30%.

However, inappropriate design decisions can put these savings at risk and spread throughout the software lifecycle. Complexity can increase maintenance costs by 25%. As maintenance costs of safety-critical systems accounts for more than 70% of the total acquisition costs, complexity by itself can increase them by more than 30%.

Model-Based Engineering: Costs Savings Estimates



Model Complexity has a large impact on maintenance activities, which represents at least 70% of the TCO

Contact: Julien Delange jdelange@sei.cmu.edu

Open Source AADL Workbench

Principal Investigators

Dr. Peter Feiler

Peter Feiler, PhD, is the technical lead and author of the SAE AS-2C Architecture Analysis & Design Language (AADL) standard. Version 2.1 of the standard was published in January 2012. He is currently leading the revision of the Error Model Annex standard for AADL. His research work focuses on safety-critical real-time systems, architecture languages software-reliant systems, and predictable system analysis & engineering

Lutz Wrage

Lutz Wrage's research focus includes applications of architectural modeling, issues in Cyber-Physical Systems, and resource allocation in real-time systems.

Background

Mismatched assumptions about hardware, software, and their interactions often result in system problems detected too late in the development lifecycle, which is an expensive and potentially dangerous situation for developers and users of mission- and safety-critical technologies. Currently, best practice in the aerospace industry has led to 80% of embedded software system issues being discovered post-unit-test with post-unit-test software rework accounting for 50% of the total system cost.

To address this problem, SAE International released the Architecture Analysis & Design Language (AADL) standard suite AS5506. The AADL standard defines a modeling notation based on a textual and graphic representation used by development organizations to conduct lightweight, rigorous—yet comparatively inexpensive—analyses of critical real-time factors, such as performance, timeliness, safety, reliability, security, and data integrity. This allows for model-based virtual system integration to discover issues early in and throughout the lifecycle through a combination of model-based analysis and simulation to complement traditional testing.

Since its first release in 2004, AADL has become a platform for industrial pilot projects and university research. For example, the Aerospace Vehicle Systems Institute (AVSI) System Architecture Virtual Integration (SAVI) initiative chose AADL as a key technology to pursue an integrate-then-build approach to reduce leakage of issues to the system integration phase. SAVI membership includes Boeing, Airbus, Embraer, Rockwell Collins, Honeywell, BAE Systems, United Technologies, ANSYS/Esterel, FAA, NASA, and the DoD. The proof-of-concept phase showed the healthy return on investment from using a virtual integration approach to cut rework cost and time considerably.

In addition, the DoD has funded several projects through which an architecture-centric virtual integration practice (ACVIP) is being developed. Virtual integration experiments have been conducted in projects for the Apache, JPL Mission Data System, and CH47F health monitor programs. In addition, the Joint Multi Role (JMR) Technical Demonstration program is piloting ACVIP to show the value of virtual system integration and mature an ACVIP engineering and acquisition practice in preparation of the Future Vertical Lift program.

Approach

The focus of this project is on maturing an open source workbench called the Open Source AADL Tool Environment (OSATE) that supports virtual system integration practices through AADL modeling, analysis, and auto generation, into a workbench that supports the development practices typically used among organizations in the Defense Industrial Base.

OSATE provides a reference implementation of the AADL core language as well as various AADL extensions (annexes) including support for data modeling, behavior modeling, fault modeling, and ARINC653 partitioned systems support. Analytical capabilities include semantic consistency checking of AADL models and its extensions, flow latency analysis, physical and computer resource budgeting, resource allocation, scheduling analysis, safety analyses in support of SAE ARP4761, functional integration consistency analysis, among others. OSATE leverages Eclipse capabilities to provide distributed model repository and team support.

OSATE has become a prototyping and transition platform for a number of international research groups. It recently has been used to demonstrate the effectiveness of structural and behavioral model checking to greatly reduce the risk of security intrusion on an unmanned air vehicle in the DARPA High-Assurance Cyber Military System (HACMS) program.

We have integrated capabilities developed by others that have been shown to be valuable for ACVIP, including the Resolute and Agree model checking capabilities by Rockwell Collins used in the HACMS program and other projects. This capability provides architecture-led contract-based compositional analysis and verification. We have enhanced flow latency analysis to take into account latency contributions of ARINC653 partitioning-based architectures and support for latency impact analysis of architecture design tradeoffs. We have improved architecture-led safety analysis capabilities including an open source fault tree analysis capability that includes cut set support. We have incorporated additional scheduling analysis capabilities. We have integrated capabilities for consistency verification of ARINC653 based AADL models and for generation of AADL specific runtime executives and configuration files for ARINC653 based architectures for VX Works and DEOS real-time operating systems. We have added support for importing SCADE and Simulink models into AADL models. Finally, we have made improvements in usability through improved context-sensitive help, a graphical editor, filtered navigation views, and guided workflows for several of the modeling and analysis processes.

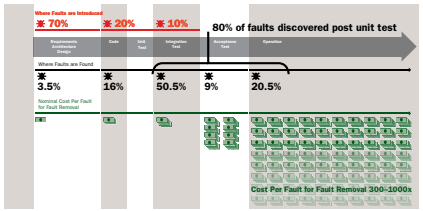
Artifacts

Public stable release of OSATE (2.1.1), an open-source tool platform to support AADL. (Available at https://github.com/osate).

AADL Workbench for Virtual System Integration

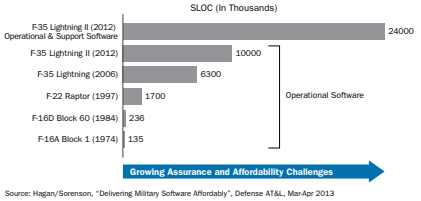
Safety and Mission Critical System Challenge

The traditional development lifecycle using existing methods of system engineering are not working for the latest generation of systems being developed. Requirements and architecture design introduce 70% of system issues, while 80% are discovered post unit test, when they are exponentially more expensive to fix.



Much of the growth in total system cost is interaction complexity and mismatched assumptions in embedded software, making systems increasingly unaffordable.

DoD Capabilities through Software

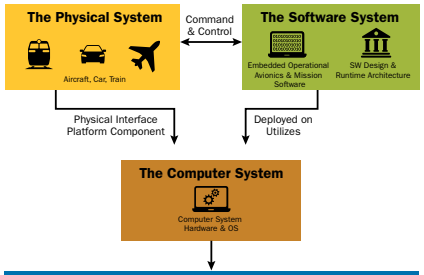


Source: Hagan/Sorenson, "Delivering Military Software Affordably", Defense AT&L, Mar-Apr 2013

Software as % of total system cost
1997: 45% → 2010: 66% → 2024: 88%

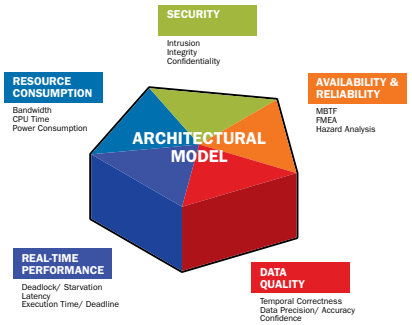
Virtual System Integration with SAE AADL

The SAE International AS-5506 Architecture Analysis & Design Language (AADL) standard suite has been developed to address this challenge through virtual system integration and analysis to discover system-level issues earlier in the life cycle.



AADL focuses on interaction between the three elements of a software-reliant mission and safety-critical systems

An Analyzable Architecture Modeling Notation. Well-defined timing semantics of a task and communication architecture deployed on distributed platforms, modeling of virtual channels, partitions, operational modes, end-to-end flows, fault behavior, and security characteristics lead to multi-dimensional analysis of virtually integrated systems and discovery of system level issues early in the lifecycle.



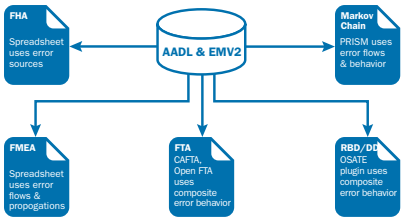
The Open Source AADL Workbench

The Open Source AADL Tool Environment (OSATE) provides a reference implementation of the SAE AADL standard suite notation and a prototyping platform for advancing research in architecture-centric system analysis and verification.

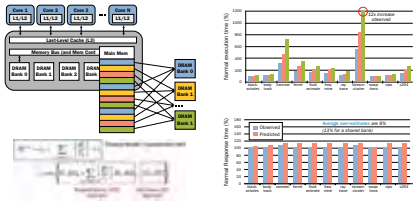
Open Source AADL Tool Environment (OSATE)

Modeling Capabilities	Analysis Capabilities	Usability Capabilities	External Contributions
AADL	Resource Budget	Context sensitive help	Resolute
EMV2	Latency	Role specific workflow	Agree
BA	Safety	Expanded Navig. Views	Ocarina Code Generation DeOS, VxWorks
Type Consistency	RMS/EDF Scheduling	Graphical Editor	MAST Scheduling
Semantic Consistency	Resource Allocation		Independent contributions
Team Mgmt	Functional Integration		
	ARINC653 Support		

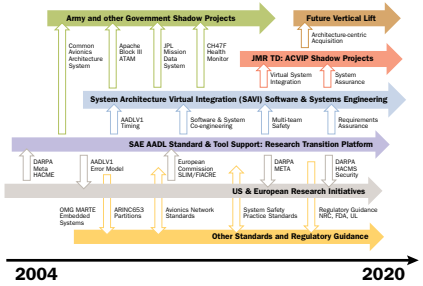
Support of SAE ARP4761 System Safety Assessment Practice



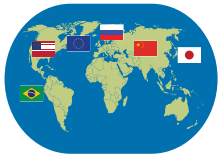
Rate Monotonic with Memory Partitioning



Towards an Architecture-centric Virtual Integation Practice (ACVIP)

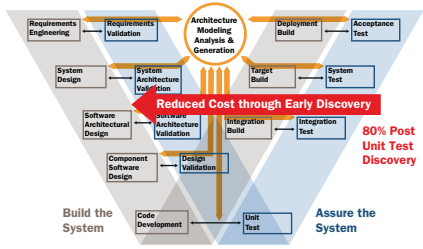


International investment and engagement



A Key Technology in the System Architecture Virtual Integration (SAVI) initiative by an international Aerospace industry consortium. Proof of concept demonstrations, return on investment, technology maturation, pilot applications, and process adapation in a multi-year self-funded effort.

Early Discovery through Virtual System Integration



Contact: Peter Feiler phf@sei.cmu.edu

Extending AADL for Security Design Assurance of the Internet of Things (IoT)

Principal Investigators

Dr. Carol Woody

Carol Woody, PhD, leads research into cybersecurity engineering: building capabilities in defining, acquiring, developing, measuring, managing, and sustaining secure software for highly complex networked systems as well as systems of systems.

Dr. Rick Kazman

Rick Kazman, PhD, does research in the areas of software architecture, software engineering ecomomics, design and analysis tools, and software visualization.

Background

Safety-critical verification of Cyber-Physical Systems (CPS) has benefited from the use of architecture fault modeling capabilities provided by the Architecture Analysis & Design Language (AADL). Architecture-led hazard analysis using architecture description languages (ADLs) such as AADL has become an effective capability in safety fault management. The cost of successfully addressing safety compliance has been greatly reduced through the use of extensions to AADL that automate safety analysis and produce safety assessment reports to meet recommended practice standards (such as SAE ARP4761) [Cervin 2006]. We can leverage these successful capabilities to the security domain especially the automated analysis of security faults [Firesmith 2003].

The SEI has used AADL to effectively address design verification for the qualities of safety, reliability, and performance [Delange 2014, Feiler 2009, and Lewis 2010]. We propose adding security analysis capabilities into AADL that would leverage existing safety and reliability analysis capabilities including automated fault-tree analysis. New rules or language extensions would be developed to address gaps. Based on our review of security vulnerability attacks assembled in an AFRL study, we have observed that current descriptions of vulnerabilities are focused on coding [Calloni 2011].

Approach

We were aware that no one had applied AADL to the problems of modeling authentication and input validation, two key security capabilities. Our initial plan to abstract the AFRL patterns into the design characteristics and constraints proved unsatisfactory because AADL is a static representation and the security patterns assumed a dynamic model, based on a description workflow. Instead, we focused on extending AADL's modeling capabilities by describing the design characteristics needed to model, analyze, and assure security based on a set of threats.

We identified an automotive infotainment example where critical interfaces between the infotainment system and other car functions such as brakes and ignition could be impacted by an attack. We instantiated the Microsoft STRIDE threats as specific attacks on an infotainment system.[1] We therefore created a description in AADL that contained enough architectural properties to demonstrate that each of the threat concerns was addressed. We used the AADL model to demonstrate how existing AADL capabilities could be leveraged to evaluate two of the six threats types. Through this example we were able to describe the kinds of properties an architect should reason about to address expected security concerns for CPS.

Task 1: Enhance AADL to Incorporate Security Design Assurance

For each STRIDE threat scenario, we identified the capabilities, constraints, and other design characteristics that reduce the risk of design vulnerabilities and mapped these to AADL to build a security analysis capability.

Task 2: Evaluating the Secure Code Generation Capabilities of AADL

There are two parts to an application generated from AADL: the runtime system, which is the same for each application, and the generated code that is unique to that application.

We performed a security analysis of an AADL code generator's C runtime system using secure coding standards. This involved sending the code through the Source Code Analysis Lab (SCALe) tool and determining what would need to be fixed to bring it into compliance with secure coding practices.[2]

Artifacts

- Design approach for applying AADL formal detail design modeling capabilities to address common security threat scenarios
- Detailed example model for the use of AADL to address two important security concerns: confidentiality of data and elevation of privileges
- Report of the secure coding standards gaps in the AADL code generator's C runtime system
- Paper describing the approach for applying formal detailed design modeling to supply chain security risk scenarios

Future Work

- Extend the AADL example to include consideration of additional threat scenarios such as spoofing and tampering
- Evaluate code generators in other languages with secure coding standards (e.g., Java)

1 STRIDE is spoofing, tampering, repudiation, information disclosure, denial of service, and elevation of privilege.

2 SCALe, developed by the SEI's CERT Division, is used to analyze various code bases for conformance to secure coding standards.

Extending AADL for Security Design Assurance of the Internet of Things

Important decisions that establish security in a system are made in the architecture.

Formal modeling provides a means to continually verify that design and code changes are consistent with security requirements.

We extended the core modeling concepts of AADL with security properties, to formally model architectural properties relevant to security (e.g., AccessMode, AccessGroup).

To drive the analysis we first established a set of *threats*. Our threat analysis was guided by Microsoft's STRIDE (Spoofing, Tampering, Repudiation, Information disclosure, Denial of service, and Elevation of privilege) model.

To reason about the satisfaction of security properties in a system architecture we need to:

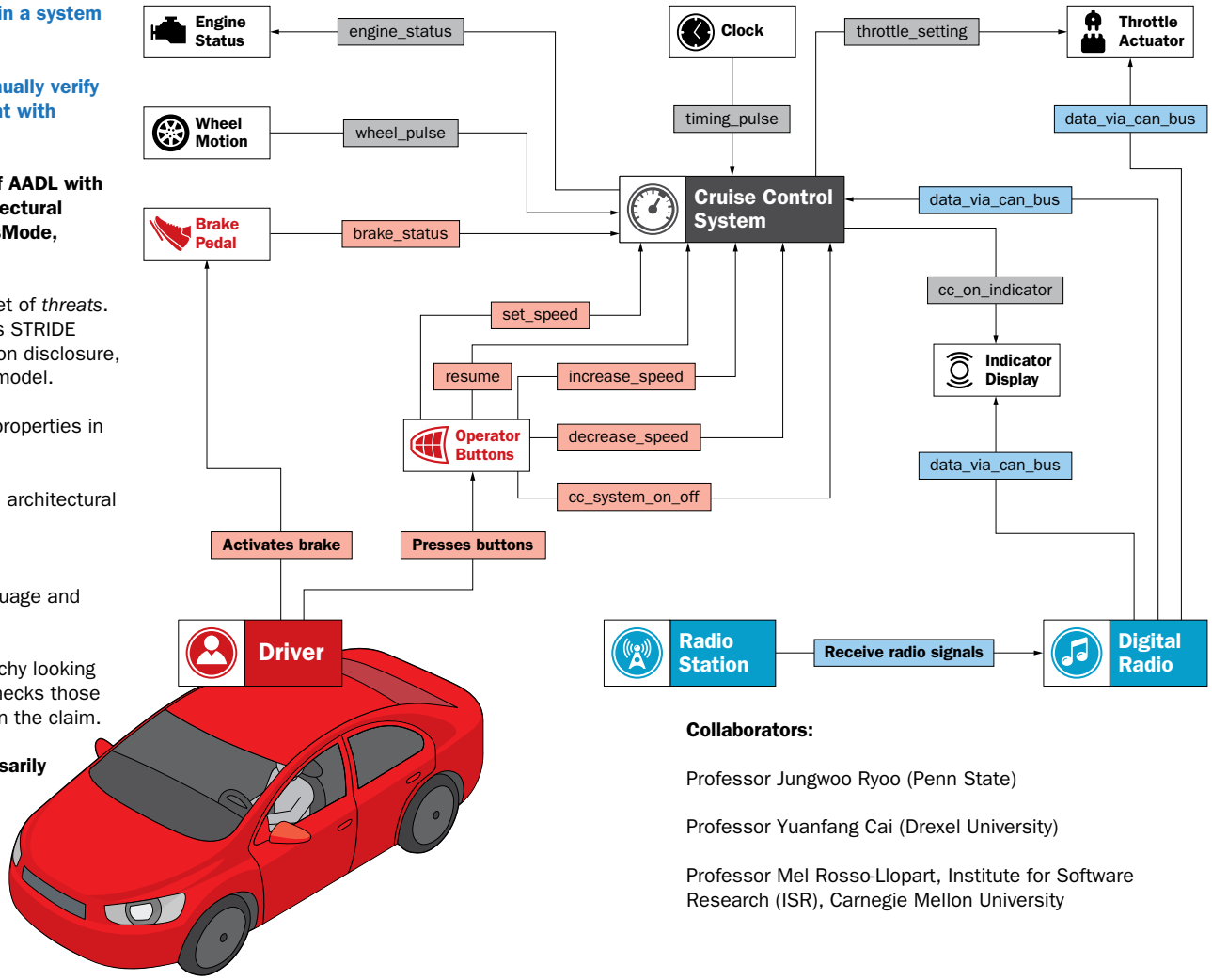
1. Specify the properties, and the associated architectural elements in AADL; and then
2. Analyze claims over those properties.

To analyze claims we used the Resolute language and model-checker.

Resolute walks the instantiated model hierarchy looking for components specified in its claims and checks those components according to the logic encoded in the claim.

Limitations: An AADL model does not necessarily support security validation. Is the system sufficiently secure for the planned usage? Possibilities:

- **An external weakness may enable an attacker to operate outside the model.**
- **The specifications may not provide the desired level of security assurance.**



Collaborators:

Professor Jungwoo Ryoo (Penn State)

Professor Yuanfang Cai (Drexel University)

Professor Mel Rosso-Llopart, Institute for Software Research (ISR), Carnegie Mellon University

Contact: Rick Kazman and Carol Woody rkazman@sei.cmu.edu, cwoody@cert.org

Increase Adoption of Secure Coding Standards

Principal Investigator

Daniel Plakosh

Daniel Plakosh's principal areas of expertise include real-time distributed systems, network communications and protocols, systems engineering, real-time 2D and 3D graphics, and Unix OS internals. Much of his recent experience has been redesigning legacy distributed systems to use the latest distributed communication technologies

Background

Coding standards are an integral part of the software development lifecycle and increasingly a requirement. The NDAA for FY2013, Section 933, requires evidence that government software development and maintenance organizations, including contractors, are conforming to DoD-identified "best practices, tools, and standards for developing and validating assured software" during software development, upgrade, and maintenance activities.

We see four challenges to fulfilling this mandate:

1. Well-specified secure coding standards must be developed for ubiquitous languages that do not have them. Where possible, the secure coding standards need to be published by international standards bodies that influence the software development toolchain to allow easy adoption by the DoD supply chain.
2. The number of actual rule violations and false positives discovered in conformance testing is excessive and must be reduced to levels that can be reasonably addressed by development teams.
3. It must be demonstrated that the adoption of secure coding standards will not degrade system performance and result in slow, bloated code.
4. New rules for underspecified language features such as C threading must be developed to augment existing rules.

