SAND2022-7038C

Sandia National Laboratories

Exceptional service in the national interest

Evidence-Based Foundations for Software Engineering Practice in Scientific Computing

Reed Milewicz, Evan Harvey, Miranda Mundt, Derek Trumbo, Wesley Coomber

2022 Tri-lab Advanced Simulation & Computing Sustainable Scientific Software Conference (ASC S3C)

May 24th-26th, 2022

Illustrations from Undraw.co, with icons from The Noun Project (artists Adrien Coquet, Kiran Shastry, Gan Khoon Lay)

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia LLC, a wholly owned subsidiary of Honeywell International Inc. for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

U.S. DEPARTMENT OF ENERGY

NNSA

Overview

- Introduce **Evidence-Based Practice (EBP)** in Software Engineering
  - EBP → Integrating current best evidence from research with practical experience and human values to improve decision-making related to software development and maintenance.
- Showcase how our team has explored the use of EBP techniques in our work, and share our **lessons learned**.
  - When we combine peer-reviewed evidence with our professional experience and put it to use in real-world contexts, we learn.
  - We share and discuss what we learn as to build consensus around those practices.

Software Development Practice Continues to Evolve

ADVANCED DEVELOPMENT PARADIGMS

ADVANCED ARCHITECTURAL PARADIGMS

AI-Augmented Software Development

- Re-envisioned software development lifecycle
- Data & data models for AI-augmented paradigms
- New forms of evidence of quality
- Automated design, evolution, and analysis tools

Assuring Continuously Evolving Software Systems

- Theory for assured composition of evidence
- Modifiable assurance arguments
- Assurance argument templates
- Automatic detection if a system change invalidates an assurance argument
- Automatic system update recommendations based on operational data
- Tool chains for combining evidence to re-assure a system
- Scaled auto-code generation & repair
- Evidence of developer acceptance

Software Construction through Compositional Correctness

- Theory of composability for model-integrated computing and quality attributes
- Documented patterns and tools for composition notations, rules, & relationships
- "Smart composition" technologies
- Integrated tool chains to assure composed behaviors at scale before & during runtime
- Intelligent interacting formalisms & assurance capabilities

Engineering AI-Enabled Software Systems

- AI-enabled system specification methods
- Design and analysis methods for AI-enabled systems
- Testing practices for AI-enabled systems
- Data management in support of AI-enabled systems
- Uncertainty management methods
- Continuous monitoring & sustainment

Engineering Societal-Scale Software Systems

- New quality attributes based on human behavior at scale
- System instrumentation to monitor effects of system on social behavior
- Automated detection & protection against misuse of socio-technical platforms
- Platforms for continuously evolving socio-technical ecosystems

Engineering Quantum Computing Software Systems

- Expanded set of quantum algorithms
- Debugging tools & techniques
- Standardized software stack interfaces
- Domain-specific languages
- Profiling tools
- Hybrid classical quantum algorithms

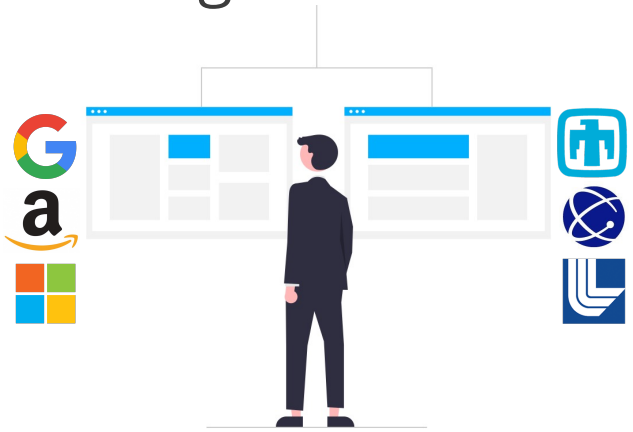
VISION
Humans and AI are trustworthy collaborators that rapidly evolve systems based on programmer intent

Research Focus Areas — **Research Topics** — **Research Topics** — **Research Focus Areas**

Carleton, Anita D., et al. *Architecting the Future of Software Engineering: A National Agenda for Software Engineering Research and Development*. Software Engineering Institute, Carnegie-Mellon University. 2021.

Staying Current With Best Practices is Challenging

- Being software professionals at the national labs, we bring to scientific computing a **rich heritage** of tools, techniques, and methodologies backed by over five decades of research and practice.
- We have a **responsibility** to act on the basis of the best available evidence as insights continue to emerge. But that's easier said than done!



What works well in conventional industry **may or may not translate** to our domain.

SE, DevOps, ITSM are **understudied** in scientific computing contexts.

We have **limited time and resources** to stay current with the latest findings and trends, and there is *always* something new to learn.

Problem Statement: How do we know we are staying current best practices and doing what's right for our customers?

What is Evidence-Based Practice?

The goal of **evidence-based practice (EBP)** in software engineering is to integrate current best evidence from research with practical experience and human values to improve decision-making related to software development and maintenance.