Approach

To address the lack of secure coding standards, we will complete the CERT C++ Secure Coding Standard. C++ is used extensively throughout the DoD, including for major weapons systems such as the Joint Strike Fighter. Existing C++ coding standards fail to address security, subset the language (e.g., MISRA C++:2008), or are outdated and unprofessional (e.g., C++ Coding Standard referenced in DISA's Application Security and Development STIG).[1] The CERT C++ Secure Coding Standard, although publicly available on a wiki, has lagged far behind the C efforts and needs to be substantially overhauled.[2]

To address the problem of excessive rule violations, we will collaborate with Clang developers from Apple and JPCERT[3] to develop additional analyses for Clang's static analyzer to check for violations of a prioritized list of secure coding rules. Clang is an open-source compiler that has been integrated into Apple's XCode IDE, which is the primary tool for developing software for iOS and OS X.

We will also collaborate with FindBugs and Oracle to develop analysis against unchecked guidelines in T*he CERT Oracle Secure Coding Standard for Java* and to integrate this analysis into Eclipse and/or Oracle JDeveloper, so that analysis results are immediately available to Java developers—including Android developers.

We will also increase the precision of the DidFail Android app static analyzer, to reduce the number of violations found in conformance testing during Android app development.

Artifacts

- Clang analysis integrated into Apple's XCode IDE
- FindBugs analysis integrated into Eclipse or JDeveloper
- Publication of enhanced versions of DidFail
- Static analysis incorporated into CERT's SCALe
- List of underspecified aspects of threads in C, as reference for future rules and standards
- Research papers and a case study

Future Work

- Development of additional secure coding rules for existing secure coding standards
- Coverage for other languages and platforms
- Additional analysis capabilities

1 MISRA is the Motor Industry Software Reliability Association; DISA is the Defense Information Systems Agency; and a STIG is a Security Technical Implementation Guide.

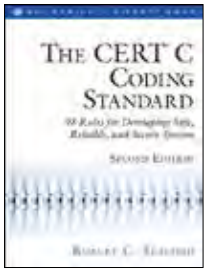
2 For more information, please visit the wiki at https://www.securecoding.cert.org/confluence/pages/viewpage.action?pageId=637

3 Japan Computer Emergency Response Team

Increasing Adoption of Secure Coding

Coding standards are an integral part of the secure software development lifecycle and increasingly a requirement. Proper coding results in fewer weakness, fewer vulnerabilities, and reduced costs for cyber protection. This research provides the foundational prescriptive rules as well as practical support for putting the rules into practice.

The Definitive Reference for C Programs



The definitive reference for prescriptive guidance in writing C programs

CERT Bimonthly Secure Coding eNewsletter

Over 1,600 contributors provide input and updates to the coding rules wiki.

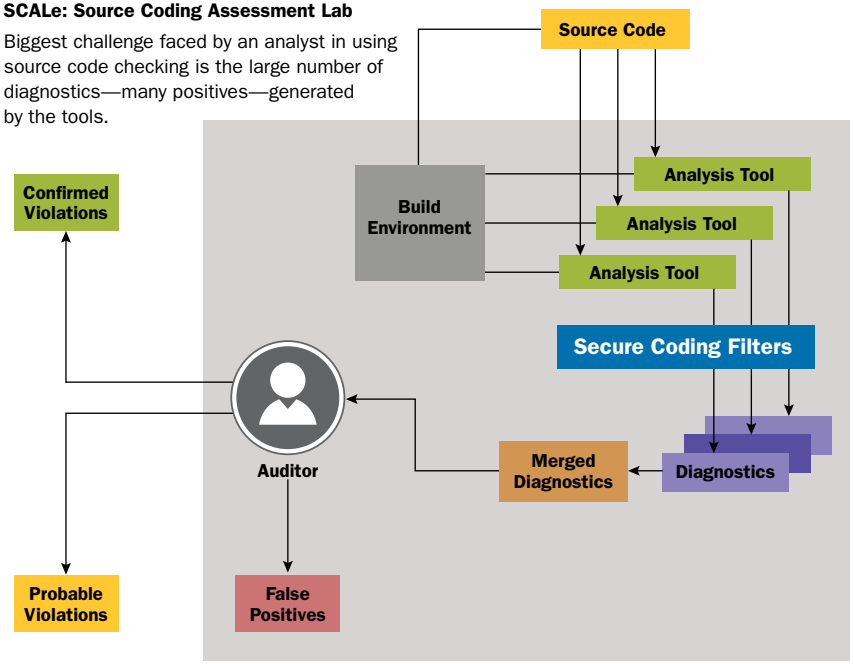
Providing the foundational rules for coding. The project's research has provided foundational rules for programming in C and Java.

The current project extends the rules to encompass new programming models—threads—and the updated C/C++ language:

- 25 new rules in FY15 specifying C and C++ weaknesses
- 99 rules dedicated to C-specific weaknesses
- 74 rules dedicated to C++-specific weaknesses (148 total rules when including overlapping C rules)
- 9 new unspecified behaviors in C threads

SCALe: Source Coding Assessment Lab

Biggest challenge faced by an analyst in using source code checking is the large number of diagnostics—many positives—generated by the tools.



SCALe makes expert review more productive by focusing on high priority violations out of the volume of diagnostics provided by tools.

- Filter select secure coding rule violations
- Eliminate irrelevant diagnostics
- Convert to common CERT Secure Coding rule labeling
- Provide single view into code and all diagnostics

SCALe maintains a record of decisions so that results about a review are maintained as code is changed and not revisited.

Integrating checkers into Integrated Development Environments (IDE)

Providing immediate feedback to developers about weaknesses in programs as code is being developed encourages secure coding without generating vast collections of diagnostics later in the SDLC.

Checking C/C++ rule violations

- Exception
- Evaluation ordering
- Function return
- Constructor
- Assertion

Checking Java rule violations

- Override
- I/O

New rules for threaded programs and C++ form a more secure underpinning for object-oriented software deployed on multithreaded architectures. New checkers can locate potential vulnerabilities, and, when integrated into IDEs, provide immediate and effective feedback to developers.

Contact: Daniel Plakosh dplakosh@cert.org

Graph Algorithms on Future Architectures

Principal Investigator

Dr. Scott McMillan

Scott McMillan, PhD, is a Senior Software Developer at the SEI, with experience in high-performance computing across a wide range of application areas including scientific computing, 2D/3D geographic information systems (GIS), link analysis, and very large-scale database systems.

Background

Since June 2013, 4 of the top 10 supercomputers on the Top500 benchmark list are Heterogeneous High-Performance Computing (HHPC) systems using either NVIDIA GPUs or Intel Xeon Phis. At the same time, heterogeneous supercomputing systems hold all 10 top spots of the Green500 benchmark that measures performance-to-power ratios. As supercomputers get larger, HHPC architectures and their attractive energy efficiencies are becoming more important to our customers.

However, because graph algorithms typically exhibit small computation-to-communication ratios and have irregular memory access patterns, it is very difficult and costly to implement them effectively on HHPC systems. As a result, these coprocessors often sit idle during graph processing. Yet, there is a growing body of research on graph analytics on these "future" architectures [Hong 2011, Munguia 2012, and Mastrostefano 2013].

Providing an easy way to leverage and expand on this body of knowledge would benefit many problem domains relevant to the DoD and the Intelligence Community (IC) such as social network analysis (ISR), cybersecurity (computer network modeling and analysis), system analysis and support (net-centric infrastructures), and command and control.

Approach

We are continuing our collaboration (funded) with Andrew Lumsdaine's research group at the Center for Research in Exascale Technologies (CREST) at Indiana University.[1] They have demonstrated the power of their architecture and abstractions by extending Boost Graph Library (BGL) for distributed-memory CPU systems with the Parallel Boost Graph Library (PBGL) [Gregor 2005]. Our focus is to leverage this experience, to determine the abstractions needed to support a more complete set of graph algorithms determined during in our FY2014 research on heterogeneous hardware and specifically GPUs.

Our research goal is to derive a specification for graph algorithm primitives that will represent the *separation of concerns* between lower-level implementations for specific hardware architectures and higher-level graph analytics concepts. This "interface" is shown by the dotted line in the following figure and is similar to what was accomplished in the scientific computing community with NIST's Basic Linear Algebra Subprograms (BLAS) specification [BLAS 2002]. Recently, our team joined with a group of graph analytics experts from across government, academia, and industry to develop the GraphBLAS specification that will represent this separation of concerns and support more heterogeneous architectures.

The software architecture for graph analytics (above, right) is similar to the BLAS approach for scientific and engineering simulation codes (above, left).

Artifacts

- A proposed specification/standard for graph algorithm primitives (eventually GraphBLAS)
- Open source graph library supporting CPU with or without GPU architectures
- Graph500 benchmark submissions using the abstractions
- Conference paper submission to appropriate venues (SC, ISC, IPDPS)

Future Work

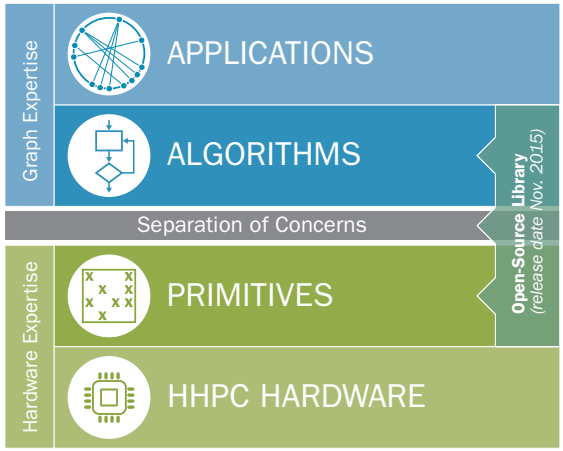
Our current work focuses on GPUs, the most prevalent coprocessor in HHPC systems. Future research will investigate support for other coprocessors, such as Xeon Phi and FPGA, and the development of additional algorithms that are built on these primitives.[2] The GraphBLAS specification effort will continue beyond the end of this project.

1 Andrew Lumsdaine is one of the top graph analytics experts in the world. His research and interests with the development of PGBL aligns with the background needed for this research.

2 FPGA is Field Programmable Gate Array.

Graph Algorithms on Future Architectures

Fast, efficient graph analysis is important and pervasive. However, achieving high levels of performance is exceedingly difficult especially in the era of complex heterogeneous high-performance computing (HHPC) architectures. By defining a set of graph primitives and operations, we are able to separate the concerns between the graph expertise needed to develop advanced graph analytics and the hardware expertise needed to achieve high levels of performance on the ever-increasing complexity of the underlying hardware.



The software architecture: the abstraction layer (gray) separates the concerns between the expertise needed to develop graph algorithms and applications (above), and intimate knowledge of the hardware needed for high performance (below).

Overview. For the last two years, the members of the Emerging Technology Center at the SEI have been collaborating with graph analytics experts at Indiana University to identify and implement a set of primitives and operations to separate the concerns between graph analytics development and the increasing complexity of programming for the underlying hardware. During that time, we have joined with other leading experts from industry, academia and government to create an application programming interface (API) standard, now called the GraphBLAS (Graph Basic Linear Algebra Subprograms), that will capture this separation of concerns (http://graphblas.org).

Graph Algorithms: Simplified by GraphBLAS API

Algorithms Implemented with Less Code. We are developing a library of graph algorithms that are implemented in terms of the new operations and data primitives currently defined by the GraphBLAS API. Classes of algorithms include

- Metrics: e.g., degree, diameter, centrality, triangle counting
- Traversals: Breadth-First Search (BFS)
- Shortest Path/Cost Minimization
- Community Detection/Clustering
- Connected Components
- (Minimum) Spanning Tree
- Maximum Flow
- PageRank

Separation of Concerns: GraphBLAS API Spec

Standardization In Progress. Researchers from the SEI, industry, academia, and the U.S. government are developing the API specification:

- The mathematical properties are defined by semi-ring algebra.
- Nine operations are specified currently (see right).
- The key primitive type is the sparse matrix.
- We are exploring extensions to this set of operations that can offer greater expressivity and greater opportunities for tuning.

Tuning the sparse matrix multiplies (MxM, MxV, VxM) is key to achieving performance on underlying hardware. Many different sparse formats already exist, and the "best" format depends on both the underlying hardware architecture and operation performed.

Graph Primitives: Tuned for GPU Architectures

Collaboration with Indiana University. Researchers including Andrew Lumsdaine from the Center for Research in Exascale Technologies have been collaborating with the SEI on this project to explore efficient implementations of graph primitives. The graph at the right shows the performance of our BFS algorithm (orange) using a compressed, sparse row matrix format on the newest generation of GPU cards using dynamic parallelism. It is compared to best-in-class implementations reported in the literature.

Future Work.

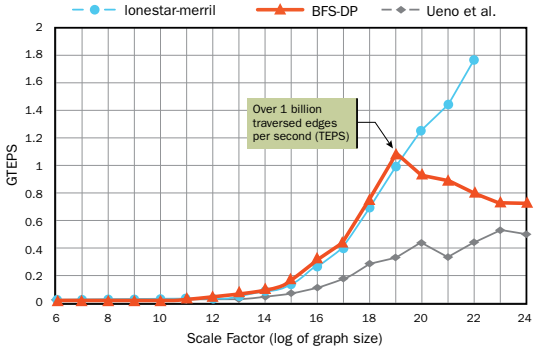
- Continued participation in the GraphBLAS standardization effort
- Addressing scaling issues for larger graphs
- Developing distributed primitives to support multiple GPU nodes.
- Tuning for a variety of different sparse matrix formats that will be required for high performance across a wide range of algorithms
- Future versions that include sparse solvers to support other important algorithms (e.g., PCA, graph partitioning)

```
void bfs(SparseMatrix<bool> const &graph,
         Vector<bool>              wavefront,
         Vector<int>               &level)
{
   visited = wavefront;
   level_val = 0;

   while (!wavefront.empty()) {
      // traverse one level from current wavefront
      wavefront   = VxM(wavefront, graph, LogicalSemiring);

      // compute which nodes have NOT been visited before
      not_visited = Apply(visited, LogicalNot);
      wavefront   = EWiseMult(not_visited, wavefront, LogicalAnd);
      visited     = EwiseAdd(visited, wavefront, LogicalOr);

      // Assign the level to all newly visited vertices
      level_val++;
      level      += EwiseMult(wavefront, level_val, Mutiply);
   }
}
```

The BFS algorithm implemented using only five GraphBLAS operations. With the masking extension proposed for matrix multiplies, BFS could be implemented with only three operations.

Operation Name	Description
BuildMatrix	Build a sparse matrix from row, column, value tuples
ExtractTuples	Extract the row, column, value tuples from a sparse matrix
MxM, MxV, VxM	Perform sparse matrix *multiplication* (e.g., BFS traversal)
Extract	Extract a sub-matrix from a larger matrix (e.g., sub-graph selection)
Assign	Assign to a sub-matrix of a larger matrix (e.g., sub-graph assignment)
EwiseAdd, EwiseMult	Element-wise *addition* and *multiplication* of matrices (e.g., graph union, intersection)
Apply	Apply *unary function* to each element of matrix (e.g., edge weight modification)
Reduce	*Reduce* along columns or rows of matrices (vertex degree)
Transpose	Swaps the rows and columns of a sparse matrix (e.g., reverse directed edges)

The principle GraphBLAS operations (as of 9/17/2015).



BFS performance reported by P. Zhang, et al., "Dynamic Parallelism for Simple and Efficient GPU Graph Algorithms," submitted to 5th IEEE Workshop on Irregular Applications: Architectures and Algorithms, Nov 2015.

Contact: Scott McMillan smcmillan@sei.cmu.edu

Assured Design References

[BLAS 2002]
Wikimedia Foundation. Basic Linear Algebra Subprograms. *Wikipedia*. http://en.wikipedia.org/wiki/Basic_Linear_Algebra_Subprograms. April 2014.

[Calloni 2011]
Calloni, B. et al. *Embedded Information Systems Technology Support (EISTS) Task Order 0006: Vulnerability Path Analysis and Demonstration (VPAD). Volume 2: White Box Definitions of Software Fault Patterns.* AFRL-RY-WP-TR-2012-0111, v2. Air Force Research Laboratory Sensors Directorate. December 2011. http://www.dtic.mil/cgi-bin/GetTRDoc?Location=U2&doc=GetTRDoc.pdf&AD=ADB381215

[Cervin 2006]
Cervin, A. & Lund U. Impact of Scheduler Choice on Controller Stability. *IEEE International Conference on Control Applications/IEEE International Symposium on Computer-Aided Control Systems Design/IEEE International Symposium on Intelligent Control (2006 CCA/CACSD/ISIC)*. Munich, Germany, October 2006.

[Chilenski 1994]
Chilenski, J. & Miller, S. Applicability of Modified Condition/Decision Coverage to Software Testing. *Software Engineering Journal*. Volume 9. Number 5. September 1994. Pages 193-200.

[Delange 2014]
Delange, J. ARP4761 - Wheel Brake System (WBS) Example. *AADL Wiki*. January 28, 2014. https://wiki.sei.cmu.edu/aadl/index.php/ARP4761_-_Wheel_Brake_System_%28WBS%29_Example

[Feiler 2009]
Feiler, Peter et al. *System Architecture Virtual Integration: An Industrial Case Study*. CMU/SEI-2009-TR-017. Software Engineering Institute, Carnegie Mellon University. 2009. http://resources.sei.cmu.edu/library/asset-view.cfm?AssetID=9145; System Architecture Virtual Integration - SAVI. http://savi.avsi.aero/

[Firesmith 2003]
Firesmith, D. *Common Concepts Underlying Safety, Security, and Survivability Engineering*. CMU/SEI-2003-TN-033. Software Engineering Institute, Carnegie Mellon University. 2003. http://resources.sei.cmu.edu/library/asset-view.cfm?assetid=6553

[Gregor 2005]
Gregor, Douglas & Lumsdaine, Andrew. *The Parallel BGL: A Generic Library for Distributed Graph Computations*. July 2005. http://www.osl.iu.edu/publications/prints/2005/Gregor:POOSC:2005.pdf

[Hong 2011]
Hong, Sungpack et al. Accelerating CUDA graph algorithms at maximum warp. *ACM SIGPLAN Notices*. Volume 46. Number 8. 2011. Pages 267-276.

[Lewis 2010]
Lewis, Bruce et al. Architectural Modeling to Verify Security and Nonfunctional Behavior. *IEEE Security & Privacy*. Volume 8. Number 1. 2010. Pages 43-49.

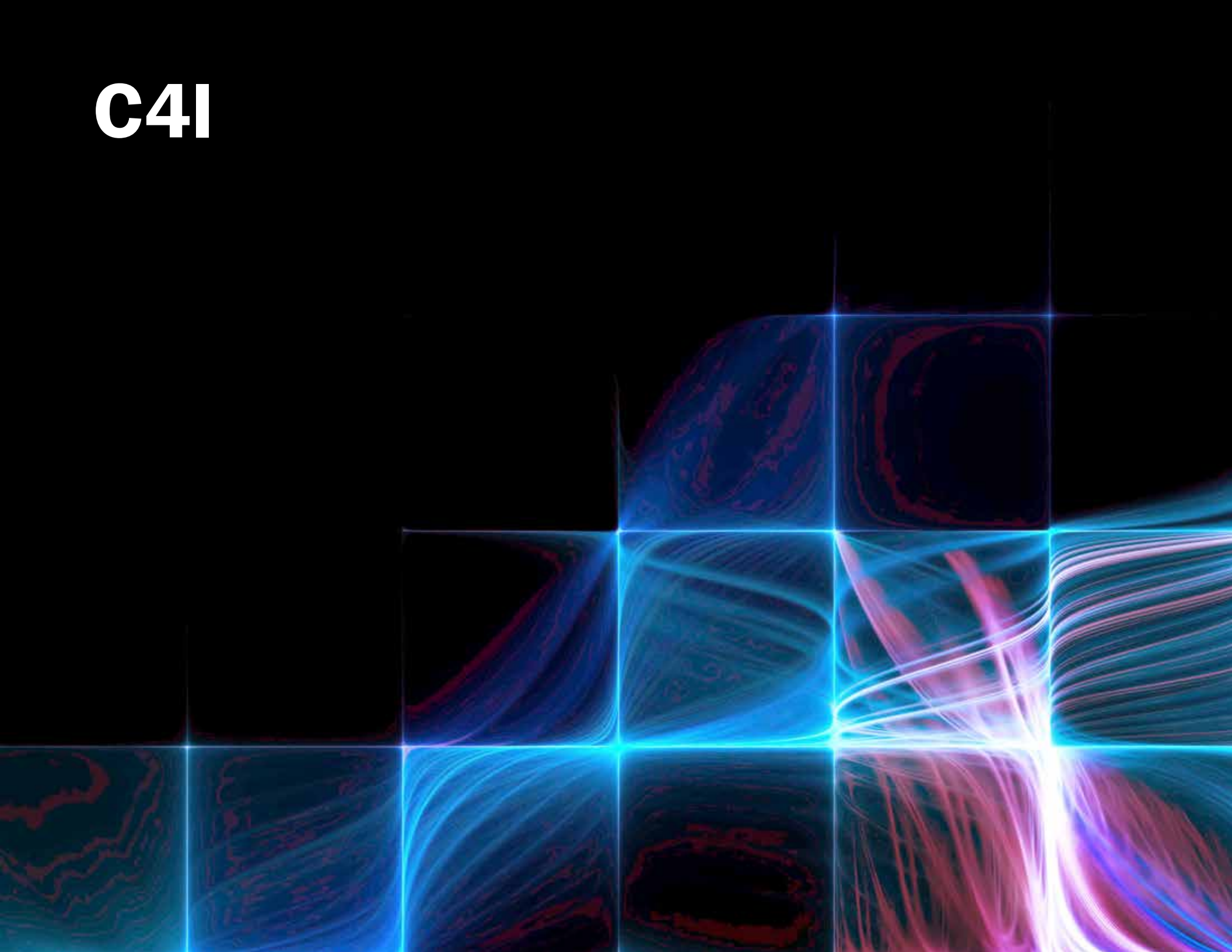
[Mastrostefano 2013]
Mastrostefano, Enrico & Bernaschi, Massimo. Efficient breadth first search on multi-GPU systems. *Journal of Parallel and Distributed Computing*. Volume 73. Number 9. Pages 1292-1305.

[Munguia 2012]
Munguia, Lluis-Miquel et al. Task-based parallel breadth-first search in heterogeneous environments. Pages 1-10. In *Proceedings of the 19th International Conference on High Performance Computing (HiPC).* Pune, India. December 2012.

[NASA 2009]
NASA Office of Chief Engineer. *NASA Study on Flight Software Complexity*. 2009. http://www.nasa.gov/pdf/418878main_FSWC_Final_Report.pdf.

URLs are valid as of the publication date of this document.

C4I



Edge-Enabled Tactical Systems

Principal Investigators

Dr. Jeffrey L. Boleng

Jeffrey L. Boleng, PhD, is a principal researcher who focuses on innovative applications of advanced technologies to aid the safety and effectiveness of soldiers at the tactical edge. He is a co-principal investigator for the Edge-Enabled Tactical Systems (EETS) research team that is pursuing a number of projects related to mobile systems at the edge.

Grace Lewis

Grace A. Lewis is the deputy lead for the SEI Advanced Mobile Systems (AMS) initiative and a co-principal investigator for the Edge-Enabled Tactical Systems (EETS) research team that is pursuing a number of projects related to mobile systems at the edge. Her main interests are Mobile Computing, Cloud Computing, Software Architecture, and Service-Oriented Architecture.

Background

Edge-Enabled Tactical Systems (EETS) adapts, extends, and innovatively investigates architectures and technologies that provide efficient and easily deployable mobile solutions for teams operating in edge environments. Edge environments are characterized by dynamic context, limited computing resources, high levels of stress, and poor network connectivity.

FY2013-14 outcomes from the EETS project have been validated technically and operationally[1] and have generated considerable stakeholder interest, which portends to provide greatly needed capability in edge environments that is not provided by commercially available solutions.

In our current work, a focus on assurance is the factor that our stakeholders have indicated is necessary for our research results to be widely adoptable and to realize their potential in the DoD.

Approach

We address our goal of efficient and trusted integration between the edge and the enterprise in the following three activities.

1. Trusted Nodes: Establishing Trusted Identities in Disconnected Tactical Environments

A common solution for establishing trust between two nodes is to use a third-party online trusted authority that validates the credentials of the requester or a certificate repository. However, the characteristics of tactical edge environments do not consistently provide access to that third-party authority or certificate repository. In FY2015, we investigated and developed techniques to establish trust between nodes in disconnected tactical environments.

2. Trusted Information: Assigning Credibility Scores to Social Media Streams in Real-Time

Trust in the credibility of information provided by social media channels is a challenge because the quality and reliability of information is not uniform due to spam, surreptitious advertising, false rumors, and impostor or compromised accounts. Tactical environments pose additional time constraints on establishing credibility—computation must be fast enough to support the tactical users even if the confidence values are approximate or probabilistic. Current research on establishing credibility is not focused on execution in tactical timeframes and use a single technique. We developed an algorithm capable of executing in tactical time frames (i.e., seconds) that uses an ensemble of existing algorithms (such as supervised machine learning, social network analysis, maximum likelihood estimation, and information diffusion) to assign a credibility score and provide a user-friendly chain of reasoning to streaming social media data.

3. Trusted Information: Fusion of Social and Physical Sensor Data

Most current situational awareness (SA) systems overlay but do not correlate data from complementary sensor classes, specifically physical sensors and social sensors. We can increase trust in situational awareness by fusing data from these complementary sensors. However, these sensor streams are orthogonal (machine-readable vs. human-understandable) and operate at different levels of abstraction, and typical algorithms for extracting meaning from sensed data are computationally-expensive and slow. In FY2015 we developed a fusion strategy that relates open source intelligence from Twitter streams with images, audio, and potentially other data sources derived from opportunistic access to handheld devices.

Artifacts

- Protocol based on identity-based encryption (IBE) and out-of-band channels for secure communication between tactical edge nodes
- Algorithm for credibility score and chain of reasoning for streaming social media data
- Prototype infrastructure and software for fusion of Twitter streams with images, audio and data derived from opportunistic access to handheld devices
- Group context aware data model extensions, Delay-Tolerant Networking (DTN) integration, and reliable User Datagram Protocol (UDP) transport for assured delivery of situational awareness data in challenged tactical radio networks
- Demos and source code that validate and incorporate research results
- Refereed papers and journal articles

Future Work

- For FY2016 the work in EETS has been divided into two projects
  - Tactical Computing and Communications (TCC) focuses on efficient and secure computing and communications for teams operating in tactical environments: extension of trusted identities to server clusters and data at rest; integration of DTN for mobile device to cloudlet to cloud communications; refinement of application layer reliable UDP transport; and architectures for data staging at the tactical edge.
  - Tactical Analytics (TA) focuses on innovative capabilities for data-to-decision in tactical environments: enhanced queueing, tipping, and fusion of sensor data; real-time credibility scoring; proactive and transfer learning for recognizing rapidly emerging events; and identification of storylines in streaming media data.

1 The project team produced 11 publications, 3 prototypes, 7 demonstrations, 2 workshops, and 4 keynote/invited talks.

Edge-Enabled Tactical Systems

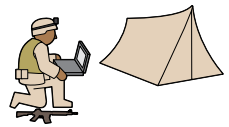
Edge environments are characterized by dynamic context, limited computing resources, high levels of stress, and poor network connectivity.

Edge-Enabled Tactical Systems (EETS) adapts, extends, and innovatively investigates architectures and technologies that provide efficient and easily deployable mobile solutions for teams operating in edge environments.

Goal for FY15: Efficient and trusted integration between the edge and the enterprise

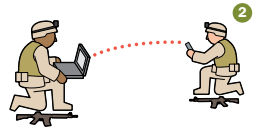
Trusted Nodes: Establishing Trusted Identities in Disconnected Tactical Environments

Method and prototype to establish trust between mobile devices and cloudlets in disconnected tactical environments



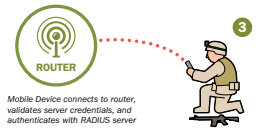
Admin logs into the Cloudlet Manager to start the Bootstrapping process

Step 1: Bootstrapping
- Generation of *Server Credentials* using IBE (Identity-Based Encryption)
- Setup of *RADIUS Server* with *Server Credentials*



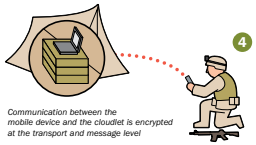
User connects mobile device to the cloudlet, and upon visual confirmation the admin starts the pairing process

Step 2: Pairing
- Generation of *Device Credentials* using IBE
- Transfer to device using Bluetooth or USB, plus visual confirmation
- Transfer to *RADIUS Server*



Mobile Device connects to router, validates server credentials, and authenticates with RADIUS server

Step 3: WiFi Authentication
RADIUS Server implements
Wi-Fi WPA2-Enterprise
802.1X EAP-TTLS with PAP
- Device receives server credentials and validates
- Devices sends its credentials for validation



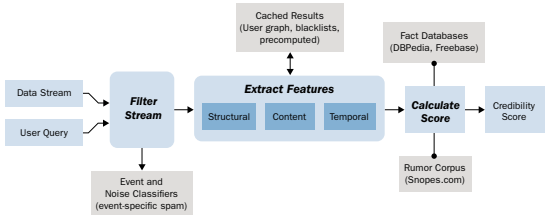
Communication between the mobile device and the cloudlet is encrypted at the transport and message level

Step 4: API Requests
- Device exchanges encrypted messages with the server
- Each exchange is validated against authorized device list

Termination
- Automatic due to timeout: Bootstrapping requires setting up mission length
- Manual due to known loss or compromise: Server Management component has revocation option

Confidence in Information: Assigning Credibility Scores to Social Media Streams in Real-Time

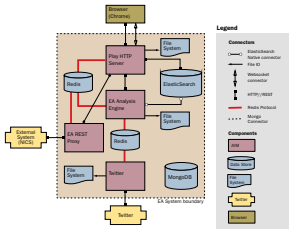
Prototype and algorithm to determine the reliability of information derived from social media.



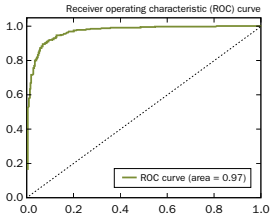
The implementation pipeline for credibility calculation.

Word cloud represents input data from a shooting event

Word cloud represents input data from a non-shooting event



C&C view of Edge Analytics



Receiver operating characteristic (ROC) curve for shooting event detection

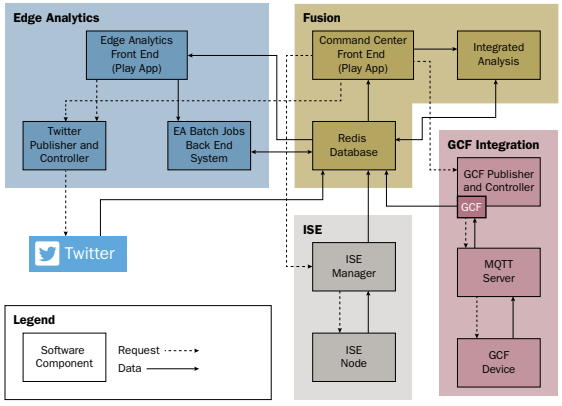
The Edge Analytics prototype

Confidence in Information: Fusion of Social and Physical Sensor Data

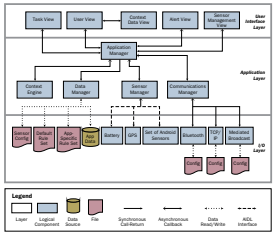
Fusion of local sensor information, gathered cooperatively and opportunistically, with streaming social media and Open Source Intelligence (OSINT) to inform strategic support and improve tactical response.

Possible Relationships
- Tweet + location (actual or inferred) cues GCF sensors
- Trending topic + similar mission keywords cues ISE sensor (events)
- ISE sensor/event + location cues GCF sensors

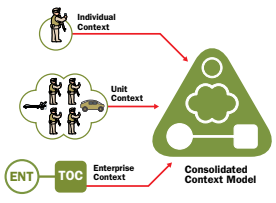
Scenarios (objective)
- Geo-tagged tweet triggers GCF sensors for collection
- Trending keyword matches with ISE event description
- Use ISE to task GCF for additional sensor data
- ISE event triggers GCF sensor collection



Fusion Architecture



ISE implementation



Cooperative Context Sharing

Contact: Grace Lewisand Jeff Boleng glewis@sei.cmu.edu, jlboleng@sei.cmu.edu

Software Engineering Institute | Carnegie Mellon University

SEI Research Review 2015

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

Effecting Large-scale Adaptive Swarms through Intelligent Collaboration

Principal Investigator

Dr. James Edmondson

James Edmondson, PhD, builds middleware for distributed artificial intelligence. He specializes in real-time systems, control, and distributed algorithms.

Background

The U.S. government maintains a fleet of approximately 11,000 unmanned autonomous systems (UAS). To control this large fleet of UAS, DoD and other agency employees depend on mostly joystick-and waypoint-navigation-based systems between one or more human operators to each UAS. While reasonably effective for the control of small, individual UASs, these control paradigms are cumbersome and difficult to manage in a mission-critical scenario (e.g., while under fire) and do not scale to control multiple UAS simultaneously.

We believe the future of autonomy, especially for multiple UAS in a DoD or DHS scenario, is to allow a single operator to specify overarching rules and mission objectives that define the types of activities that the UAS should attempt to perform. Because objectives and rules are likely to change, the UAS swarm should be able to respond dynamically and proactively to environment and mission changes, to better support the user without feedback or guidance. In order to accomplish this, we are focusing our research on a more adaptive and robust distributed operating environment for UAS that scales as required.

Approach

We are building on prior research at the SEI (e.g., the FY2013 SMASH LENS and the FY2014 GAMS initiative[1]). We also know from DoD documents that the UAS fleet is heterogeneous, there is no plan to replace the fleet (only utilize it better), and any operating environment must be portable to these platforms and open-architecture in nature. Though the DoD is interested in control of multiple UAS, the various DoD agencies appear to believe that the technology is too far out to be immediately planned for [Brannen 2014 and Defense Science Board 2012]. However, because of previous SEI research and expertise in scalable-middleware-despite-reduced-connectivity, we have unique resources and connections that can make this capability a reality.

In this project, we developed an open-sourced, scalable, and portable distributed operating environment that enables a single human operator to control and understand a large-scale (e.g., 20+) group of UAS. We added quality-of-service throughout the networking, algorithm, and platform internals of the Group Autonomy for Mobile Systems project to address issues with control, timing, and suppression of emergent behavior.

We then implemented a dynamic, distributed algorithm infrastructure that allows each UAS to morph its mission and objectives according to freshly perceived context or user commands so teams of UAS can respond to new threats or mission parameters. Last, we created robust middleware that provides reflective frame translations between disparate coordinate systems to ease the transition from simulated UAS to real world autonomous boats from Platypus LLC, which were tested outdoors as multi-agent systems in Qatar and Pittsburgh.

Artifacts

Tools developed in this project and its predecessors are available at https://github.com/jredmondson/gams and http://madara.sourceforge.net.

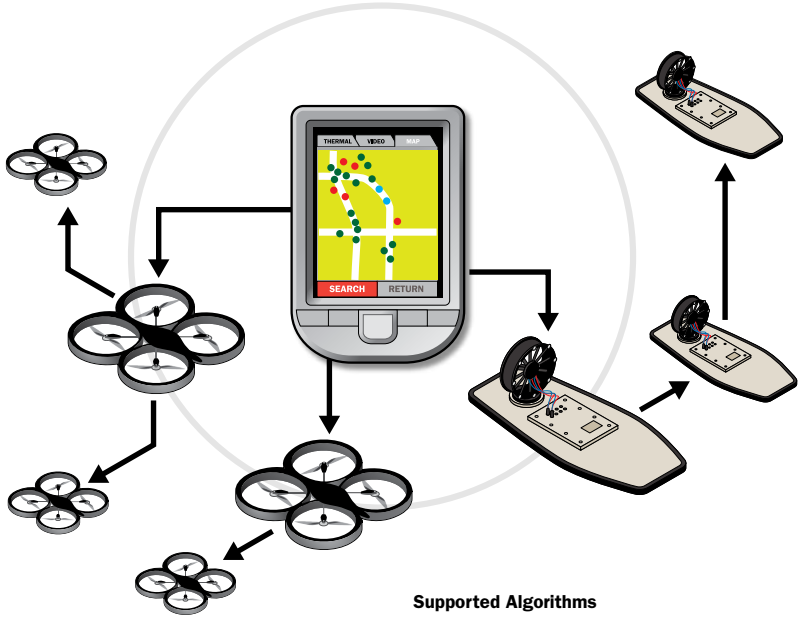
1 SMASH is Self-Governing Mobile Adhocs with Sensors and Handhelds; GAMS is Group Autonomy for Mobile Systems

Effecting Large-scale Adaptive Swarms through Intelligent Collaboration (ELASTIC)

The current control paradigm for unmanned systems is largely centralized and involves multiple pilots and analysts. ELASTIC seeks to invert the paradigm and provide scalable, quality-of-service-enabled distributed control from a central pilot interface. We provide these features through middleware and software tools for extensible distributed algorithms and robotic platforms.

ELASTIC technologies provide key features for UAS developers and operators

- Knowledge and Reasoning
- Asynchronous Networking
- Extensible Distributed Algorithms
- Platform Abstractions
- Threading, Timing, and Control
- Open Sourced, Well-Documented Code
  - Multi-Agent Distributed Adaptive Resource Allocation (MADARA) for the distributed OS layer: http://madara.sourceforge.net/
  - Group Autonomy for Mobile Systems (GAMS) for the algorithms and robotics platforms: http://jredmondson.github.io/gams/



ELASTIC project members made changes to MADARA and GAMS to provide significant enhancements to quality-of-service, timing, control, abstraction and scalability of swarm management

- Controllable threaded algorithm execution
- Auxiliary algorithms
- Containers for knowledge abstraction of algorithm and platform attributes and status
- High Performance Filters for Network Events
- Interoperability bindings between Java and C++ algorithms and platforms
- New formation algorithms
- New platforms for Platypus™ LLC boats, ROS Pioneer 3DX
- Scalable to dozens and hundreds of autonomous agents

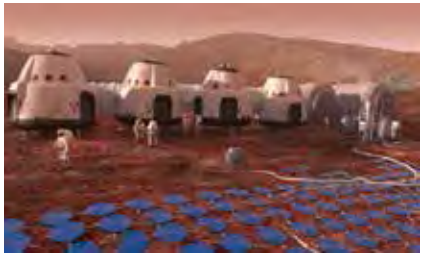
Supported Algorithms

- Formation Coverage
- Prioritized Region Coverage
- Minimum Time Coverage
- Serpentine Coverage
- Waypoints
- Formation Follow
- Synchronized Formations

Supported Platforms

- VREP Boat
- VREP Quadcopter
- VREP Ant Robot
- ROS Pioneer 3DX
- Platypus LLC Lutra Boat
- C++, Java, Android, Python
- ARM, Intel

Keck Institute for Space Studies proposal for Multi-Planetary Smart Tiles features ELASTIC middlewares for autonomous robotic tile collaboration.



Chuck Carter/Keck Institute for Space Studies

ELASTIC results are now used in various projects inside and outside of SEI

- Main code generation target of the DART FY 2015-2016 SEI Line Project
- Autonomy framework proposed for use in the Caltech Keck Institute for Space Studies Multi-Planetary Smart Tile project
- Platypus LLC now uses GAMS and MADARA in their deployed boats for customers in Europe, North America, the Middle East, and South America

20 Node ODROID Cluster for GAMS and MADARA testing for use by SEI DART and MADPARTS FY2016 projects

Platypus LLC Boats deployed in Qatar

Contact: James R. Edmondson jredmondson@sei.cmu.edu

C4I References

[Brannen 2014]
Brannen, Samuel J. *Sustaining the U.S. Lead in Unmanned Systems: Military and Homeland Considerations through 2025*. Center for Strategic and International Studies. February 2014. http://csis.org/files/publication/140227_Brannen_UnmannedSystems_Web.pdf

[Defense Science Board 2012]
Defense Science Board. *Task Force Report: The Role of Autonomy in DoD Systems*. July, 2012. http://www.acq.osd.mil/dsb/reports/AutonomyReport.pdf

URLs are valid as of the publication date of this document.

Human Factors

Generalized Automated Cyber-Readiness Evaluator (ACE)

Principal Investigator

Rotem Guttman

Rotem Guttman researches ways to develop an objective, reliable, valid, and scalable means to evaluate cyber workforce and readiness. His current project is the development of an automated tool that evaluates cyber readiness by tracking actual performance.

Background

The DoD has a need for assessing the capability and capacity of its workforce to support cyber operations. This assessment capability is a key determinant of operational mission readiness.[1] However, the DoD, does not yet have a scalable, objective assessment capability that it can use to validate the hands-on, technical knowledge and skills of its cyber workforce. Workforce improvement is a priority for the DoD as outlined in DoD Directive 8570.1[2] and an integral part of this workforce improvement is the ability to assess and verify the technical capability of its cyber workforce. Two of the six workforce management objectives (C1.3.2 and C1.3.5)[3] defined in the DoD Information Assurance Workforce Improvement manual focus on assessment activities. Additionally, assessment of cyber knowledge and skill (i.e., mission readiness assessment) is a priority for U.S. Cyber Command and is a core component of its Joint Cyberspace Training and Certification Standards.