Dyba, Tore, Barbara A. Kitchenham, and Magne Jorgensen. "Evidence-based software engineering for practitioners." *IEEE software* 22.1 (2005): 58-65.

Parallels with Evidence-Based Practice in Medicine

Evidence-based medicine is not just about the research. Research is imperfect. And even if the evidence is perfectly quantifiable, neither your experience nor patient values are.

Part of the beauty and joy of medicine is that it can't be reduced down to a set of optimized algorithms. Instinct, judgment, and communication all play key roles.

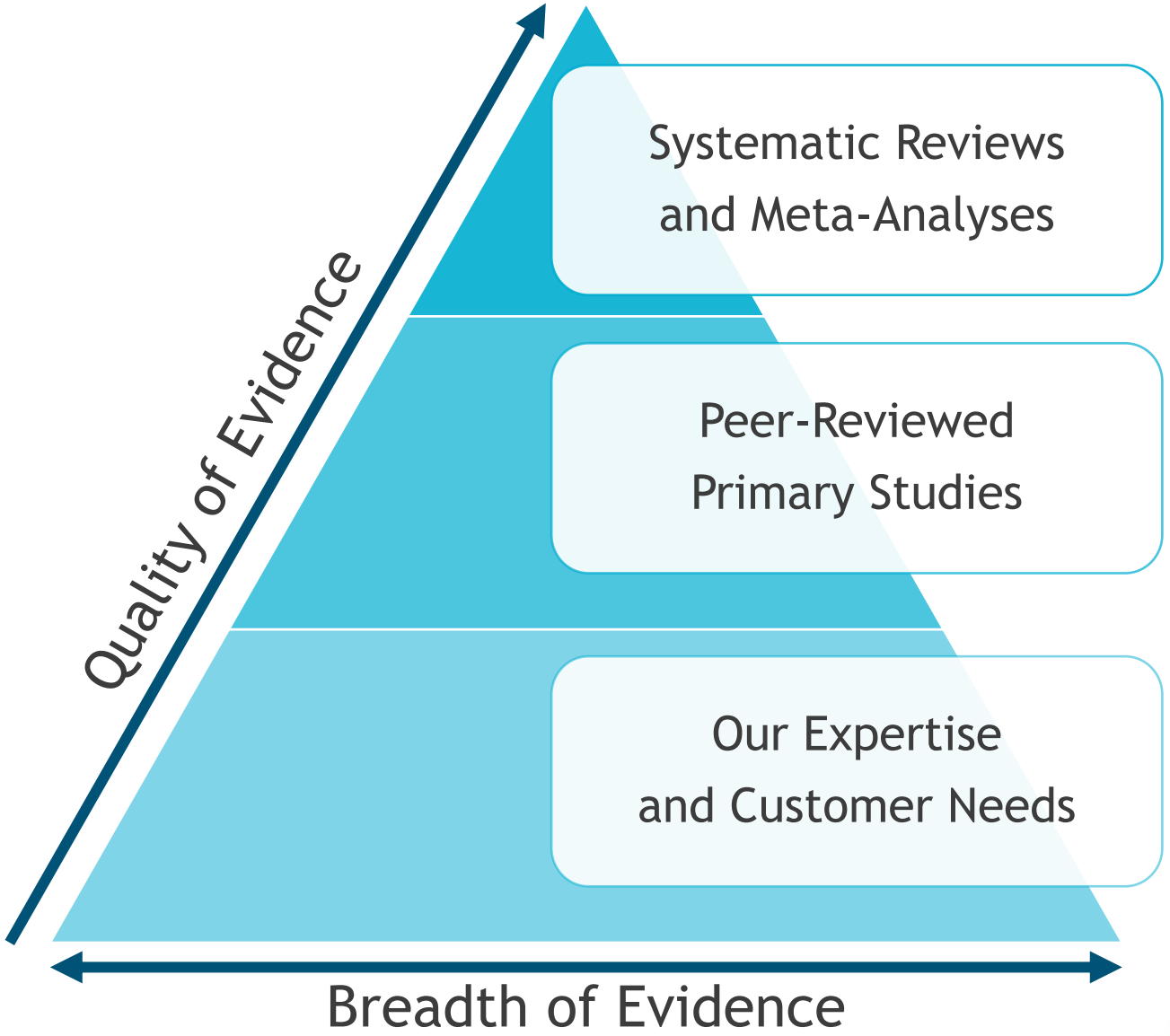
However, we still need the skills to appraise the evidence we're using, even if we can't perfectly measure and quantify its validity. Otherwise, we'd be practicing medicine in the dark, operating completely on faith that what we're doing is helping our patients.