Approach

To address this problem, we developed the first generation of the Automated Cyber-Readiness Evaluator (ACE)—a system designed to interpret automatically the actions a user performs on a computer screen and objectively measure that user's competence within a defined knowledge and skill set. In this project, we are building a more generalized and extensible platform that can be more easily scaled to evaluate mission readiness in a wide range of computing environments with a variety of DoD cyber personnel.

During the course of this past year, the researchers created a corpus of behaviors in order to train the system on a variety of tools and activities. These activities are mapped to a set of knowledge, skills, and abilities that pertain to a given job role.

As a part of the pilot program, ACE and a set of qualified human evaluators will both make mission readiness determinations about the trainees. We will then use this data to ascertain the validity of ACE's mission readiness determinations by comparing the results from ACE with the normalized results from the human evaluators [Gordon 2012, Falakmasir 2013, and Martin 2011].

Artifacts

- Automated Cyber-Readiness Evaluation Platform
- Peer-reviewed paper on system design
- Peer-reviewed paper on system effectiveness

Future Work

- Extending ACE support for additional job roles (i.e., beyond the job roles from the pilot programs)
- Identifying new, data-backed evaluation criteria for cyber operational mission readiness

1 USCYBERCOM Concept of Operations, "Joint Cyberspace Training and Certification Standards." February 7, 2012, Version 1.2.
2 DoD Directive 8570.1, "Information Assurance Training, Certification, and Workforce Management." August 15, 2004.
3 DoD Manual 8570.01-M, "Information Assurance Workforce Improvement Program." December 19, 2005 (updated January 24, 2012).

Automated Cyber-Readiness Evaluator
ACE

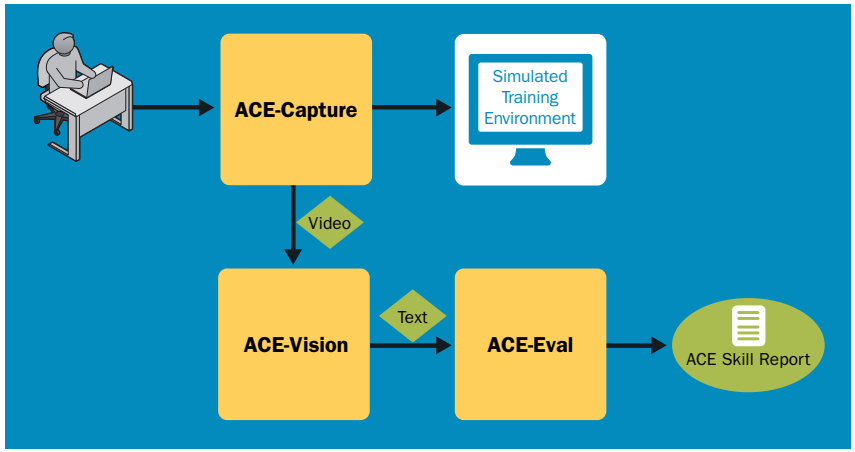
Assessing the mission readiness of all DoD cyber operators is a daunting task that is not achievable using individual one-on-one evaluation techniques. Our project utilizes advanced computer vision and machine learning techniques to evaluate the activity of cyber operators in a realistic scenario in order to determine their mission readiness.

Mission Readiness Assessment

The DoD must assess the capability and capacity of its cyber workforce to support operations conducted in the cyberspace domain and this assessment capability is a key determinant of operational mission readiness. However, because cyber is a relatively new domain for the DoD, it does not yet have a scalable, objective assessment capability that it can use to validate the hands-on, technical knowledge and skills of its cyber workforce.

ACE Philosophy

Current evaluation methods involve checklists of prompted activities or individual assessments. These methods are not reliable, not uniform, and not scalable to DoD requirements. The ACE philosophy is that true mission readiness assessments can only be performed in a realistic environment. ACE users are placed in an environment that mimics their real work environment. Our automated system then observes and understands the actions performed within this environment as users attempt to complete a mission. Based on their activities, our system assesses their knowledge, skills, and abilities.

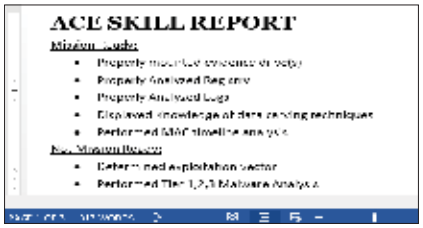


ACE Architecture Overview - User logged into Simulated Training Environment observed using Capture System. Captured video is analyzed and transcribed utilizing ACE-Vision. Vision output is processed by ACE-Eval and used to generate the ACE Skill Report.

ACE-Capture

ACE evaluation scenarios are conducted in the CERT®Simulation, Training, and Exercise Platform (STEP). This platform allows us to push out realistic simulations of real DoD networks through a web browser. The ACE-Capture module has been integrated into the STEP platform, allowing unattended background recording of participants within an evaluation scenario. This recording is performed on the backend servers and consists only of the views we provide to the end users—thus avoiding the possibility of accidentally collecting any personal information that may exists on their personal workstation. Our recording system is highly scalable. It allows us to simultaneously record dozens of users per allocated machine and natively scales with available hardware.

Diagram Title for the Process and Stuff



ACE Skill Reports include an assessment of the capabilities displayed during the evaluation.

ACE-Vision

Video recorded by the ACE-Capture system is processed by a dedicated vision engine that detects a wide array of GUI elements, as well as a set of relevant console commands. These detections (and their associated confidence measures) are generated utilizing a highly optimized, parallelizable algorithm that takes advantage of the unique conditions available within our simulation environment.

ACE-Eval

The detections generated within the ACE-Vision system provide the data for evaluation by ACE-Eval. This system is comprised of two layers. Layer 1 maps groups of detection events with associated higher level activities such as “Opened file examiner_notes.txt for editing in gedit”, “Mounted the evidence drive”, etc. Layer 2 maps these high level activities with the knowledge, skills, and abilities they represent.

ACE Skill Report

The final output of the ACE-Eval system is the ACE Skill Report. This report contains an assessment of the areas in which the participant met or exceeded the requirements for mission-readiness, as well as those areas in which they failed to do so.

By utilizing the automated generation of reliable skill reports, commanders may easily assess the capabilities of their troops, at scale, and with the resources already available.

Contact: Rotem D. Guttman rdguttman@cert.org

Human-Computer Decision Systems for Cybersecurity

Principal Investigator

Brian Lindauer

Brian Lindauer is a research scientist who focuses on applications of machine learning and data mining to cybersecurity problems.

Background

Security decision systems aim to distinguish malicious activity from benign and often use a combination of human expert and automated analysis, including machine learning (ML). Systems using only human experts scale badly; pure ML systems are susceptible to structured attack by adversaries and, in most cases, have unsatisfactory performance on their own. [Jones 2014]

In order for machine learning to reach its potential as a tool for cyber-defense, we must improve collaboration between experts and automation. The well-established subfield of Active Learning studies the ability of an ML algorithm to learn from ongoing human analyst feedback, but it provides only an incomplete view of the system. We must also account for the influence that the ML algorithm has on the human (e.g., through experience gained, boredom induced, or other factors). Furthermore, the Active Learning literature usually assumes that an analyst is a single, flawless, oracle rather than a heterogeneous and fallible team of individuals.

Separate bodies of work acknowledge the mutability and induced biases in human judgment and the profound impact that training data selection can have on ML quality. Nevertheless, Active Learning results have rarely, if ever, been confirmed through human-subject experiments. In order to deploy these systems with confidence in operational computer network defense settings, we must understand the overall system performance and not just the performance of an ML classifier under ideal conditions.

Our research centers on measuring and improving the overall system performance of these human-computer decision systems under realistic assumptions for cybersecurity operations.

Approach

We are working towards the ability to evaluate the human and computer components in the context of overall system success. We have created a model problem (proxy problem, with ground truth, and simplified architecture) with similar statistical characteristics to the real security problem of malware classification solvable by people without security expertise.

This model problem arrangement allows us to conduct large-scale, repeatable, and affordable experiments testing Active Learning advances. The current iteration of our model problem converts openly available malware data into visual figures, which experimental subjects are asked to categorize into “families.” This categorization problem has been implemented within an Amazon Mechanical Turk-compatible experimentation framework.

In parallel with the development of the model problem and experimentation framework, we are working to generalize today’s Active Learning models to loosen what we believe are unrealistic assumptions about the nature of the oracle providing data labels. The mathematical framework we have established allows for dynamic weighting of various Active Learning strategies, allowing us to add human-centered strategies in the future.

Our human-subject experiments will validate (1) that the model problem task is learnable and (2) that the dynamic weighting framework is an improvement over static active learning approaches. In future work, we will be adding dynamic components to model a human analyst and, eventually, multiple analysts working in a team. By employing an adaptive, mixed strategy, we may be able to increase the cost of attacks, because an attacker would have to understand, track, and manipulate a more complex data space.

Artifacts

- Model problem for human-computer decision systems
- Online human-subject experimentation framework
- Dynamic Active Learning mathematical framework and simulations

Future Work

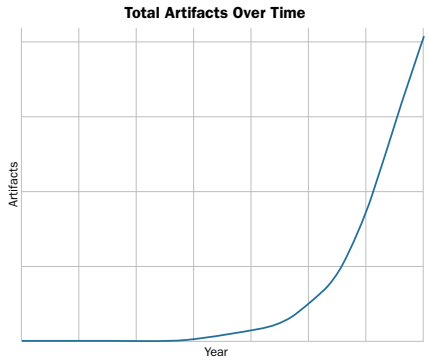
- Validate the dynamic active learning model
- Extend of the system to model the effects on analysts
- Reproduce results using actual security problem and analysts
- Merge results from other SEI research projects addressing the Education and Human Factors technical focus area

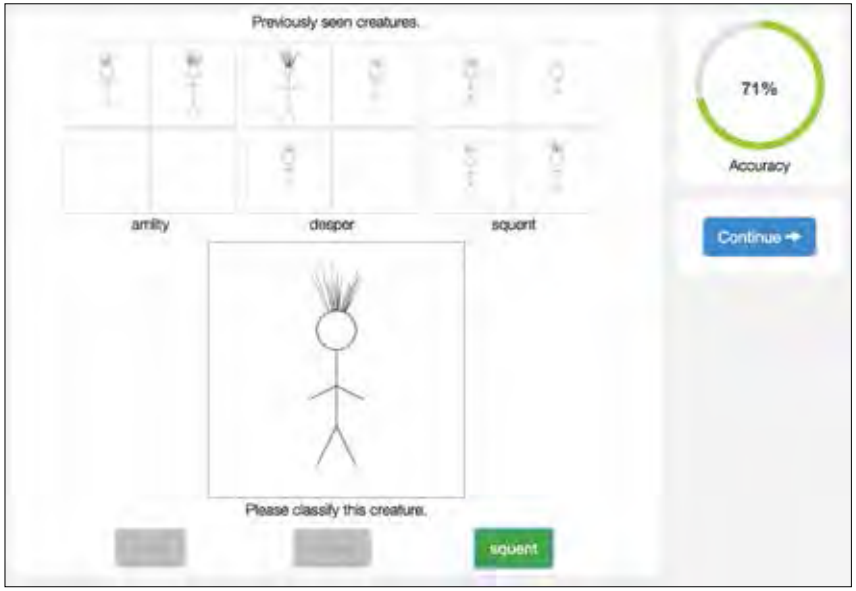
Human-Computer Decision Systems

Security decision systems aim to distinguish malicious activity from benign and often use a combination of human expert and automated analysis, including machine learning (ML). Systems using only human experts scale badly; pure ML systems are susceptible to structured attack by adversaries and, in most cases, have unsatisfactory performance on their own.

- Many operational security problems depend on a small number of skilled analysts to process a large and growing firehose of potentially malicious data.
- Traditional active learning tries to address this situation by suggesting allocation of limited analysis resources that optimize the convergence of a machine learning classifier.

Growth of CERT Artifact Catalog

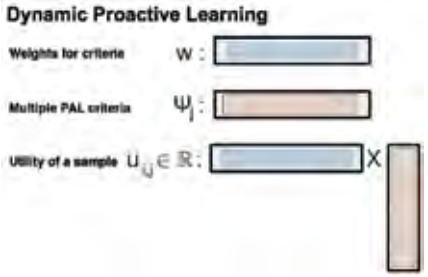




A screenshot of the experimentation system built using Mechanical Turk and Psiturk.

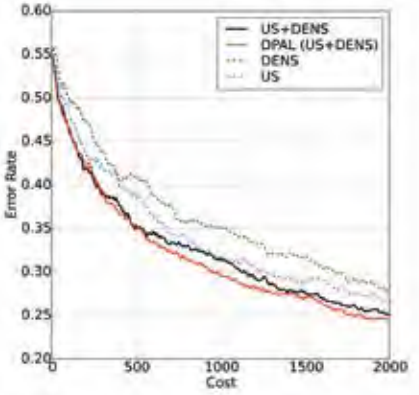
- The human-computer collaboration model will improve upon traditional active learning by optimizing not simply for convergence of the ML component, but also for future performance of the overall system, including mutable human analysts.
- We test the performance of new models not only through simulation, but also through human-subject experiments.
- Because conducting these experiments using real security analysts performing their normal tasks would be prohibitively expensive, we instead developed a proxy problem of identifying fictional creatures and leveraged non-experts on Amazon's Mechanical Turk platform. The process of generating the fictional creatures adheres to the statistical distributions of real malware classes.

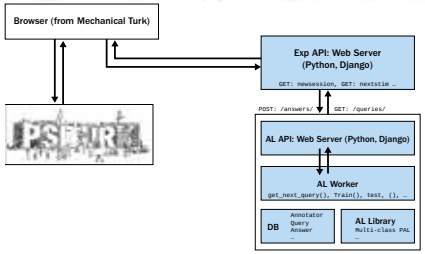
Dynamic Proactive Learning



DPAL provides a framework to combine multiple factors in choosing points, including factors related to analyst performance. It shows promise in simulation and will be put to the test in a human-subject experiment.

Preliminary DPAL Results





Future work includes joint optimization of classifier and analyst objectives, extension of the experimentation software to support multi-session and team experimental trials, and a test of transferability of the model problem results to the target domain.

To keep pace with adaptive adversaries, our cybersecurity defenses must take advantage of both machine learning and human analyst strengths. Future solutions should optimize for success of the overall system.

Contact: Brian Lindauer lindauer@sei.cmu.edu

Insider Threat Mitigation

Principal Investigators

Dr. William Claycomb

William Claycomb, PhD, is Lead Research Scientist for the CERT Enterprise Threat and Vulnerability Management team. His primary research interests focus on insider threats, specifically prediction, detection, and mitigation. He also works across teams exploring cloud computing, incident response, systems modeling, and vulnerability analysis.

Andrew Moore

Andrew Moore currently serves as Lead Researcher of the Insider Threat Center at the CERT Division of SEI. He explores ways to improve the security and resilience of enterprise systems through insider threat analysis, threat/defense modeling and simulation, and system/software assurance engineering.

Background

Despite the high impact of insider attacks, DoD and U.S. Government agencies are challenged to implement effective insider threat programs mandated by Executive Order 13587. The Insider Threat Task Force of the Intelligence and National Security Alliance Cyber Council has found that many organizations have no insider threat program in place, and most of those that do have serious deficiencies. The sociotechnical nature of the insider threat problem, combined with the difficulty of distinguishing malicious from benign acts, make this problem both operationally and technically challenging.

We are addressing the question, "Is there a significant difference between sociotechnical network indications for malicious insider and that for the baseline user population?" Sociotechnical networks include social networks, which provide early indication of insider disaffection, and information flow networks, which can provide indication of suspicious or illicit information flows on and off the organization's computing networks.

Approach

Our research is establishing the scientific validity of the indicators involving insider social networks and developing associated risk measures. We are working with Dr. Kathleen Carley's group at CMU to analyze insider social networks using the Organization Risk Analyzer (ORA). Our working hypothesis is that, over time, the strength of the malicious insider's social network ties (connections) exhibits the following: (1) tie strength with coworkers decreases and (2) tie strength with the external adversary increases. Furthermore, we hypothesize that the change in tie strength for malicious insiders is statistically different from the change in tie strength in the baseline insider population. Some potential ways of measuring tie strength include communication frequency, volume, duration, reciprocity, emotional intensity, and honesty. We are analyzing these measures, validating their ability to distinguish malicious from benign acts, and building models that extend existing theory related to social capital growth and decline based on our findings.

While social network analysis can play an important role in quantifying the risk due to malicious insider threats, information flow network analysis can augment the analysis and potentially reduce the false positive rate of insider threat detection. We are testing the hypothesis that the flow (trajectory) of documents through an organization during purposeful (and unauthorized) data exfiltration is significantly different from established baseline document flows. The baseline document flow will be established through the analysis of a mediated normal data set using document similarity metrics based on, for example, plagiarism detection algorithms, automatic keyword identification, or natural language processing.

Artifacts

Analysis of CERT incidents using CMU's ORA has supported the development and refinement of four working hypotheses regarding insider social networks and a preliminary model of the evolution of these social networks over time. We've developed a Receiver Operating Characteristic (ROC) curve for the analysis of Enron email data that shows the ability to distinguish an Enron insider from non-insiders using a machine learning approach based only on email header data. Our analysis is showing that insider spies distance themselves from both organizational and family social networks, while strengthening relationships among colluder and adversary social networks. Social capital, measurable using standard social network metrics, can serve to indicate dwindling organization commitment and family connections as well as a means to bolster an insider's connections in a way that disincentivizes the threat and improves employee productivity.

Future Work

Existing guidance to insider threat programs does not consider using co-optive measures to disincentivize insider threat by improving employee engagement (e.g., through team building, strength-based management) in addition to traditional coercive measures that constrain employee behavior and punish misbehavior. Coercive measures, especially in isolation, can result in negative unintended consequences that diminish organizational performance, or worse, exacerbate the insider threat. But how can co-optive measures reduce the DoD/Defense Industrial Base insider threat to classified information?

The objective of our future work is to evaluate co-optive measures in the context of a partner's programs and prototype technology that will assist security analysts to know when and how to use the measures to mitigate the threat.[1] The insider's weakening social connections with an employer may indicate a need to take co-optive action, and sociotechnical network analysis techniques developed in our current work may provide the key for needed technology.

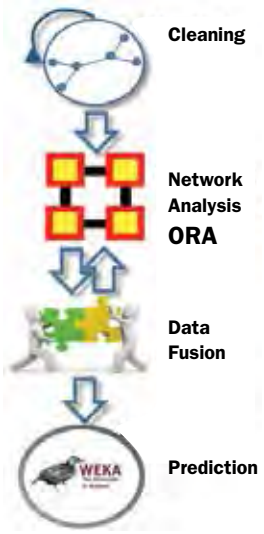
1 This research in this project spans FY2015 and FY2016.

Dynamic Networks of Insider Threats – Growing Holes

Dynamic network metrics can be used to identify those people who are potential insider threats. The key: they grow structural holes and have unusual betweenness.

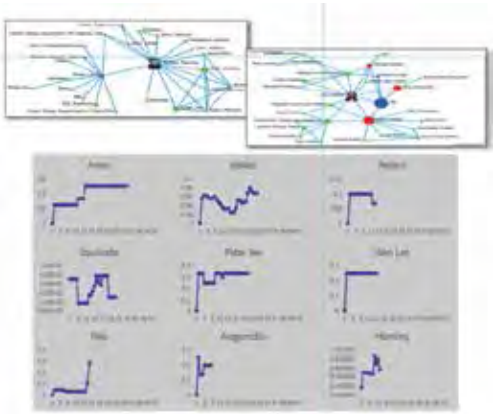
Approach

- Dynamic meta-networks—linking people, personality traits, organizational role traits
- Case Studies: 9 espionage cases
  - Extract dynamic networks
  - Compare networks using graph techniques
- Email Studies: Enron corpus
  - Use machine learning to characterize insiders based on network metrics
  - Use machine learning to characterize insiders using other features derived from case studies
- Identify commonalities across two sub-studies



Espionage Case Studies

Extracted people, traits, and task features
People classified as in organization, external, or family



Key Findings

Characteristics of Insider threats

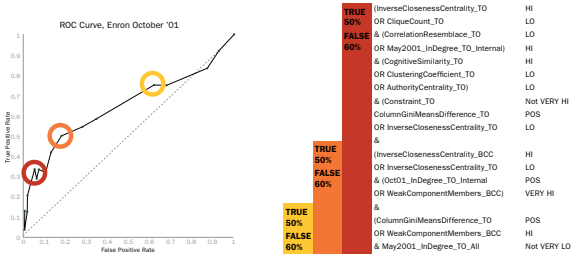
Disengagement with work and family, increasing ties outside

- Network features
  - Build structural holes
  - NOT: degree centrality
  - Member of multiple local clusters
  - Increasing betweenness at "group" level
  - Decreasing ties to family or break with significant other
- Social or Organizational features
  - Access
  - Had or was in military service
  - Minimal supervision
- Psychological features
  - Intelligent
  - Wanted to "use the system" for own gain
  - Wanted change (money/psych change)
- Network features similar to other covert actors
- Complementary patterns in case studies and email
- Not testable
- To be tested

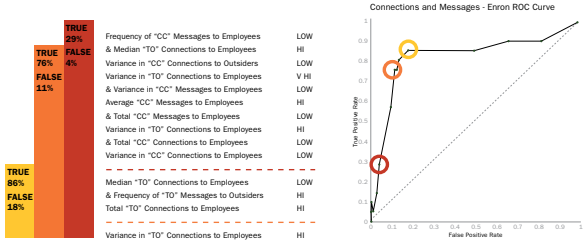
ENRON Email

ORA for Network Metrics, JRIP for ML

Individual Level Network Metrics

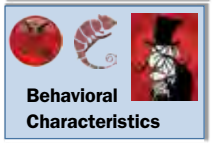


Group Level Network Metrics



Future Work: Add "psychological" features to "network"

Characteristics	Proxy	Source
Aggressive	Repeated Interrogatives ???	Behavior
Aggressive	Repeated Exclamations !!!	Behavior
Aggressive	Sentiment	Behavior
Aggressive	Email Length (Short)	Behavior
Smart	Characters Per Word	Behavior
Smart	Characters Per Sentence	Behavior
Smart	Sentence Length	Behavior
Smart	AVG Reading Ease	Behavior
Chameleon	Variability of Behavior	Network
Chameleon	High Shared Symbols across groups	Network
Compartmentalization	Local Betweenness	Network
	High Variability in Reciprocity	Network
Compartmentalization	High Number of External Connections	Network



Geoffrey Morgan, Neal Altman, Matt Collins
Contact: Andrew Moore apm@cert.org, Kathleen M. Carley kathleen.carley@cs.cmu.edu

Human Factors References

[Falakmasir 2013]
Falakmasir, M. H. et al. A spectral learning approach to knowledge tracing. Pages 28-35. In *Proceedings of the Sixth International Conference on Educational Data Mining (EDM)*. Memphis, TN (USA). July 2013. http://www.bibsonomy.org/bibtex/266779d44de2290d4931139a54dadf9ec/brusilovsky

[Gordon 2012]
Gordon, J. et al. Spectral approaches to learning latent variable models. *Tutorial at Int'l Conf. on Machine Learning (ICML)*, Edinburgh, Scotland. UK, June 26-July 1, 2012. http://techtalks.tv/icml/2012/icml_colt_2012_tutorials/spectral-approaches-to-learning-latent-variable-mo/

[Jones 2014]
Jones, G & Stogoski, John. Alternatives to Signatures (ALTS). White Paper CERT-CC-2014-35. Software Engineering Institute, Carnegie Mellon University. April 2014. http://resources.sei.cmu.edu/asset_files/WhitePaper/2014_019_001_296152.pdf

[Martin 2011]
Martin, B. et al. Evaluating and improving adaptive educational systems with learning curves. *User Model User-Adap Inter.* Volume 21. 2011. Pages 249–283.

URLs are valid as of the publication date of this document.

Please turn to page 59 for information on CyLab at CMU and a sample of its research presented at the SEI Research Review.

Verification & Validation

Parallel Software Model Checking

Principal Investigator

Dr. Sagar Chaki

Sagar Chaki, PhD, researches the theory and applications of formal methods to improving software quality—in particular, specification, verification, and validation of software with particular focus on concurrent software, real-time and Cyber-Physical systems, and software security.

Background

As the DoD continues to become software reliant, rigorous techniques to assure the correct behavior of programs are in great demand. Software model checking (SMC) is a promising candidate, but its scalability remains unsatisfactory.

Recent years have seen the emergence of high performance computing (HPC) technologies (e.g., multi-core processors and clusters). Yet, few software model checkers are designed to use this cheap and abundant computing power. A key reason is that model checking is at its core a graph search—where the graph is the state-space of the model—which is difficult to parallelize effectively (i.e., obtain reasonable speedups).

The main challenge is to partition the search among the CPUs in a way that limits duplicated effort and communication bottlenecks. A promising approach is to start with a verification algorithm that maintains a worklist and to distribute elements of the worklist to different CPUs in a balanced manner. New elements are added to the worklist as a result of processing an existing element. For example, this strategy has been used successfully to parallelize the breadth-first-search in the Simple Promota Interpreter (SPIN) model checker. This project will explore this strategy to parallelize the Generalized Property Driven Reachability (GPDR) algorithm for software model checking.[1]

Approach

Task 1. Develop parallel GPDR algorithm.

We have developed a parallel version of the GPDR algorithm as follows. Overall, the sequential version of GPDR consists of two steps that are performed iteratively till the problem is solved: (i) **lemma generation** in which new facts are learned about the reachable statespace of the system being verified; and (ii) **inductiveness check** in which learned facts are simplified and strengthened.

Our parallel GPDR runs multiple copies of the sequential algorithm that share learned lemmas with each other. The lemmas are shared in batches. In this way, the copies help each other search for a solution to the verification problem, while being loosely coupled (i.e., avoiding a communication bottleneck).

Of course, finding a good communication pattern between the copies is a challenge. Moreover, we have observed that sharing causes the system to become unpredictable. The order in which lemmas are shared between copies changes the search path and influences the future behavior and overall runtime (but not the final result) of the algorithm. However, by using not one but a portfolio of such parallel GPDR solvers and stopping as soon as one finds a solution, we can combat this unpredictability.

We have done a statistical analysis of this unpredictability and used it to compute good portfolio sizes and number of copies to be used in each parallel GPDR solver. This approach has had good results on hardware model checking competition benchmarks. We are currently implementing and evaluating this strategy to target software verification problems.

Task 2. Design scalable architecture for parallel PGPDR.

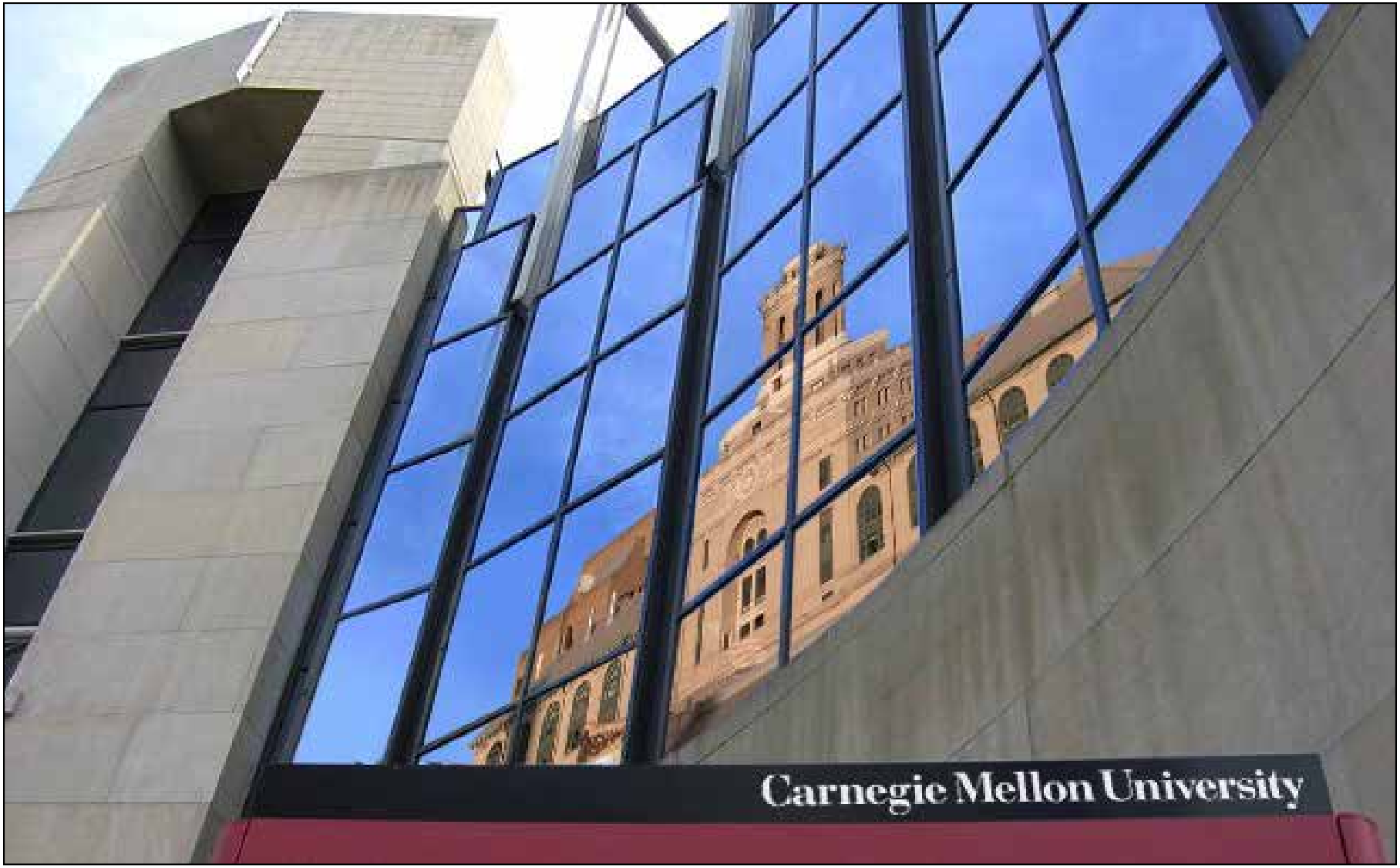
We have implemented two types of parallelization:

- **Single Node.** On a single machine with multiple cores. This uses POSIX threads to implement parallel GPDR solvers. We also implemented tools to construct solver portfolios using shell scripting languages. This was used to run experiments on a machine with 128 cores and 1TB RAM.
- **Clustered.** We also implemented an infrastructure to run large-scale experiments over a cluster of 11 machines each with tens of cores and over 100GB RAM. The infrastructure was implemented using Apache MESOS. For the hardware experiments, each parallel solver runs on a single machine. However, multiple solvers within a portfolio are distributed over different machines in the cluster. For the software experiments, creating a thread-safe version of GPDR was non-trivial. As a result, we are running each sequential copy of GPDR as a separate process and implementing communication between them via the GASNET library.

Artifacts

- Single-node parallel GPDR algorithm
- Clustered parallel GPDR algorithm
- Experimental results
- Peer-reviewed paper

1 GPDR is an algorithm for solving HORN-SMT reachability (HSR) problems [Hoder 2012].



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 is uniquely positioned to bring its organic capabilities in cybersecurity and software across the entire software-related spectrum—from acquisition, to operations, to testing, and to sustainment—to bear on pressing government challenges.

Runtime Assurance for Big Data Systems

Principal Investigator

John Klein

John Klein consults with commercial and government organizations to develop and evolve architectures that satisfy business and mission goals. Through these consulting engagements, John identifies common challenges, and then conducts research to develop practical and repeatable solutions across the entire architecture lifecycle.

Background

The scale of Big Data systems is daunting, comprising many thousands of nodes in widely distributed data centers. This scale creates significant unsolved problems, as evidenced by DoD's $250 million annual investment in Big Data research in the Data-to-Decisions S&T Emphasis Area. Observability, providing deep insights for system design and sustainment from monitoring live systems, is fundamental for Big Data systems. However, the state of the practice is one-off custom observability solutions.

Our work with government agencies and other FFRDCs provides first-hand experience on the limits of ad hoc solutions. It has also contributed to our identifying two pervasive assurance challenges for Big Data systems: (1) efficiently using application-specific requirements to characterize and select storage and analytics components and (2) assuring a deployed system's qualities at runtime as the inputs, workloads, and shared infrastructure evolve.

Approach

The project leverages and extends our FY2014 Data-Intensive Systems work, adding formal metamodels, tooling, code generation and visualization to our Big Data technology evaluation process (LEAP4BD) and architecture design approach (QuABaseBD).[1]

To meet the observability challenges at "Big Data scale," we will create a model-driven toolkit that enables designers to generate system-specific observability monitors and visualizations for Big Data systems from simple graphical models. Specifically, the toolkit

- automates the generation and execution of database evaluation suites, reducing the design effort to assure the scalability of database components for a specific system from months to days
- generates frameworks to insert runtime monitoring into heterogeneous database systems and visualize the results, enabling system sustainment and operation to be predictive rather than diagnostic
- promotes architecturally driven development processes
- is extensible by third parties to increase utility and impact
- is released as an open source platform

Our toolkit is based upon a formally specified set of architecture archetypes for Big Data systems. These archetypes compose primitive architecture styles [Shaw 1996] to define reusable architectures for a wide range of Big Data systems. We use an architecture description language to specify the properties of Big Data architecture archetypes and translate these into meta-models to form the basis of our toolkit.

Then, we create the database evaluation and runtime observability toolkits by extending these core metamodels with elements for creating database evaluation suites (e.g., test data and workload generators) and runtime monitoring (e.g., latency and request mix).

The resulting metamodels will precisely represent the semantics and syntax [Schmidt 2006] of scalable databases and Big Data architectures.

Artifacts

- A model-driven toolkit that generates database evaluation suites and runtime observability frameworks for Big Data systems from simple user-specified graphical models
- Formally specified architecture archetypes for Big Data systems, providing template designs with verified properties for the DoD and software industry to exploit
- A rich, open source visualization toolkit for Big Data systems, with documented APIs for integration with existing observability frameworks

Future Work

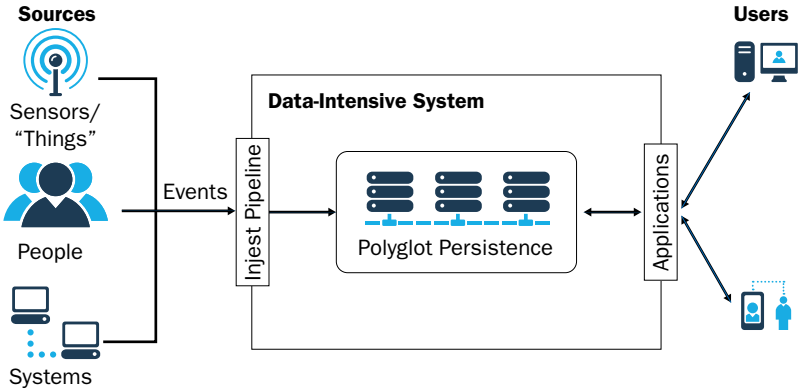
- Incorporate additional archetypes and advanced features into the frameworks for aggregating component-level observations to assess system health
- Incorporate observability capabilities for Big Data analysis engines such as Hadoop and Storm
- Extend the visualization framework by leveraging novel technologies from the Pacific Northwest National Laboratory Visual Analytics group.[2]

1 For more information, view the SEI Webinar "Software Architecture for Big Data Systems" that is available at http://www.sei.cmu.edu/webinars/view_webinar.cfm?webinarid=298346&gaWebinar=SoftwareArchitectureforBigDataSystems.

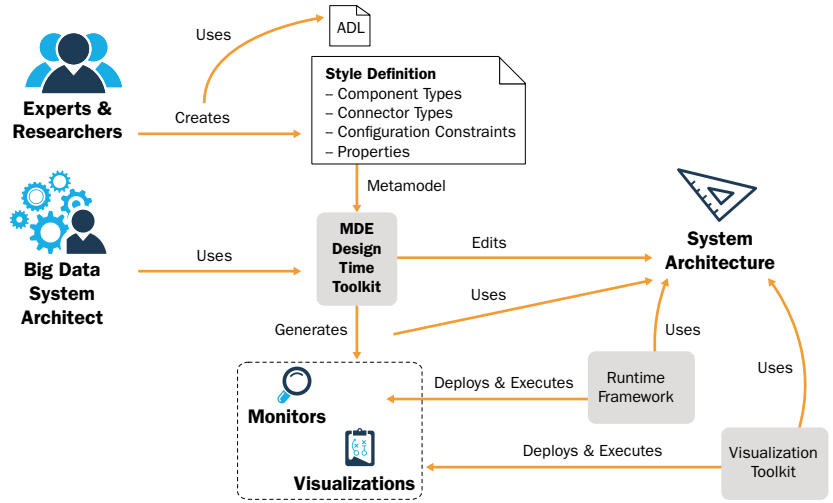
2 http://vis.pnnl.gov/

Runtime Assurance for Big Data Systems

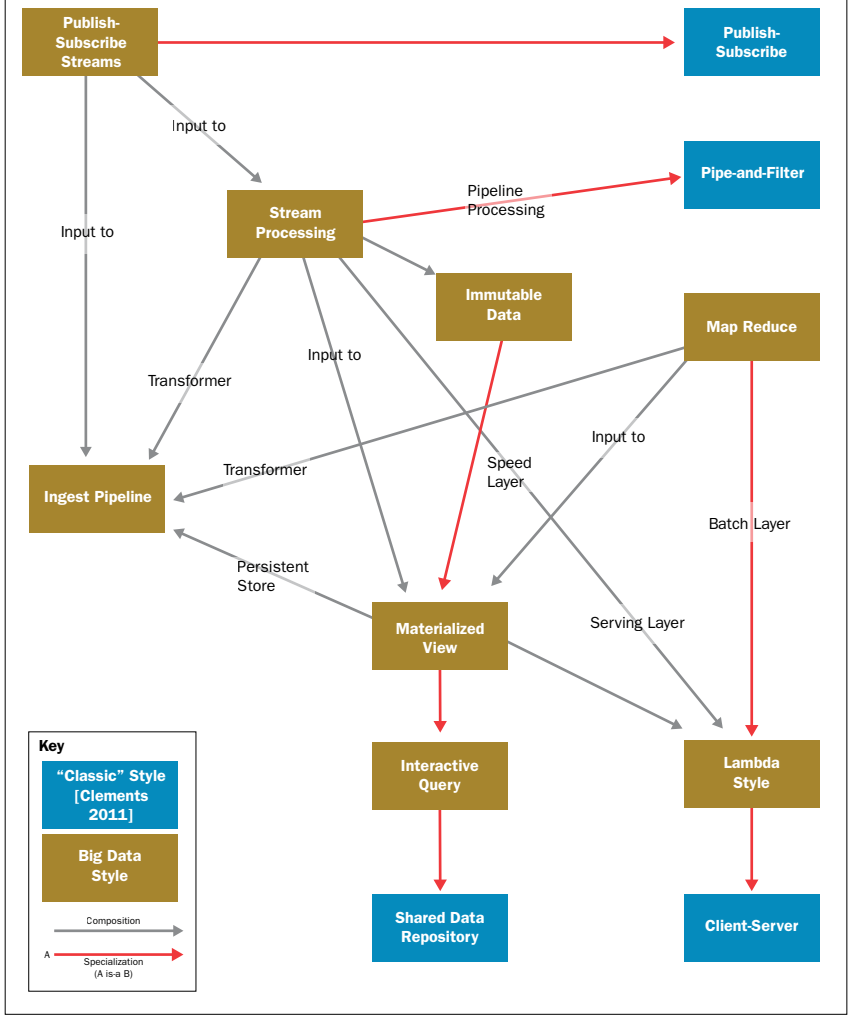
Big Data System Context



Using Architecture Styles to Generate Monitors and Collection



Big Data Architecture Styles



Contact: John Klein jklein@sei.cmu.edu

Incremental Lifecycle Assurance of Critical Systems

Principal Investigator

Dr. Peter Feiler

Peter Feiler, PhD, is the technical lead and author of the SAE AS-2C Architecture Analysis & Design Language (AADL) standard. Version 2.1 of the standard was published in January 2012. He is currently leading the revision of the Error Model Annex standard for AADL. His research work focuses on safety-critical real-time systems, architecture languages, software-reliant systems, and predictable system analysis & engineering

Background

The current practice of build-then-test for software-reliant (safety and mission) critical systems results in rapidly increasing certification-related rework costs with 70 percent of issues introduced during requirements and architecture design, while 80 percent of embedded software system issues are discovered post unit test. The resulting backlog of software changes to remove defects, due to high re-certification cost, presents an increasing safety risk.