Dr. Eric N. Strong, MD

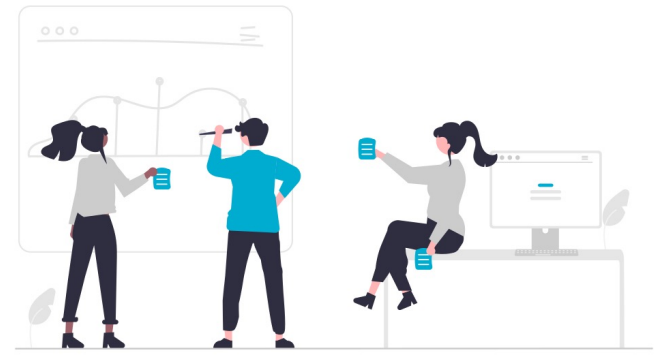
Strong, Eric. "An Introduction to Evidence-Based Medicine." *Strong Medicine;* YouTube. 2017 <https://www.youtube.com/P-G2veeYC1Q>

Navigating the Evidence Hierarchy

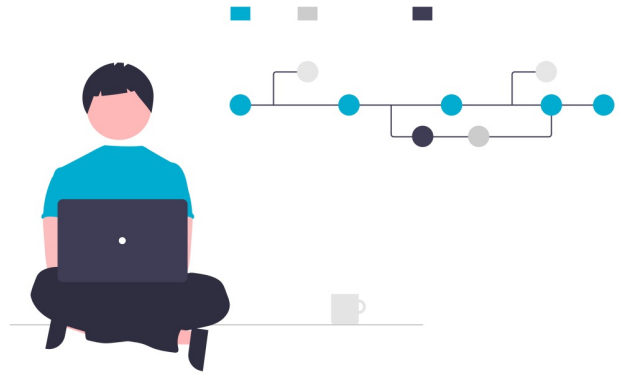
- The foundation of all decision-making is our **experiences as practitioners** and the **needs of our customers**.
- Incorporating high-quality evidence helps **reduce bias** and **mitigate risk**, enabling better decision-making.



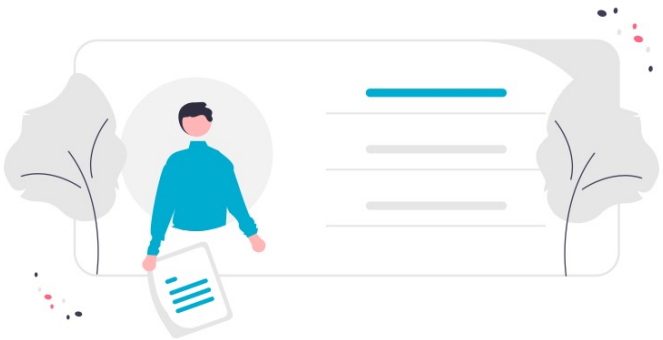
Example: How Do We Build Secure DevSecOps Infrastructure?



What is our **consensus as a team** on best practices in this space? What do we **already know** about this topic?



What has worked well in **previous DevOps pipeline solutions** we've built? How can we **extend** these solutions to incorporate security?

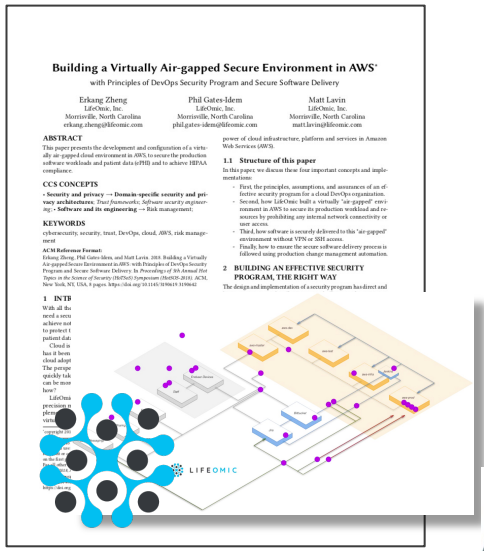


What do we know about what our customers **want and need**? What are their **values and priorities**?

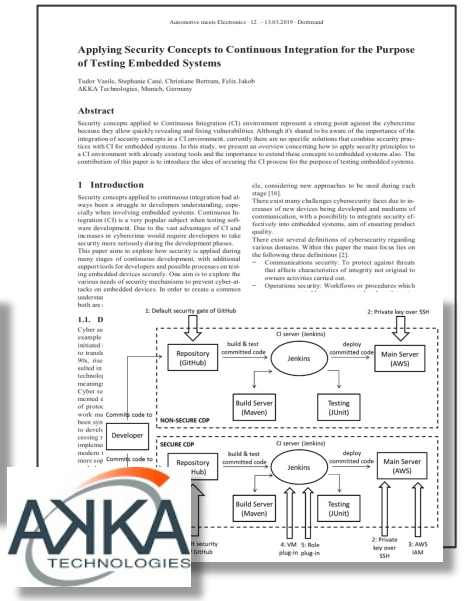
Example: How Do We Build Secure DevSecOps Infrastructure?