The focus of certification is to assure the quality of a delivered system by presenting sufficient evidence that the system implementation meets system requirements. In other words, both the quality of requirements and the quality of evidence determine the confidence in the quality of the system. The high cost of system certification—currently certification related software rework cost is at 50 percent of the total system cost and growing—presents a challenge of a desired recertification cost that is in proportion with the impact of change.

In this project, we are addressing these assurance challenges by developing an Eclipse-based open source Architecture-Led Incremental System Assurance (ALISA) workbench that implements a technical strategy outlined in the Reliability Validation & Improvement Framework study for the U.S. Army AMRDEC.[1] This strategy combines architecture-centric virtual system integration with assurance case concepts to demonstrate the feasibility of greatly reducing certification costs and measurably improving confidence in the quality of software-reliant critical systems.

Approach

In this project, we improve system certification through incremental system assurance throughout the lifecycle by

- improving the quality of requirements through measurable improvement of requirement coverage by an architecture-led requirement specification approach that incorporates operational quality attribute and safety/security hazard ontologies
- assuring hazard and derived requirement coverage during architecture design iterations by an architecture-led hazard analysis and mitigation approach
- compositionally verifying models of the evolving architecture design using virtual system integration principles centered around the SAE International Architecture Analysis & Design Language (AADL) standard suite AS5506
- providing a continuous measure of confidence throughout the lifecycle by tracking requirement coverage in verification plans and tracking verification/test results as evidence in support of or as counter examples to meeting requirements

In our first year, we have focused on developing a notation for capturing system requirements, verification plans, and a multi-valued argumentation logic by unifying concepts from different existing frameworks and prototyping its implementation in the ALISA workbench. We have also developed a capability that enables users to configure assurance plans for specific systems and assurance tasks. Instances of such plans are then executed and verification results are tracked. Requirement and verification result metrics are collected automatically to provide insight into potential problem areas in assuring the system.

Artifacts

- Prototype release of the ALISA workbench
- Demonstration of ALISA on a public large-scale example

Future Work

In the second year of this project we will investigate effective management of uncertainty in requirements and architecture design through assurance support focused on key quality attributes of interest. We will also investigate incremental change impact analysis across requirements, architecture and detailed design, source code, and verification activities.

In addition, we will demonstrate the effectiveness of compositional and incremental verification to achieve re-certification cost. Finally, we will draw on the experience with the ALISA notation to influence ongoing standardization activities at the SAE International AS-2C committee on a requirement specification language and a constraint verification language as part of the AADL standard suite as well as the OMG Structured Assurance Case Model (SACM) standard.

1 From the SEI study *CMU/SEI-2012-SR-013 prepared for* Dr. Lewis, Director of AMRDEC Aviation Engineering Directorate, (now S&T)

Incremental Lifecycle Assurance of Critical Systems

Critical System Assurance Challenge

The traditional development lifecycle using existing methods of system engineering result in

- Assurance-related post-unit test software rework at 50% of total system cost and growing
- Labor-intensive system safety analysis without addressing software as major hazard source
- High percentage of operator work arounds for software fixes due to high recertification cost

NIST Study

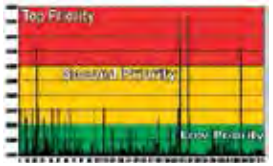
Current requirement engineering practice relies on stakeholders traceability and document reviews resulting in high rate of requirement change

Requirements error	%
Incomplete	21%
Missing	33%
Incorrect	24%
Ambiguous	6%
Inconsistent	5%

Rolls Royce Study

Managed awareness of requirement uncertainty can lead to 50% reduction in requirement changes

Selection	Weight	Precedence
Low Precedence	9	No experience of concept, or environment. Historically volatile.
Medium Precedence	3	Some experience in related environments. Some historic volatility.
High Precedence	1	Concept already in service. Low historic volatility.



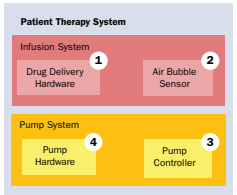
U Minnesota Study

Requirements often span multiple architecture layers

Textual Requirements for a Patient Therapy System

1. The patient shall never be infused with a single air bubble more than 5ml volume.
2. When a single air bubble more than 5ml volume is detected, the system shall stop infusion within 0.2 seconds.
3. When piston stop is received, the system shall stop piston movement within 0.01 seconds.
4. The system shall always stop the piston at the bottom or top of the chamber.

Same Requirements Mapped to an Architecture Model



Importance of understanding system boundary

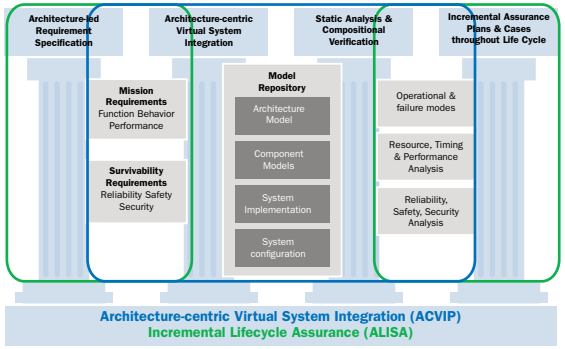
We have effectively specified a system partial architecture

Incremental Lifecycle Assurance Goals

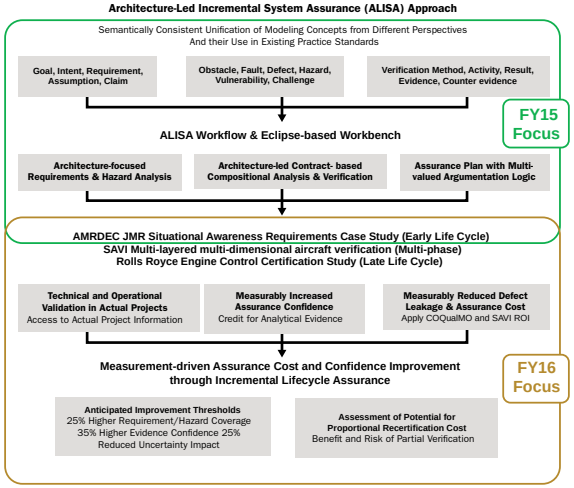
- Improve requirement quality through coverage and managed uncertainty
- Improve evidence quality through compositional analytical verification
- Measurably reduce certification related rework cost through virtual integration and verification automation

Assurance & Qualification Improvement Strategy

Assurance: Sufficient evidence that a system implementation meets system requirements

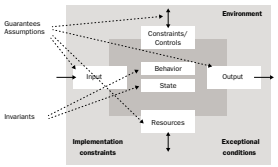


Project Approach

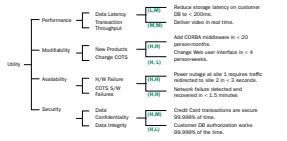


Three Dimensions of Requirement Coverage

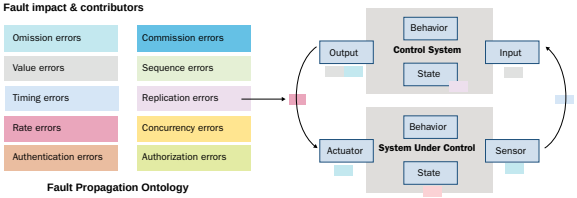
System interactions, state, behavior



Design & operational quality attributes



Fault impact & contributors

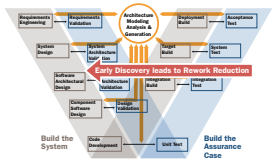


Fault Propagation Ontology

Three Dimensions of Incremental Assurance

Incremental assurance through virtual system integration for early discovery

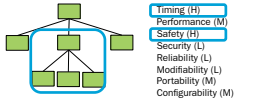
Return on Investment study by SAVI*



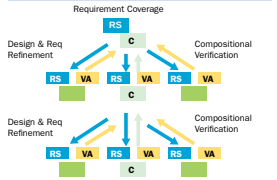
*System Architecture Virtual Integration (SAVI) Aerospace industry initiative

Priority focused architecture design exploration for high payoff

Measurable improvement (Rolls Royce)



Compositional verification and partitions to limit assurance impact



Impact and Alignment

- AMRDEC Joint Multi-Role (JMR) Tech Demo: maturation of ACVIP for Future Vertical Lift (FVL)
- Aerospace industry System Architecture Virtual Integration (SAVI) multi-year initiative
- Standards: SAE AS-2C (AADL Requirements, Constraints), SAE S18 (ARP4761 System Safety)
- Regulatory agencies: NRC, FDA, AAMI/UL

Contact: Peter Feiler phf@sei.cmu.edu

Verifying DART Systems

Principal Investigators

Dr. Sagar Chaki

Sagar Chaki, PhD, researches the theory and applications of formal methods to improving software quality—in particular, specification, verification, and validation of software with particular focus on concurrent software, real-time and Cyber-Physical systems, and software security.

Dr. Dionisio de Niz

Dionisio de Niz, PhD, investigates timing verification and resource allocation (real-time scheduling) of Cyber-Physical systems to address application needs and new processor technologies. His research focus is on the functional consolidation of features of different criticality into the same platform, known as mixed-criticality systems.

Background

Distributed Adaptive Real-Time (DART) systems are key to DoD capability. We are developing assurance techniques for DART systems via (1) new temporal isolation mechanisms to protect high-critical threads from low-critical ones on processors with shared hardware resources, (2) new compositional model checking algorithms to verify high-critical properties of distributed software, and (3) new proactive self-adaptation approaches to achieve low-critical properties under uncertainty and assure them using statistical model checking. We will validate these techniques synergistically on a DoD-relevant DART system.

This work continues our long-term research into mixed-criticality and real-time scheduling, model checking, and High Confidence Cyber-Physical systems (HCCPS). This project extends our HCCPS results by adding a focus on verifying distributed and self-adaptive systems and by explicitly engaging with DoD stakeholders to achieve operational relevance. It also aligns with our ongoing and shorter-term research in model checking distributed software, probabilistic analysis of time-sensitive systems (a project funded by the Air Force Office of Scientific Research) using statistical model checking, and latency-aware self-adaptation.

Approach

In general, formally verifying a DART system is intractable. To overcome this challenge, we will use two strategies:

- a specific architecture to provably isolate the high-critical parts of the system from the low-critical ones
- a set of automated analyses that leverage isolation and compositionality to perform scalable verification while allowing proactive self-adaptation

Task 1: We have developed a new mixed-criticality scheduling algorithm for end-to-end distributed tasks running in a pipeline structure. We also designed an efficient enforcement mechanism to protect high-criticality tasks that reduces the number of timers used and tasks that need to be stopped. This approach will reduce the scheduling overhead and network messages necessary for the end-to-end protection of critical tasks. In addition, it can increase the CPU cycles available to be used by other applications

Task 2: We have developed a new domain specific language, called DART Modeling and Programming Language (DMPL), for programming DART software. DMPL automatically enforces the DART runtime architecture and supports both synchronous and asynchronous threads with multiple levels of priority and criticality. We have developed a compiler for generating C++ code from DMPL that uses a middleware for communication, a scheduler for meeting real-time thread deadlines, and a simulator for implementing the physical aspects of a DART system. The compiler also verifies correctness of sequential threads via software model checking. We used it to verify correctness of a collision avoidance protocol with manually supplied invariants.

To verify asynchronous threads, we developed assume-guarantee proof rules for concurrent programs assuming communication over a zero-delay network. This model extends the synchronous thread semantics and is a step toward fully asynchronous behavior. We manually created an abstract model of the synchronous collision avoidance algorithm for this new semantics. The collision avoidance property was then proven automatically for this model by constructing inductive invariants using our verification engine Spacer.

Task 3: We developed an approach for proactive latency-aware adaptation that takes into account the uncertainty of the environment. The approach uses formal specification and the Alloy analyzer to compute offline a relation with the feasible transitions of a Markov decision process (MDP) taking into account the latency and effect of adaptation tactics, the evolution of system state, and conflicts between tactics. This relation is used at runtime to solve the joint MDP of the system and the environment using stochastic dynamic programming. The approach can make adaptation decisions 10 times faster than our previous approach under PRISM.

Task 4: We are developing a DoD-relevant prototype DART (e.g., a multi-UAS) system and will verify it using the techniques developed in Tasks 1-3.

Artifacts

- Mixed criticality scheduling tool, scheduler for Linux, RT Mutex for Linux, experimental results
- DART compositional model checking tool, experimental results
- Proactive self-adaptation implementation, experimental results
- Experimental results on DoD-relevant DART system
- Submissions to peer-reviewed venues

Future Work

Some possibilities are considering faults and fault-tolerance, developing additional assume-guarantee rule, distributed self-adaptation, and transition.

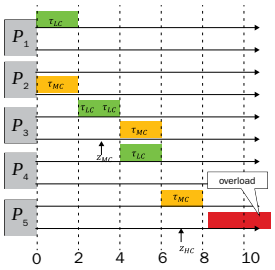
Verifying Distributed Adaptive Real-Time (DART) Systems

Pipelined ZSRM Scheduling

- Reduces pipeline to single-resource scheduling
- Avoids assuming worst alignment in all stages

But need to deal with transitive interferences due to zero-slack

Ongoing work: theory worked out, implementing scheduler in Linux



Functional Verification

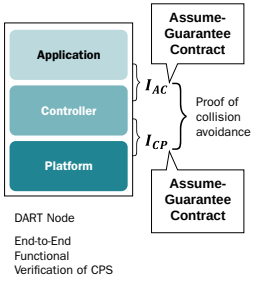
Prove application-controller contract for unbounded time

- Previously limited to bounded verification only

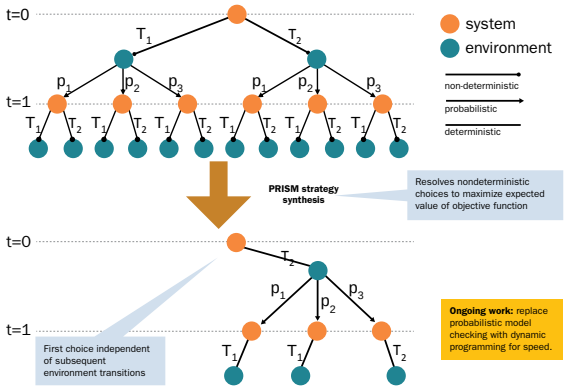
Prove controller-platform contract via hybrid reachability analysis

- Done by AFRL

Working on automation and asynchronous model of computation



Proactive Self-Adaptation Using Probabilistic Model Checking

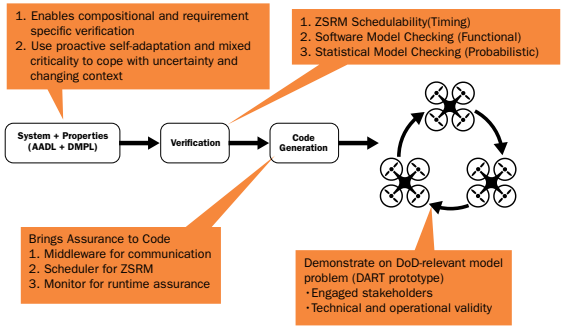


DART Vision

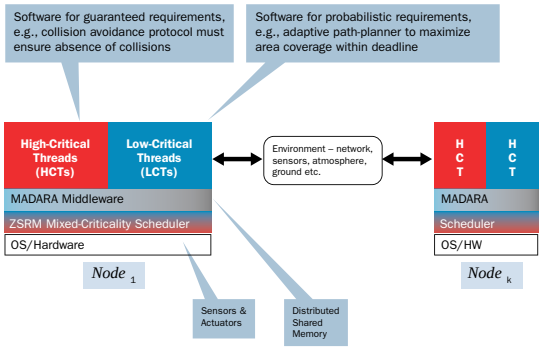
A sound engineering approach based on the judicious use of precise semantics, formal analysis and design constraints leads to assured behavior of (DART) systems while accounting for

- critical requirements
- probabilistic requirements
- uncertain environments
- necessary coordination
- assurance at source code level

DART Process



DART Architecture

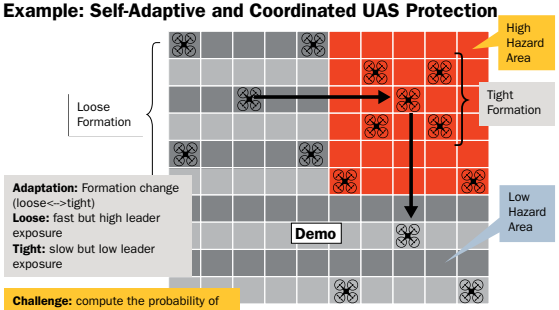


DMPL: DART Modeling and Programming Language

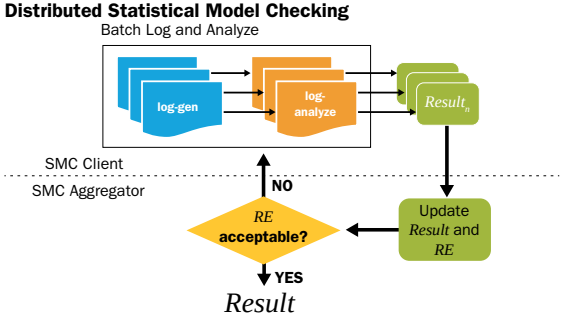
- C-like language that can express distributed, real-time systems
- Semantics are precise
- Supports formal assertions usable for model checking and probabilistic model checking
- Physical and logical concurrency can be expressed in sufficient detail to perform timing analysis
- Can call external libraries
- Generates compilable C++
- Developed syntax, semantics, and compiler (dmplc)

DMPL supports the right level of abstraction. github.com/cps-sei/dart

Example: Self-Adaptive and Coordinated UAS Protection



Distributed Statistical Model Checking



Contact: Sagar Chaki chaki@sei.cmu.edu

Verification & Validation References

[Clements 2011]
Clements, Paul & Bachmann, Felix. *Documenting Software Architectures: Views and Beyond (2nd Edition).* Addison-Wesley. 2011. ISBN 978-0321552686

[Hoder 2012]
Hoder, Krystof & Bjørner, Nikolaj. Generalized Property Directed Reachability. Pages 157-171. In Proceedings of the *15th International Conference on Theory and Applications of Satisfiability Testing (SAT).* Trento, Italy. June 2012. http://research.microsoft.com/en-us/people/nbjorner/z3pdr.pdf

[Schmidt 2006]
Schmidt, D. C. Model-Driven Engineering. *IEEE Computer.* Volume 39. Number 2. February 2006. Pages 25-31. DOI: 10.1109/MC.2006.58

[Shaw 1996]
Shaw, M. & Garlan, D. *Software Architecture: Perspectives on an Emerging Discipline.* Prentice Hall. 1996.

URLs are valid as of the publication date of this document.

Cybersecurity

Design Pattern Recovery from Malware Binaries

Principal Investigator

Cory Cohen

Cory Cohen researches ways to improve manual and automated analysis of malware. His primary focus is on static analysis techniques using the SEI's Pharos framework, which is built on the ROSE compiler infrastructure. His work aims to reduce the analyst time and network defense cost required to analyze malware.

Background

The DoD continues to face a wide variety of ongoing problems with malware. CERT provides operational malware analysis capabilities, including those focused on malware family evolution and similarity. Our sponsors look to us for innovative new advances in these fields.

Previous malware similarity work has focused on low-level syntactic features such as op-codes or semantic concepts such as code-level functional equivalence. In this work, we focus on the design patterns used by malware authors. Malware authors are now developing reusable software components to help address common software engineering problems [Havrilla 2012]. CERT analysts have noted malware families with similar designs but obviously different implementations [CERT/CC 2013]. We expanded our existing automated malware analysis infrastructure built in ROSE to find such similar abstractions using ideas inspired by existing research on design pattern recovery from source code.

Our goal for this project is to develop an automated tool to provide human analysts with the raw data required to make design-level similarity decisions. Such a tool will dramatically reduce the time required to gather data for comparisons. Specifically, we propose to extend our existing automated analysis capability built in the ROSE open source compiler framework to provide the required data [Lee 2011, ElWazeer 2013].

Approach

We are developing a type recovery system to automatically recover the prototype declarations of malware functions and a design pattern matching system to look for patterns in malware. We are collecting information about the types of each parameter used in various operating system calls by harvesting this information from source code headers. The type recovery framework will be used to propagate those types to the user-defined malicious function prototypes via data flow analysis.

Our existing instruction emulation framework provides globally unique symbolic variables that we can associate with type constraints. Upper and lower bounds on types will be derived from instruction semantics where strong types are not available from API parameters. Custom object types can be obtained from our previous work on C++ class recovery [Jin 2014]. Automatically recovered function prototypes containing detailed type information are a rich source of information about the overall design and architecture of the program and will be valuable for a variety of manual reverse engineering tasks that support existing work with SEI users.