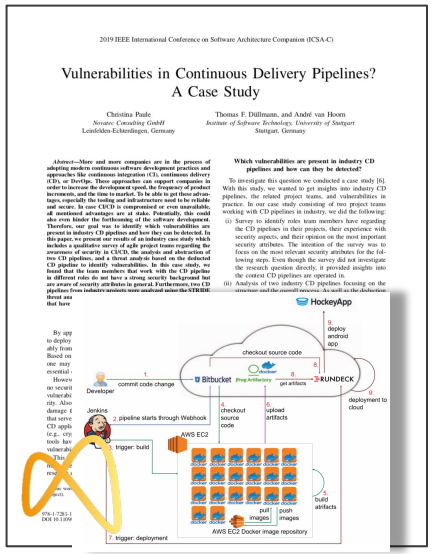
Peer-Reviewed Primary Studies

How have **others** approached this problem? Are there **vetted and independently peer-reviewed** case studies?

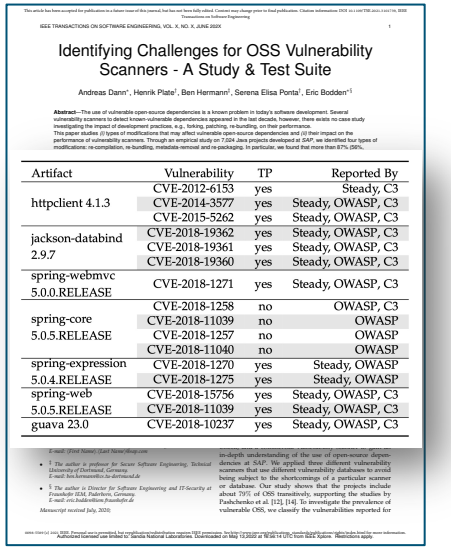


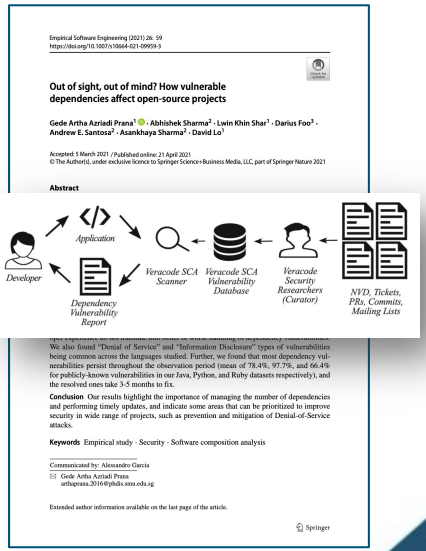
LIFEOMIC





Are there best practices in the use of **tools and techniques** that could help us in achieving our goals?





Example: How Do We Build Secure DevSecOps Infrastructure?

Systematic Reviews and Meta-Analyses

Information and Software Technology 141 (2022) 106700

Contents lists available at ScienceDirect

Information and Software Technology

journal homepage: www.elsevier.com/locate/infsof

Challenges and solutions when adopting DevSecOps: A systematic review

Roshan N. Rajapakse [a,b,*], Mansooreh Zahedi [a], M. Ali Babar [a,b], Haifeng Shen [c]

[a] CREST - The Centre for Research on Engineering Software Technologies, School of Computer Science, The University of Adelaide, Adelaide, Australia
[b] Cyber Security Cooperative Research Centre, Australia
[c] The HiLabb, Peter Faber Business School, Australian Catholic University, Sydney, Australia

ARTICLE INFO

Keywords:
DevOps
Security
DevSecOps
Continuous Software Engineering
Systematic Literature Review

ABSTRACT

Context: DevOps (Development and Operations) has become one of the fastest-growing software development paradigms in the industry. However, this trend has presented the challenge of ensuring secure software delivery while maintaining the agility of DevOps. The efforts to integrate security in DevOps have resulted in the DevSecOps paradigm, which is gaining significant interest from both industry and academia. However, the adoption of DevSecOps in practice is proving to be a challenge.
Objective: This study aims to systemize the knowledge about the challenges faced by practitioners when adopting DevSecOps and the proposed solutions reported in the literature. We also aim to identify the areas that need further research in the future.
Method: We conducted a Systematic Literature Review of 54 peer-reviewed studies. The thematic analysis method was applied to analyze the extracted data.
Results: We identified 21 challenges related to adopting DevSecOps, 31 specific solutions, and the mapping between these findings. We also determined key gap areas in this domain by holistically evaluating the available solutions against the challenges. The results of the study were classified into four themes: People, Practices, Tools, and Infrastructure. Our findings demonstrate that tool-related challenges and solutions were the most frequently reported, driven by the need for automation in this paradigm. Shift-left security and continuous security assessment were two key practices recommended for DevSecOps. People-related factors were considered critical for successful DevSecOps adoption but less studied.
Conclusions: We highlight the need for developer-centered application security testing tools that target the continuous practices in DevSecOps. More research is needed on how the traditionally manual security practices can be automated to suit rapid software deployment cycles. Finally, achieving a suitable balance between the speed of delivery and security is a significant issue practitioners face in the DevSecOps paradigm.

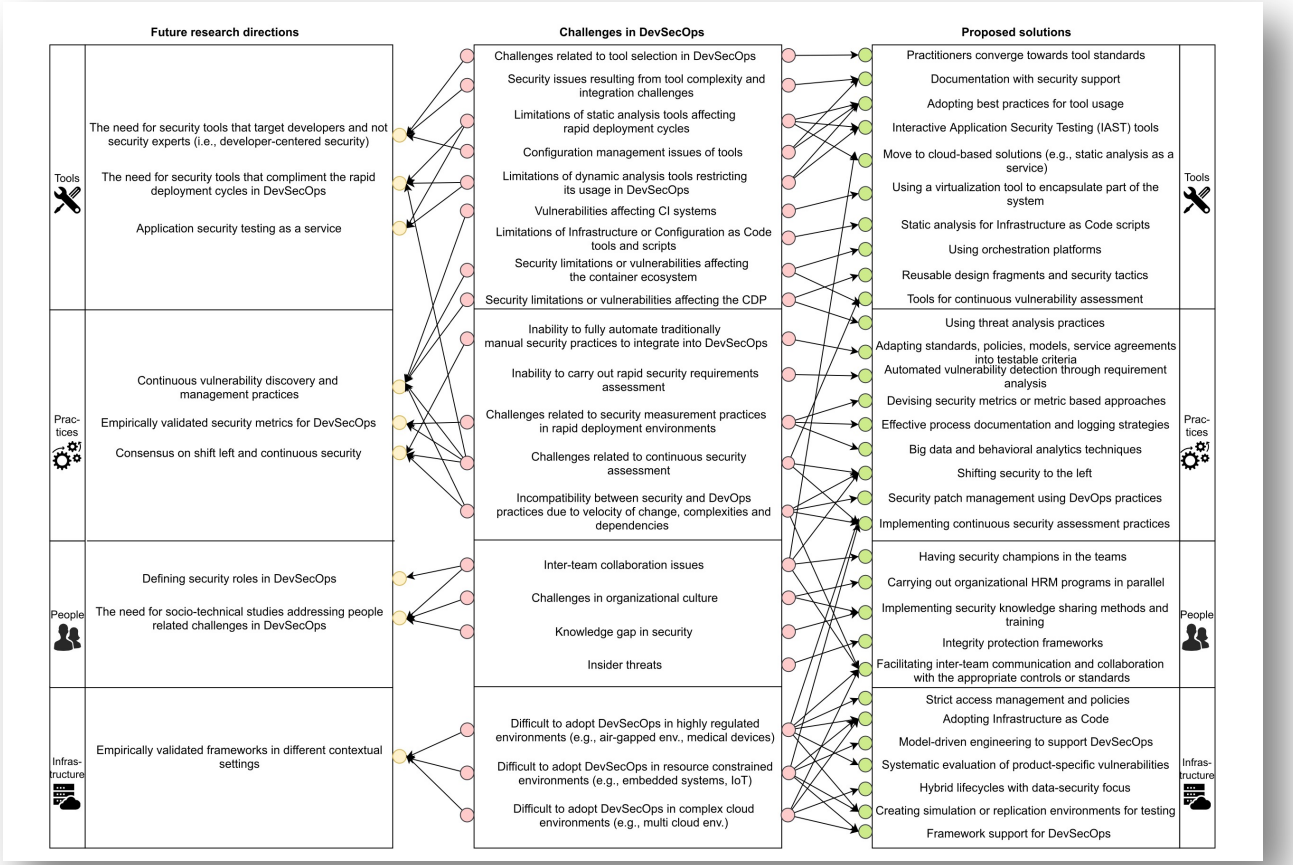
1. Introduction

DevOps (**Dev**elopment and **Op**erations) has led to a paradigm shift aimed at removing the traditional boundaries (or "silos") of the software development and software operations teams [1]. This shift resulted in reducing the time between committing a modification in a system and that change being placed in a production environment [2]. DevOps is currently a widely adopted software development paradigm in the industry [3]. This interest in adoption is due to the gains in business value reported by industry practitioners and academic researchers [4]. The most commonly reported benefit is the ability to deploy releases faster and more frequently [5]. However, the practices of rapid delivery have presented new challenges to organizations. One such challenge is ensuring the security of software outputs to stakeholders while maintaining the agility of DevOps [6].