To evaluate our type recovery system, we will compare the results against answers obtained from debugging information for a set of non-malicious programs, leveraging validation techniques developed during our C++ class recovery research. This method should allow us to generate precision and conservativeness statistics for the accuracy of the recovered prototypes.

While design pattern recovery can be tested against well-known patterns, it is unclear to what extent these patterns are relevant to malware analysis. If determined to be worthwhile, our design pattern-matching infrastructure can be evaluated against binary libraries with known patterns [Fontana 2012], common libraries like Boost,[1] or the Microsoft standard template library implementation.

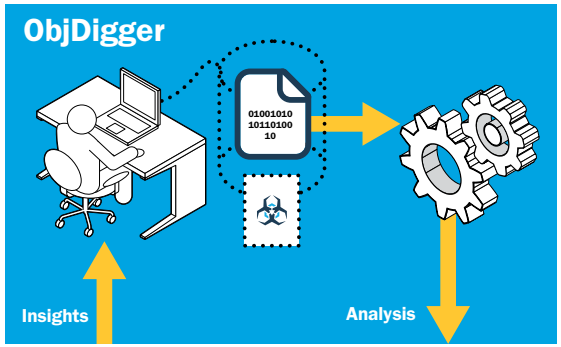
For a more malware-centric evaluation, we will use families of malware with manually determined similarity to evaluate whether there is substantial similarity in the function prototypes across variations in the family. Ultimately, we will seek analyst feedback on whether the recovered prototypes and design pattern matching capabilities are as useful as currently anticipated.

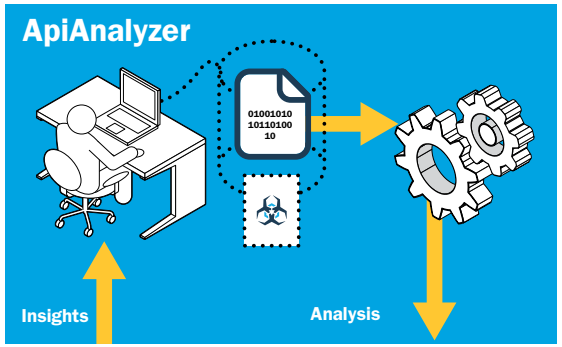
Artifacts

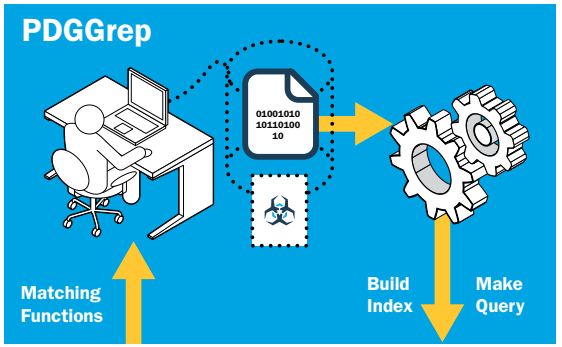
- A type recovery system extending or current analysis capability
- A design pattern matching algorithm and prototype
- A peer-reviewed academic paper on design pattern similarity in malware

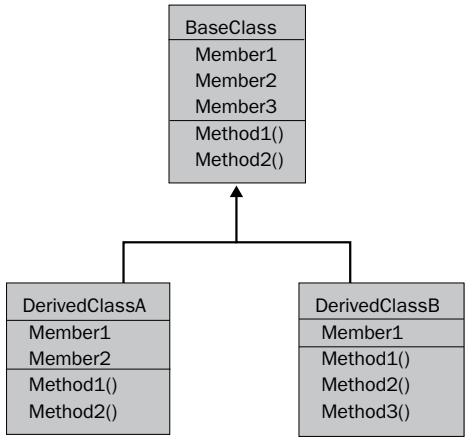
1 Boost is a set of libraries for the C++ language. The libraries support linear algebra, image processing, and similar tasks.

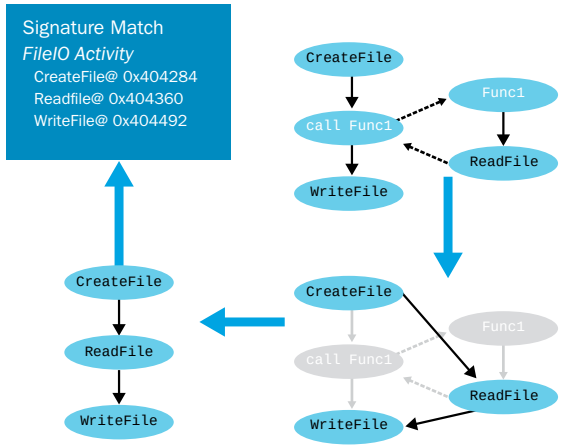
Pharos Static Analysis Tools

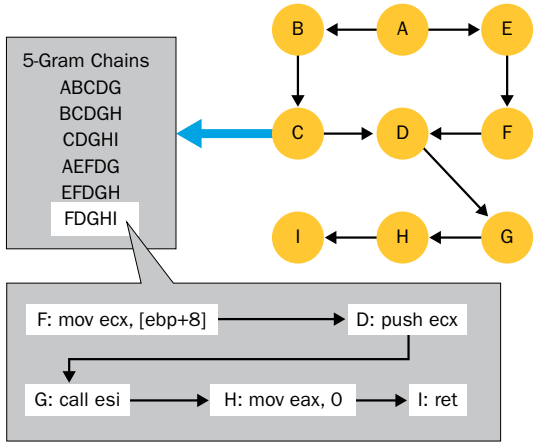












Analyzes OO (Object-Oriented) program and finds:

- **Classes with members**
- **Methods for each class**
- **Virtual function tables**
- **Member accesses**
- **Virtual function calls**

Analyzes API call graph:

- **Reads EXE and signatures**
- **Simplifies CFG (just API calls)**
- **Searches CFG recursively**
- **Substitutes subgraphs in CFG**
- **Returns signature matches**

ROSE use-def makes chains:

- **Index is built from chains**
- **Query is built the same way**
- **Insensitive to register changes**
- **Match or ignore constants**
- **More semantic than BigGrep**

Contact: Cory Cohen cfc@sei.cmu.edu

API Usability and Security

Principal Investigator

Dr. Samuel Weber

Samuel Weber, PhD, gathers empirical evidence about security engineering. His research aims to develop and test actionable principles that will result in improved secure development practices.

Background

How can APIs be designed so that programmers will be less likely to write insecure code? In this project, we aim to gather empirical evidence about the security impacts of API design. Ultimately, the cause of many cybersecurity failures is flawed code written by programmers. Our philosophy is that programmers are people, and we need to study how to design usable APIs—that is, APIs with which it is easy to develop secure code. Our objective is to improve the design of APIs to be more usable and less prone to programmer mistakes that result in vulnerabilities.

It is well known that API design can affect security. For example, buffer overflows were documented as early as 1972, but they remain one of the most common vulnerabilities. Likewise, the gets() C Standard library function has been the cause of innumerable security vulnerabilities. Furthermore, APIs are designed by a small number of experienced developers, but they have an extremely long life-span. Therefore, good API design has a large multiplicative effect. Experts and researchers are increasingly urging API designers to consider usability.

We are starting our project by investigating an aspect of design known to have security implications: state management and immutability. This project will determine the security and usability issues involving system state, so that reasoned decisions can be made by architects and language and system designs can better assist developers.

Approach

Our research methodology relied on a mix of literature review, controlled interviews, prototype experimental systems and laboratory studies with human subjects under more controlled conditions, [Ellis 2007, Nielsen 1993, Styles 2007, Yoon 2001, and Chin 1988].

To determine how programmers managed state and used concepts like immutability, and what problems and issues they experienced, we conducted a series of semi-structured interviews with professional programmers. Our subjects all had more than 7 and an average of 15 years of experience, and participants included those with DoD-relevant backgrounds. Because previous research shows that users often do not know what features will benefit them most, our interviews were designed to elicit their experiences with large systems: how they designed such systems, what features they used, and what problems they experienced.

From these interviews we determined that state management was a frequent and severe problem experienced by developers and that the language features available to them were inadequate. We developed a set of design requirements for features that would address these issues. Prototypes of two alternate Java language extensions were built that would address the issues experienced by professionals. Controlled user-studies of these extensions are currently in-progress in order to validate our work.

Artifacts

- Academic paper comparing transitive and intransitive object immutability
- Academic paper on language support for immutability
- Two Java language extensions implemented

Future Work

- User-studies on Java language extensions (currently in progress)
- Java language extension and supporting user tooling to explore additional state-management support (currently in progress)
- Papers summarizing results of above studies

API Usability and Security

Our goal is to develop and empirically test *concrete* and *actionable* API design principles that lead to more secure code.

APIs are the boundaries between system components, defining how they interact. Programmers failing to commonly understand how an API should be used causes failures.

Project Principles:

- Secure development practices should be empirically tested and validated.
- Programmers and designers are people too.

Why APIs?

- Large impact on system security
- Long-lasting
- Designed by a small number of more-experienced people

Initial Focus:

- State management: how state of objects defined by API can be changed

Methodology

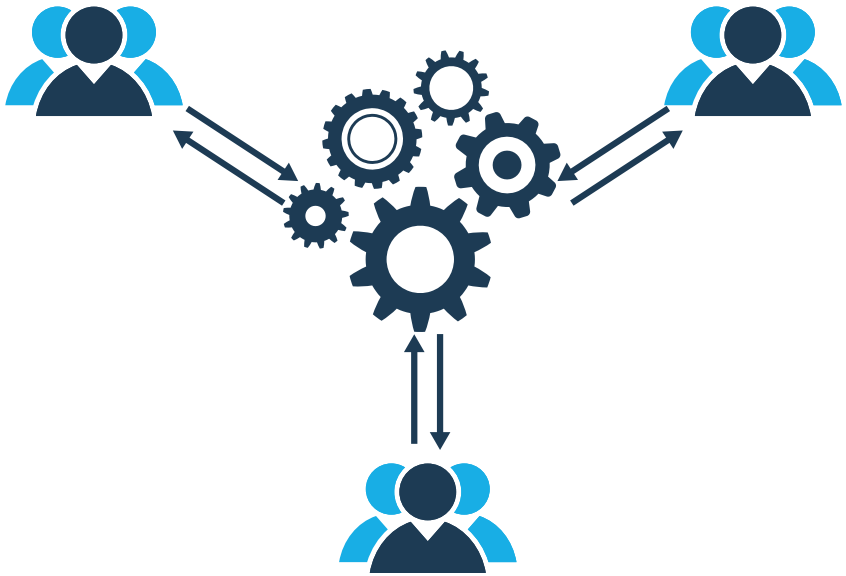
1. Semi-structured Interviews with developers
2. Prototypes addressing issues
3. Evaluation with user studies

Co-funded by NSF award 1423054

SEI participants:
Forrest Shull, Robert Seacord, David Keaton

CMU participants:
Dr. Brad Myers, Dr. Jonathan Aldrich, Dr Joshua Sunshine, Michael Coblenz

Summer students:
Sophie Gairo, Paul Peng



Programmers use APIs to get things done, but must understand each other's roles and responsibilities.

Semi-structured Interviews

Interviewed experienced programmers: at least seven and a mean of 15 years of experience, most with DoD-relevant projects

Results include:

- Controlling where/when state is changed and by whom is a serious problem
- Programmers do use concepts like immutability (data structures that cannot be changed after being created)
- Language features, like `const`, don't satisfy programmer's needs.

Issues:

- Object vs class immutability
- Abstract vs concrete state
- Viral nature of C++ `const, ...`

Question: What fraction of bugs are the result of [unexpected] state changing?

Answer:

"Oh, gosh, like, most of them!"

Prototypes

Designed three Java language extensions to address common use cases raised by developers.

- Support for transitively immutable objects, non-transitive immutable objects, and objects that are mutably only during construction phase
- Two currently implemented, one in progress

Evaluation with user studies

Use participatory design techniques. Give developers tasks and elicit how they would solve them before introducing extensions we are testing. Have developers then use features (or standard Java), eliciting their thoughts. Evaluation based upon bugs, developer speed, effectiveness, and developer feedback.

- Pilot studies in progress

Publications:

- *"Empirical Evaluation of API Usability and Security,"* LAW workshop, associated with ACSAC '14
- *"Comparing Transitive to Intransitive Object Immutability"* accepted at PLATEAU workshop, associated with SPLASH '15
- *"A Course-Based Usability Analysis of Cilk Plus and OpenMP"* accepted at VL/HCC '15 conference
- *"Exploring Language Support for Immutability"* submitted to ICSE '16

Poor state management in API design is a serious problem for developers. Language features to assist designers and programmers will improve both system security and usability.

Contact: Sam Weber samweber@cert.org

Vulnerability Discovery

Principal Investigator

Dr. Edward Schwartz

Edward Schwartz, PhD, is a research scientist in the CERT Division. Dr. Schwartz's research focuses on employing both dynamic and static binary analysis to find vulnerabilities. He is additionally interested in automatic methods for evaluating and bypassing commodity software defenses such as Address Space Layout Randomization (ASLR) and Data Execution Prevention (DEP).

Background

The goal of this project is to reduce the number of vulnerabilities in critical DoD and U.S. government (USG) systems by advancing and transitioning novel research in vulnerability discovery to high-impact DoD and USG stakeholders. This project is focused on advancing the state-of-the-art in research and the state-of-practice of stakeholder operations in two categories of DoD-critical system security: (1) automated and sound vulnerability discovery and prioritization in both traditional and non-traditional (mobile) computing platforms and (2) vulnerability discovery and correlation in emerging networked technologies. A tertiary goal is to transition our work to the DoD acquisition community as well as learn from DoD stakeholders to guide future research. Adoption of the proposed techniques into the DoD software acquisition and support processes will result in software applications that are hardened—and more secure—before and after they are deployed into the DoD infrastructure.

Approach

Improved Vulnerability Uniqueness Determination. CERT's fuzzing tools are becoming so effective that they regularly generate more crashing test cases than a vendor can reasonably triage. For example, CERT recently found over 40,000 crashes in a major software vendor's product that were judged likely to be exploitable. Although each of these crashes was determined to be unique by fuzzy call stack hashing, manual analysis has shown the actual number of unique crashes is much lower. If we could precisely determine which crashes are unique, we would be able to discard redundant crashes and supply more actionable data to vendors. To this end, we propose to employ dynamic instrumentation to record crashing test cases and use artifacts from the saved executions to cluster crashes. The goal of vulnerability uniqueness is, when given a large set of crashing inputs C (e.g., from fuzzing or sound, automatic vulnerability discovery), to output the smallest subset C´ such that if a vendor analyzes and fixes all crashes in C´, then as a byproduct they will have fixed all crashes in C as well.

Improved Automated Black Box Testing Techniques. Our FY2014 experimentation compared five fuzzing seed file selection algorithms [Alexandre 2014]. However, there is still work to be done to improve seed file corpus distillation and measure correlation of representative seed file sets across a wide diversity of software. We are experimentally comparing fuzzing results of minimum set corpuses across a range of software.

Extend Automated and Sound Vulnerability Discovery. The current techniques leveraged by automated and sound vulnerability discovery (ASVD) systems, such as Mayhem, are effective but slow compared to alternative techniques such as fuzzing. We are creating a hybrid tool that uses the AFL fuzzer to rapidly explore large portions of a program's input space. When AFL reaches its limits, we then engage Mayhem to produce inputs that drive execution to new parts of the program. These new inputs are then fed back to AFL, so that fuzzing can explore different parts of the program.

Artifacts

- Prototype tools
  - **Uniqueness determination prototype.** This prototype allows us to determine which vulnerabilities are triggered by a crashing test case.
  - **SMART: The Synergistic Mayhem AFL Research Tool.** This tool integrates a state-of-the-art black-box fuzzer (AFL) with an Automated and Sound Vulnerability Discovery tool (Mayhem).
- Submissions to research-oriented security conferences and/or journals
  - Submission to Network and Distributed Security Symposium (NDSS 2016): "One, Two, . . . Three? Counting Vulnerabilities is Harder than You Think." **Abstract:** In this paper, we present a new methodology for precisely and naturally defining vulnerabilities through the creation of patches. These patches are then used to identify which vulnerabilities a particular crasher triggers. We use our methodology to (1) evaluate the state of the art in vulnerability uniqueness: stack backtrace hashing, (2) measure the effects of address sanitization on fuzzing, (3) assess the impact of seed selection, and 4) determine whether different fuzzers find different vulnerabilities. Our results offer pragmatic guidance for more effective fuzzing, including evidence that current solutions such as stack backtrace hashing often incorrectly classify a vulnerability's uniqueness, which can lead to inaccurate vulnerability counts.
  - We are also planning a submission describing our SMART tool. Our main contribution is showing that by synchronizing a fuzzer and concolic tester, we can find more bugs than by running the tools separately.

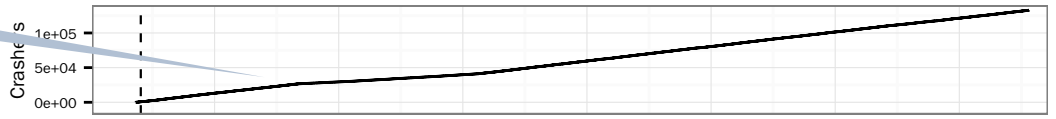
Vulnerability Discovery
Solving the vulnerability uniqueness problem

Current vulnerability discovery techniques such as black-box fuzz testing and concolic testing are so effective that they routinely find hundreds of thousands of crashers, which crash the target program. We created a new methodology for precisely and naturally defining vulnerabilities through the creation of patches. We use our methodology to study important questions regarding the practice of fuzzing.

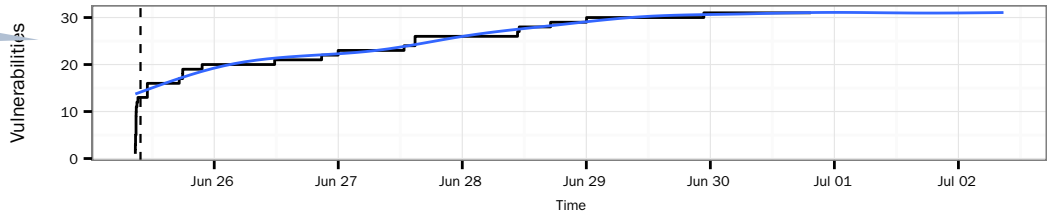
Experiment setup

We fuzzed ImageMagick5.3.0 for a week under various configurations, which yielded over 130,000 crashes. We patched each crash using our methodology, which yielded 31 vulnerabilities. We used this data to answer:

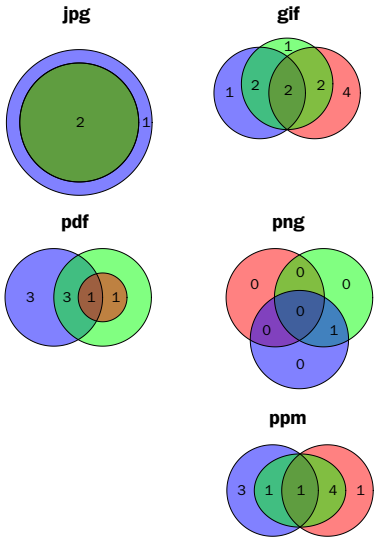
Crash rate is constant



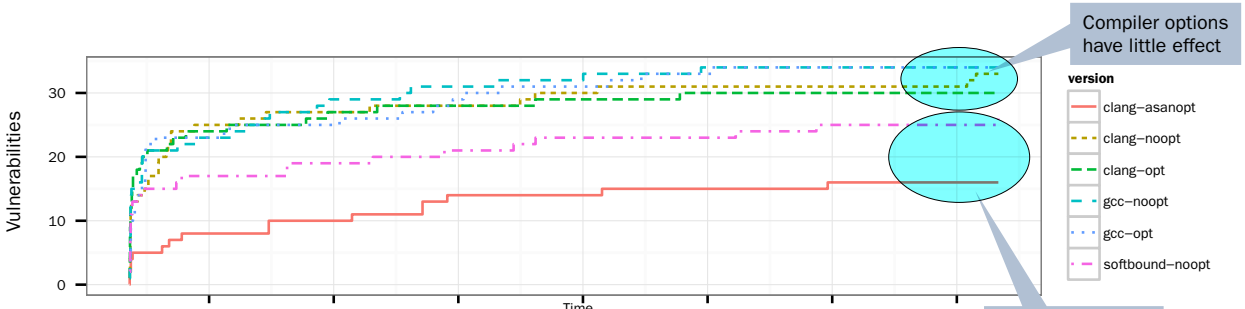
Vulnerability discovery rate slows down dramatically



Different seed files discover different vulnerabilities

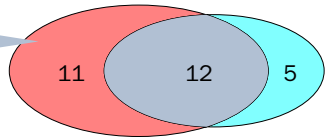


Compiler options have little effect



Sanitizers may not be worth their overhead

Different fuzzers find different vulnerabilities



Most crashes trigger multiple vulnerabilities

Vuls	1	2	3	4	5
Crashes	45859	79626	6860	21	1

Contact: Edward Schwartz eschwartz@cert.org

Cybersecurity via Signaling Games

Principal Investigator

Dr. Will Casey

William Casey, PhD, develops computational algorithms for malicious code detection, characterization, and analysis to expose the malicious design goals (of malware) or aberrant behaviors of software (vulnerabilities/exploits).