Traditionally, security is treated as a non-functional requirement [10], which is handled at a later stage of the software development life-cycle [11,12]. Accordingly, a set of standard application security tests or activities are conducted on a software release. These activities either need substantial manual effort (e.g., security code review [13]) or are time consuming tasks (e.g., Dynamic Application Security Testing (DAST) [14]). Therefore, applying the same security tests in the context of DevOps would hinder the speed of deployments. At the same time, with the rising number of attacks, the security of software is critical in today's context, particularly in a cloud environment. There are

* Corresponding author at: CREST - The Centre for Research on Engineering Software Technologies, School of Computer Science, The University of Adelaide, Adelaide, Australia.
E-mail addresses: roshan.rajapakse@adelaide.edu.au (R.N. Rajapakse), mansooreh.zahedi@adelaide.edu.au (M. Zahedi), ali.babar@adelaide.edu.au (M.A. Babar), Haifeng.Shen@acu.edu.au (H. Shen).

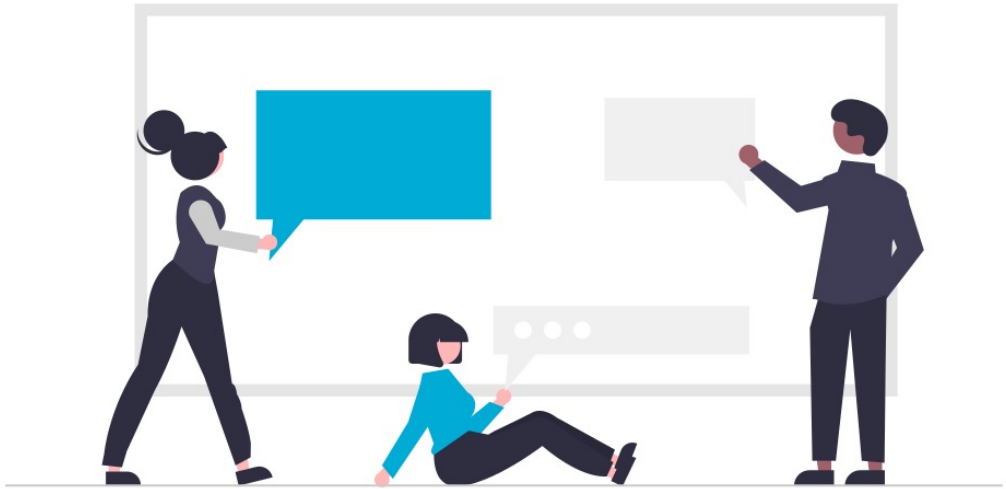
https://doi.org/10.1016/j.infsof.2021.106700
Received 14 March 2021; Received in revised form 22 July 2021; Accepted 27 July 2021
Available online 22 August 2021
0950-5849/



Are there **trends** in DevSecOps we should be aware of? Is the field **converging on certain solutions**?

Processes Our Team Has Experimented With

Software Engineering & Research
Department 1424



Best Practices Meetings

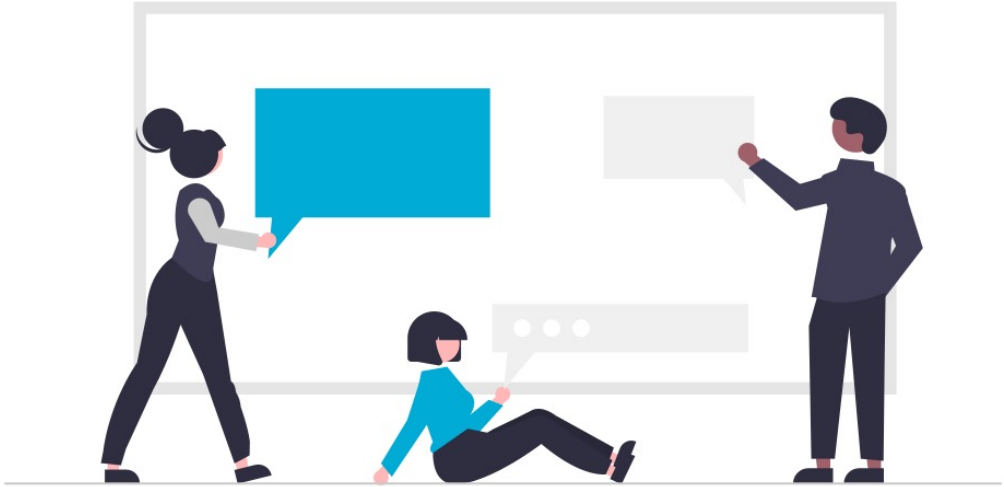
Rapid Reviews

When we combine peer-reviewed evidence with our professional experience and put it to use in real-world contexts, we learn. We then share and discuss what we learn to build consensus around those practices.

Milewicz, Reed, James Willenbring, and Dena Vigil. "Research, Develop, Deploy: Building a Full Spectrum Software Engineering and Research Department." *Research Software Engineers in HPC* (*RSE-HPC-2020*). 2020. SAND2020-11072C

Key Process: Best Practices Meetings

- We have to stay current with tools and best practices, and we must always be looking for better ways to design, develop, and maintain software. We must build **strong teams** and promote **long-term growth**.
- Our team holds weekly **Best Practices Meetings**, round-table discussions where team members join together to deliberate and discuss the processes and principles that lead to high-quality software.
- Examples include…
  - Strategies for backlog prioritization
  - Containers and how to use them
  - Understanding the Liskov Substitution Principle
  - How to conduct effective code reviews



Areas of Improvement Include…

Cultivating Knowledge and Skills	Enhancing Productivity
Empowering Independence	Facilitating Teamwork
Improving Decision-Making	Raising Morale

D. Trumbo and R. Milewicz, “Poster: Towards a Culture of Continuous Learning and Improvement within RSE Teams.” 2021 Collegeville Workshop on Scientific Software, 2021.

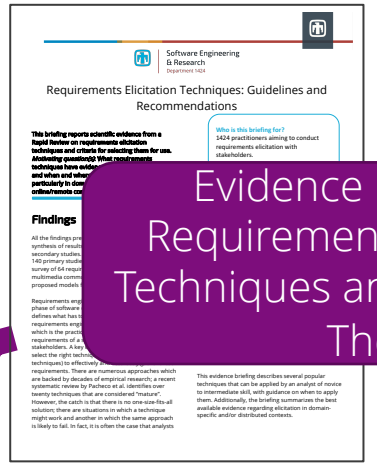
Key Process: Rapid Reviews

- A **Rapid Review (RR) Protocol** is a systematic, time-boxed literature review designed to deliver evidence in a timely and accessible way.
  - Motivated by **practical problems** and report results **directly to practitioners** in the field.
  - Simplify or omit certain steps from full systematic reviews, enabling turnaround times measured in **days rather than months**
- RR topics have included…
  - Requirements gathering techniques
  - Software quality incentivization
  - Best practices in CI/CD pipelines
  - Optimizations for containers

R. Milewicz, “Towards Evidence-Based Practice in Scientific Software Development,” Sandia National Lab.(SNL-NM), Albuquerque, NM (United States), Tech. Rep., 2020.

Example: Rapid Review on Requirements Elicitation Techniques

What requirements techniques have evidence for their effectiveness, and when and where should they be applied, particularly in domain-specific and/or online/remote contexts?


Requirements Elicitation Techniques: Guidelines and Recommendations

Evidence Briefing of Requirements Gathering Techniques and How to Use Them

Best Practices Meeting on RQ Elicitation

Ask → **Acquire** → **Appraise** → **Apply** → **Analyze**

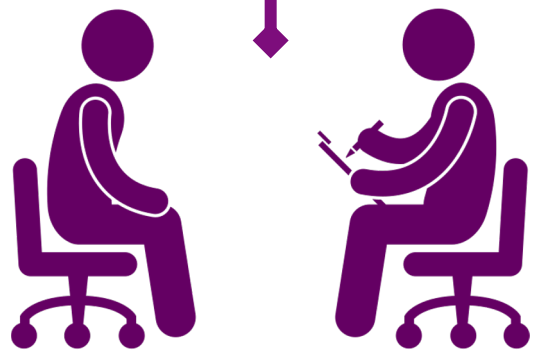
1000
149
38
19


Software Professionals' Attitudes Towards Video as a Medium in Requirements Engineering

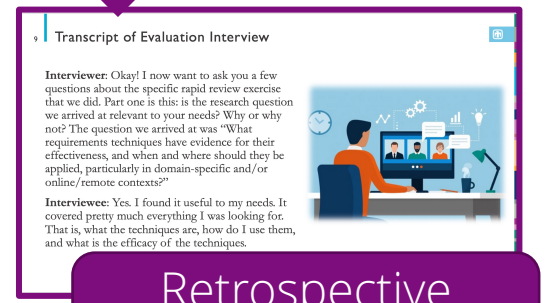

Contextual attributes impacting the effectiveness of requirements elicitation Techniques: Mapping theoretical and empirical research