Background

Recently, we[1] applied information asymmetry game models to control insider threat (for which we won a best paper award at the seventh *ACM CCS International Workshop on Managing Insider Security Threats*). The evolution of this work goes back to our approach to cybersecurity via signaling games with additional mechanism provided by verifiers and recommenders. In our earlier work, we made clear that cybersecurity depends not only on the states and actions supported by the Cyber-Physical structures, but also on the individual utilities of agents who operate with partial and asymmetric information associated with the states/configuration of devices, properties of systems, and incentives of other agents. These utilities can be modeled by information-asymmetric signaling games to understand adversarial and deceptive utilities. The aggregate outcomes of these games determine the macro-level dynamics of the social-technological system.

We construct an agent-based system of evolutionary strategies that play various forms of signaling games, in order to understand better the behaviors of cyber-systems and the ways in which various security components may affect a system's safety and liveness properties. We tailor the basic information asymmetric signaling game to several cybersecurity scenarios including attack detection, malware, pattern verification, vulnerability testing, user compliance, insider threats, and identity-based attacks. We use computer simulation experiments as well as analytic calculation to better understand these specific scenarios and characterize effects of re-engineering the micro level interactions of these signaling games.

Our study considers several novel mechanism designs including bio-inspired self-adaptive systems, closed loop risk sensitive compliance control, and the design of a recommendation/verification system supported by model-checking and machine-learning agents to reduce informational asymmetries and degrade the operational period for deceptive attacks.

To illustrate the nature of information asymmetries and non-cooperative strategies in social technological systems, consider the Flashlight app for Android phones that also tracks the device's GPS positions [Kassner 2013]. The smartphone app is advertised to enable the phone to act as a flashlight. However, the additional designed component (i.e., the GPS tracker) is not advertised or revealed; rather, it is held by the software distributor as private information—at least during the operational attack period where trusting users remain unaware of the software's full design. The Flashlight app collects GPS data from the device and compromises users' privacy without their knowledge. This scenario describes a loss of private information; it generalizes to an adversarial and engineered attempt to increase informational asymmetries, and this action exposes the trusting user to various risks and exploitation possibilities, any of which may hold grave and negative consequences.

Approach

Our problem is aimed directly at how users may establish trust, manage risk, and mitigate deception while their strategic options are bounded by decision-making constraints inherent to cyber-systems (e.g., a user could theoretically scrutinize by hand all possible computations of an app and discover its reachable states).

Our approach features a central game-theoretical model and mathematical means to create and explore mechanisms that allow us to examine how micro-level interactions affect macro-level bulk behaviors. Agents interact and explore a wide range of strategic constructs; the essential game affects device states, system properties, and agent's choice of utilities. In this setting deceptive strategies induce and exploit information asymmetries and we are able to explore for effective counter strategies (e.g., agent based trace learning for malware, epistatic signaling and minority games, compliance control, conformance checking) that enhance trust.

Our approach has led us to suggest a social technological response carried out by agent types whose organization is aligned with the desired properties of cyber-systems (e.g., verifiers and recommender players—verifiers dynamically maintain the safety properties and recommenders dynamically maintain the liveness properties).

1 The CERT team of Dr. Will Casey, Dr. Jose Morales, Dr. Rhiannon Weaver, Evan Wright, in collaboration with Dr. Bud Mishra (New York University).

Cybersecurity via Signaling Games

Toward a deception-free cyber future

Problem: How can we establish trust, manage risk, and mitigate deceptive cyber attacks when decision-making is constrained by limited awareness of vulnerabilities, threats and systems properties.

Proposed Solution: A fundamental model of human actions and the formal machine properties they affect.

Our approach: A **game-theoretical** model to simultaneously study systems states/properties and human incentives:

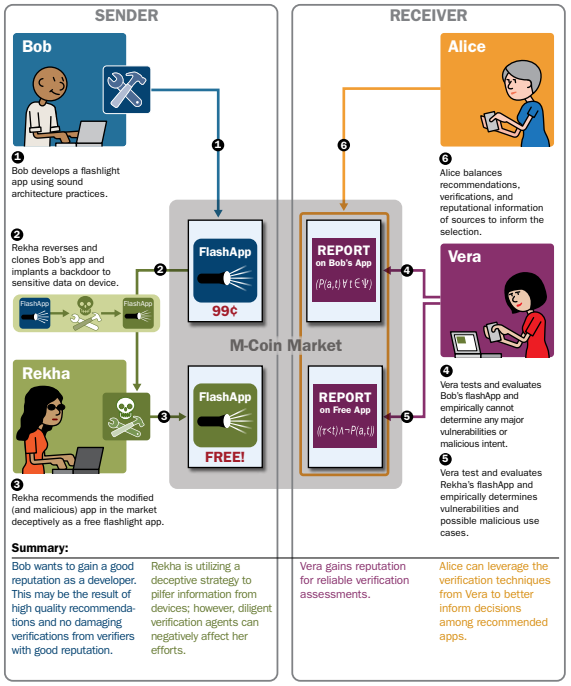
- Gives mathematical (and virtual) means to create and explore a wide range of mechanisms that re-design the microcosm of interactions
- Agent based model/simulation and analytic calculation applied toward understanding behavior modes in social technological systems

Applications: Using the game model we have:

- Studied multiple cybersecurity problems
- Qualitatively characterized modes of social-technological systems with adversarial strategies
- Identified novel mechanisms for mode selection including self-adaptive systems and risk-sensitive controllers

Key insight: Cybersecurity depends on

- The formal states, properties, and actions supported by the cyber-physical structures
- Utilities of agents operating with limited information of machine properties, device states, configurations. Agent utilities (or types) are not common knowledge leading to asymmetries.
- **These Utilities can be formalized with information-asymmetric signaling games to model adversarial and deceptive strategies.**



In cyber systems, several information asymmetric games can arise. This image illustrates malware injection and malicious recommendation.

The Scenario:

To illustrate the nature of information asymmetries and non-cooperative strategies in social-technological systems, consider the 'flashlight' app that tracks a device's GPS positions. The app is advertised as benign but the GPS tracker component is hidden by the software distributor – at least during the 'operational attack period' where trusting users remain unaware of that capability. The flashlight app collects GPS data from the device and compromises the users' privacy without their knowledge. This scenario describes a loss of private information but generalizes to an adversarial attempt to increases informational asymmetries, and this disposes the trusting user to various risks and grave exploitation possibilities.

Explored applications:

- **Fundamental dynamics of agent based systems.** Computer simulations of evolutionary games and mutable strategies explore the qualitative system modes parameterized by costs/benefits of signaling games. Talk is cheap, and costly signaling helps. Scaling a 'market of proof checkers' with an m-coin mechanism provides a way to impute costly signaling and a strong recovery pathway for systems under high levels of deception.
- **Epistatic signaling and minority games.** Considers games over multiple vulnerabilities (exploits) and examines how a social system may response to curtail evolving deceptions. Preferential early mover advantages have similar effects to maintaining strong global effectiveness measures but will be easier to implement.
- **App markets and malware:** We consider a means to create and evolve defensive strategies to malware built from semi-supervised learning of formal properties associated with malware traces. We outline a means to deploy these defenses with a Recommendation/Verifier System.
- **Insider threat and risk sensitive compliance controller.** We consider the problem of intentional and unintentional malicious insiders and the relation between their actions/signals and the risk they cause. We discover a means to estimate risk and actuate compliance incentives in a closed control loop.

Publications:

- **Awarded Best Paper: "Compliance Control: Managed Vulnerability Surface in Social-Technological Systems via Signaling Games,"** 2015 ACM CCS International Workshop on Managing Insider Security Threats
- **"Cyber Security via Minority Games with Epistatic Signaling,"** 2014 International Conference on Bio-inspired Information and Communications Technologies
- **"Agent-Based Trace Learning in a Recommendation-Verification System for Cybersecurity,"** 2014 IEEE International Conference on Malicious and Unwanted Software
- **"Cyber Security via Signaling Games: Toward a Science of Cyber Security."** 2014 International Conference on Distributed Computing and Internet Technology.

Future Work:

- Formalize notion of deception and risk in games (utilities/properties).
- Application to identity deceptions and Sybil attacks.
- Scaling the agent based recommendation/verification system.
- Policy optimization built on data science and inference.

Contact: W. Casey, J.A. Morales, R. Weaver, E. Wright at CMU SEI , B. Mishra (Courant Inst. NYU)

Cybersecurity References

[Alexandre 2014]
Alexandre, Rebert et al. Optimizing Seed Selection for Fuzzing. Pages 861-875. In *Proceedings of the 23rd USENIX Security Symposium (USENIX Security 14)*. San Diego, CA (USA). August 2014. https://www.usenix.org/conference/usenixsecurity14/technical-sessions/presentation/rebert

[CERT/CC 2013]
CERT/CC Malicious Code Team. Sogu/Plug-X Longitudinal Report. CERT/CC-2013-59. November 2013.

[Chin 1988]
Chin, John P. et al. Development of an instrument measuring user satisfaction of the human-computer interface. Pages 213-218. In Proceedings of the *SIGCHI conference on Human factors in computing systems. (CHI'88)*. Washington, D.C., USA. 1988.

[ElWazeeer 2013]
ElWazeer, Khaled, et al. Scalable variable and data type detection in a binary rewriter. Pages 51-60. In *Proceedings of the 34th ACM SIGPLAN Conference on Programming Language Design and Implementation.* Seattle, WA (USA). June 2013.

[Ellis 2007]
Brian Ellis et al. The Factory Pattern in API Design: A Usability Evaluation. Pages 302-312. In *Proceedings of the 29th International Conference on Software Engineering (ICSE'2007)*. Minneapolis, MN (USA). May 2007.

[Fontana 2012]
Fontana, F. A. et al. Dpb: A benchmark for design pattern detection tools. Pages 235-244. In *Proceedings of the 16th European Conference on Software Maintenance and Reengineering.* Szeged, Hungary. March 2012.

[Havrilla 2013]
Havrilla, Jeff, et al. Trends in Malware Sophistication: The Industrialization of Malicious Code. CERT/CC-2013-43. August 2013.

[Jin 2014]
Jin, Wesley et al. Recovering C++ Objects From Binaries Using Inter-Procedural Data-Flow Analysis. In *Proceedings of the 3rd ACM SIGPLAN on Program Protection and Reverse Engineering Workshop 2014*. San Diego, CA (USA). January 2014. http://dl.acm.org/citation.cfm?id=2556465&CFID=544065544&CFTOKEN=61670460

[Kassner 2013]
Kassner, M. Android flashlight app tracks users via gps, ftc says hold on [blog post]. *Security*. December 11, 2013, 9:49 PM PST. http://www.techrepublic.com/blog/it-security/why-does-an- android-flashlight-app-need-gps-permission/

[Lee 2011]
Lee, JongHyup et al. TIE: Principled Reverse Engineering of Types in Binary Programs. In *Proceedings of the 18th Network and Distributed System Security Symposium*. San Diego, CA (USA). February 2011. http://www.isoc.org/isoc/conferences/ndss/11/pdf/5_3.pdf

[Nielsen 1993]
Nielsen, Jakob. *Usability Engineering*. Academic Press. 1993.

[Porche III 2013]
Porche, Isaac R. III et al. Redefining Information Warfare Boundaries for an Army in a Wireless World. RAND Corporation, 2013. http://www.rand.org/pubs/monographs/MG1113.html

[Stylos 2007]
Stylos, Jeffrey & Clarke, Steven. Usability Implications of Requiring Parameters in Objects' Constructors. Pages 529-539. In *Proceedings of the 29th International Conference on Software Engineering (ICSE'2007)*. Minneapolis, MN (USA). May 2007.

[Yoon 2011]
Yoon, YoungSeok & Myers, Brad A. Capturing and Analyzing Low-Level Events from the Code Editor. Pages 25-30. In *Proceedings of PLATEAU 2011: Workshop on Evaluation and Usability of Programming Languages and Tools*. Portland, Oregon (USA). October 2011. http://www.cs.cmu.edu/~NatProg/papers/plateau2011-yoon.pdf

URLs are valid as of the publication date of this document.

CyLab at CMU

The CMU SEI is pleased to include a sample of CyLab research in its research review. Carnegie Mellon University CyLab is a bold and visionary effort, which establishes public-private partnerships to develop new technologies for measurable, secure, available, trustworthy, and sustainable computing and communications systems. For more information, visit https://www.cylab.cmu.edu/.

Dr. Lorrie Cranor

Professor, Computer Science and Engineering & Public Policy

Director, CyLab Usable Privacy and Security Laboratory

Co-director, MSIT-Privacy Engineering Master's Program

In the SEI 2015 Research Review, Dr. Cranor provided insight into CMU research regarding usable privacy and security.

About the speaker
Lorrie Faith Cranor, PhD, is a professor of Computer Science and of Engineering and Public Policy at Carnegie Mellon University where she is director of the CyLab Usable Privacy and Security Laboratory (CUPS) and co-director of the MSIT-Privacy Engineering Master's Program. She is also a co-founder of Wombat Security Technologies, Inc.

Dr. Cranor has authored over 150 research papers on online privacy, usable security, and other topics. She has played a key role in building the usable privacy and security research community, having co-edited the seminal book *Security and Usability* (O'Reilly 2005) and founded the Symposium on Usable Privacy and Security (SOUPS). She chaired the Platform for Privacy Preferences Project (P3P) Specification Working Group at the W3C and authored the book *Web Privacy with P3P* (O'Reilly 2002). She also directs an NSF-funded Integrative Graduate Education and Research Traineeship (IGERT) program on usable privacy and security

Dr. Cranor has served on a number of boards, including the Electronic Frontier Foundation Board of Directors, and on the editorial boards of several journals. In 2003, she was named one of the top 100 innovators 35 or younger by *Technology Review* magazine. In 2014, she was named an ACM Fellow for her contributions to usable privacy and security research and education. She has received faculty research awards from IBM, Microsoft, and Google. She came to CMU in December 2003 after seven years at AT&T Labs-Research. While at AT&T, she also taught in the Stern School of Business at New York University.

Dr. Cranor consults for companies and non-profits on privacy policies, P3P, usable privacy and security, and technology policy. She has served as an expert witness in patent litigation, privacy cases, and in cases challenging the constitutionality of Internet harmful-to-minors laws, including the ACLU's successful challenge to the 1998 Children's Online Protection Act.

Dr. Marios Savvides

Research Professor, Electrical & Computer Engineering

Founder and Director, CyLab Biometrics Center

In the SEI 2015 Research Review, Dr. Savvides discussed CMU research into unconstrained face and iris recognition in real-world DoD and law enforcement applications.

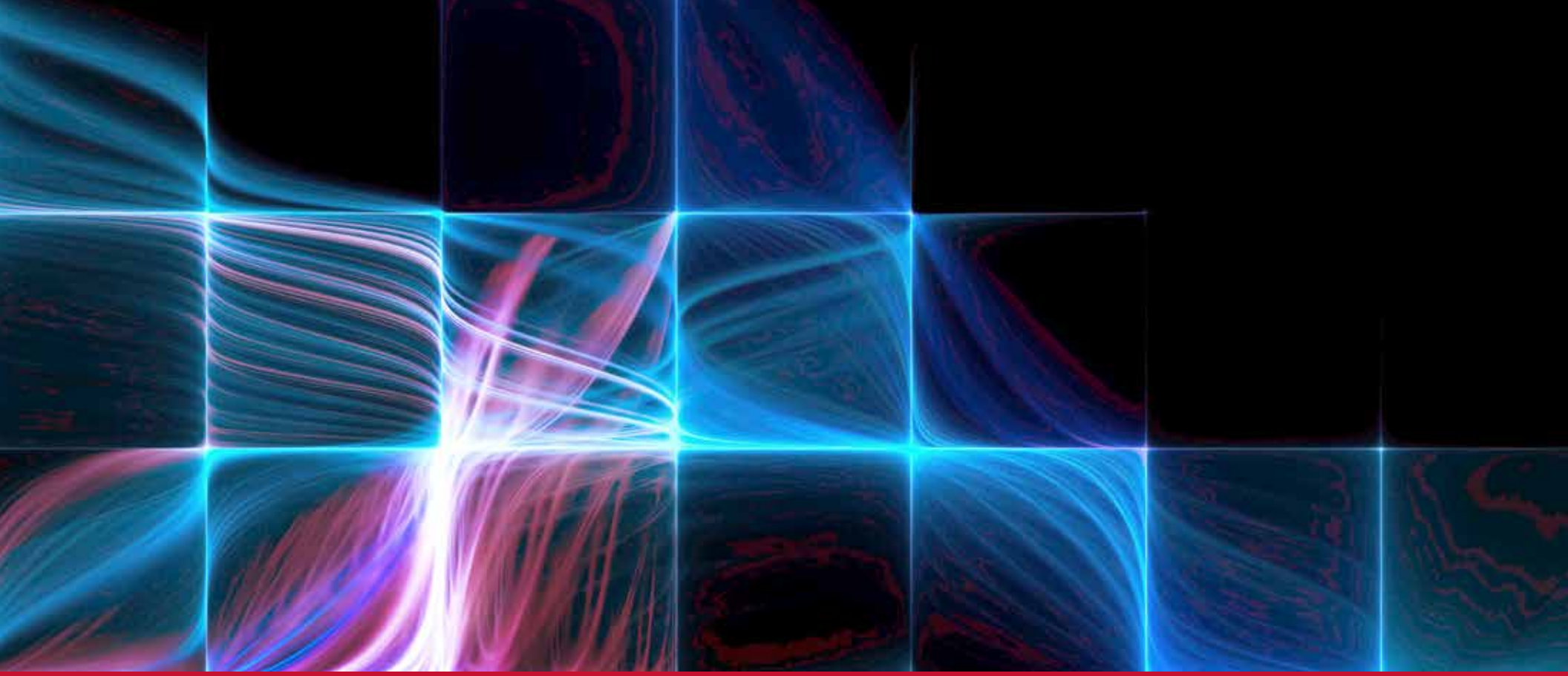
About the speaker
Marios Savvides, PhD, is the founder and director of the CyLab Biometrics Center at Carnegie Mellon University and is Research Professor at the Electrical & Computer Engineering Department and CMU CyLab. He is also one of the researchers tapped to form the Office of the Director of National Intelligence (ODNI) 1st Center of Academic Excellence in Science and Technology.

Dr. Savvides' research is mainly focused on developing algorithms for robust face and iris biometrics as well as pattern recognition, machine vision, and computer image understanding for enhancing biometric systems performance. His achievements in this area include leading the research and development in CMU's participation at NIST's Face Recognition Grand Challenge 2005 and developing the first long-range iris system capable of capturing enrollment quality irises 40 feet away.

Recently, Dr. Savvides has been spearheading and leading CMU efforts in the Face Recognition Grand Challenge (FRGC) and the Iris Challenge Evaluation (ICE), which are parts of NIST's efforts in evaluating and identifying key performance technologies in face recognition and iris recognition.

Dr. Savvides is on the program committee on several biometric conferences such as IEEE BTAS, ICPR, SPIE Biometric Identification, IEEE AutoID and others. He also has organized and co-chaired the Robust Biometrics Understanding the Science & Technology (ROBUST 2008) conference.

Dr. Savvides has authored or co-authored more than 170 journal and conference publications, including several book chapters in the area of biometrics. In addition, he has served as the area editor of the Springer's *Encyclopedia of Biometrics*. He is the IEEE Vice-President of Education of the IEEE Biometric Council. He has filed over 20 patent applications in the area of biometrics and is the recipient of CMU's 2009 Carnegie Institute of Technology (CIT) Outstanding Faculty Research Award.



Carnegie Mellon University

About

The Software Engineering Institute (SEI) is a not-for-profit Federally Funded Research and Development Center (FFRDC) at Carnegie Mellon University, specifically established by the U.S. Department of Defense (DoD) to focus on software and cybersecurity. As an FFRDC, the SEI fills voids where in-house and private sector research and development centers are unable to meet DoD core technology needs.

Contact Us

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

Phone: 412.268.5800 | 888.201.4479
Web: sei.cmu.edu | cert.org
Email: info@sei.cmu.edu