Primary Studies

Secondary Studies



Apply Techniques in Real-World Contexts



Retrospective Interview About RR

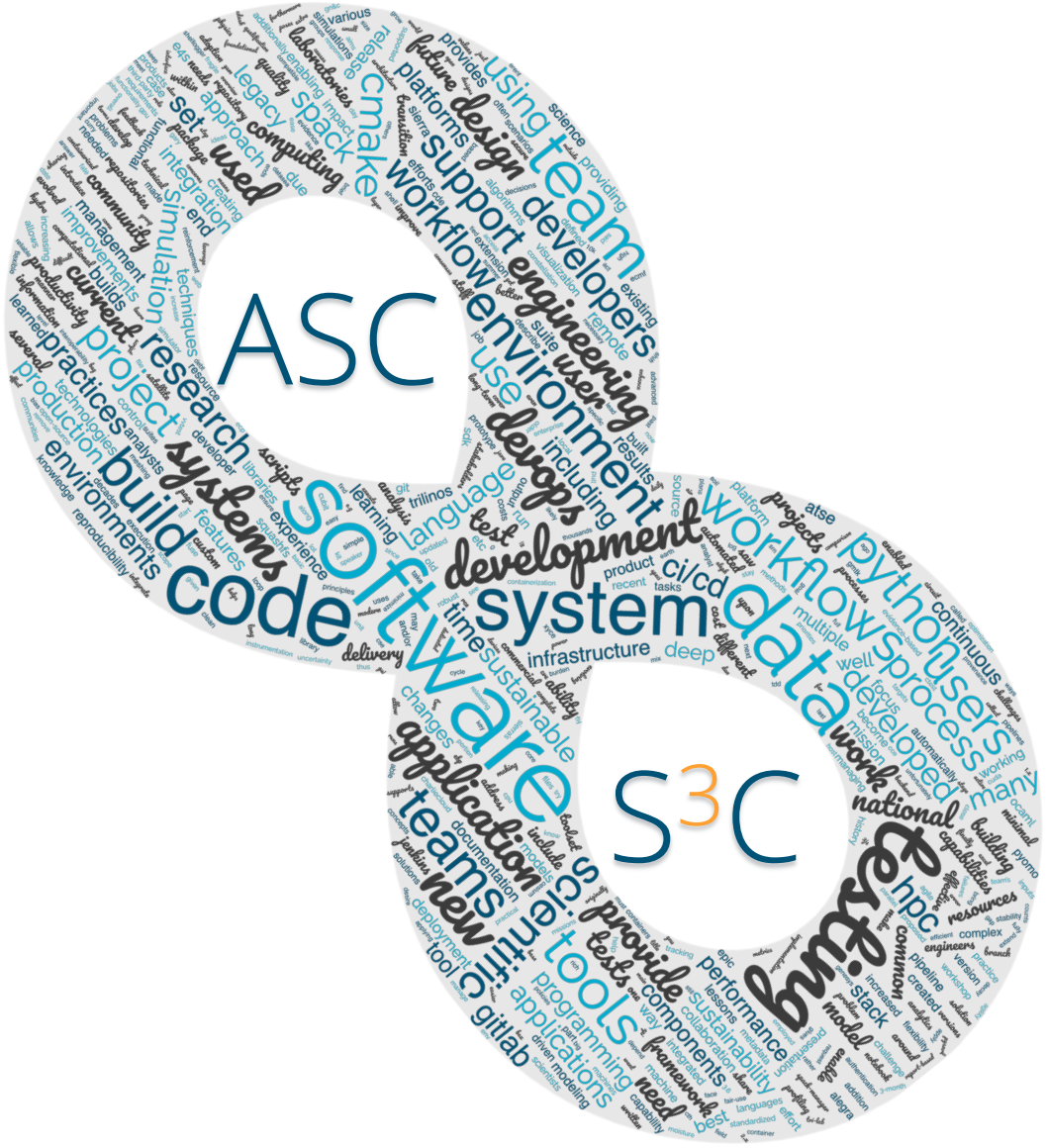
Discussion: Finding Common Ground With Evidence-Based Practice

- We accept that no team member is perfect, no team is perfect, and no product is perfect. There is always room for improvement.
- We should always strive for excellence, make continuous learning and improvement activities part of the team culture, and keep each other accountable.
- This requires a commitment to humility, honesty, forgiveness, self-reflection, and a willingness both to give constructive feedback and receive constructive feedback.

- We accept that we have a responsibility to act on the basis of the best available evidence as acquired through systematic and rigorous investigation.
- We should always strive to integrate current best evidence with practical experience and human values to improve our decision-making.
- This requires balancing imperfect research alongside our instinct, judgment, and communication. Even if the evidence is perfectly quantifiable, neither our experience nor customer values are.

Conclusion

- In this talk, we…
  - We **defined** evidence-based practice (EBP) in software development means and why it matters.
  - We **described** the techniques our department has explored to build consensus around best practices.
  - We **discussed** how to unite principles of team-based continuous learning and improvement with empiricism in software engineering.
- Food For Thought: On the right is a word cloud of the topics covered by all the talks and tutorials at ASC S³C. Consider all of the challenges that we face as practitioners. How might EBP techniques help us meet those challenges?



ABOUT THE SANDIA ANGLES TEMPLATES

Create impactful presentations, reports, and visuals with Sandia branded PowerPoint templates.

FEATURES

- 16:9 HD widescreen format
- Embedded Sandia font & colors
- Professional photo and text layouts available in the Sample Layouts deck
- Access fully editable charts, maps, and icons in the PowerPoint Graphics Library
- Easy to use placeholders crop photos without distortion
  Note: To reduce file sizes, templates do not include Images. UUR photos can be accessed at Sandia's Flickr page.
- Before submitting to Sandia Review and Approval, ensure only the appropriate markings are applied to content slides and Slide Masters.

Questions?

Get immediate support from Creative Services.

NM: (505) 844-7167 | **CA:** (925) 294-1010 | creative.sandia.gov

Revised 03.04.21

Want more?

Browse additional design templates at creative.sandia.gov