Industrial Scale Proof Engineering for Critical Trustworthy Applications (INSPECTA)

DARPA PROVERS : Pipelined Reasoning of Verifiers Enabling Robust Systems

seL4 Summit | 15 October 2024
Darren Cofer | Principal Fellow
Applied Research & Technology (ART)

PROVERS PROGRAM OBJECTIVES

Develop automated, scalable formal methods tools that are integrated into traditional software development pipelines.

DEFENSE & AEROSPACE

Enable traditional software developers to incrementally produce and maintain high-assurance national security systems.

aws

RTX

GD

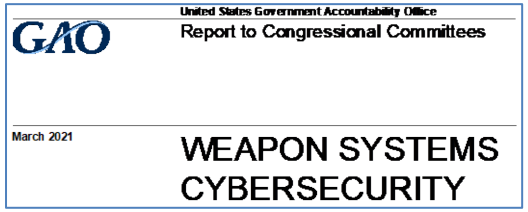
Adoption of formal methods in Defense Industrial Base development processes

Our weapon systems are vulnerable

2021 DOT&E Annual Report

Although all exercises that Director Operational Test and Evaluation (DOT&E) participates in include a DOT&E-sponsored Red Team, exercise authorities **seldom permit warfighters to experience representative adversarial cyber effects because of the risk of degrading other training objectives**. The net result of this limitation is a false sense of confidence by warfighters and leadership alike: **failure to train in realistic cyber environments leaves warfighter skills and playbooks immature**.

General Accounting Office (GAO) Report to Congressional Committees GAO-21-179

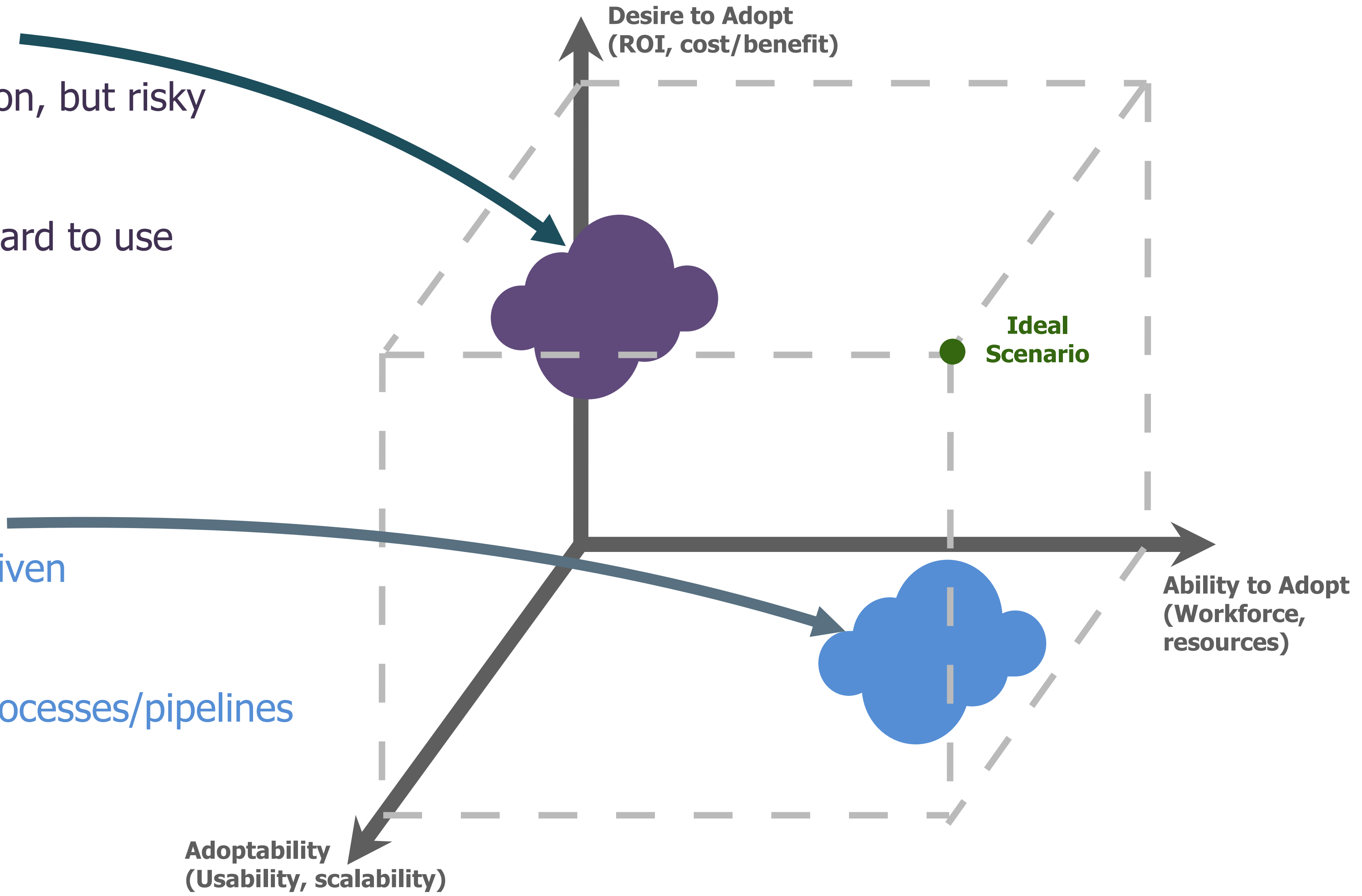

GAO
United States Government Accountability Office
Report to Congressional Committees
March 2021
WEAPON SYSTEMS CYBERSECURITY

In operational testing, DOD **routinely found mission-critical cybersecurity vulnerabilities** in systems under development. Using relatively simple tools and techniques, testers were able to **take control of systems and largely operate undetected**, due in part to basic issues such as poor password management and unencrypted communications. In addition, due to limitations in the extent and sophistication of testing, **DOD was likely aware of only a fraction of the total vulnerabilities in its weapon systems**.

Today: Workforce, resource, and tool challenges hamper Defense Industrial Base (DIB) adoption of formal methods

DoD contractors:
- High need to adopt for mission, but risky
- Workforce is hard to recruit
- Limited resources
- Tools are often fragile, and hard to use
- Tools don't fit within existing processes/pipelines

Big-name tech companies:
- Adoption is typically profit driven
- Highly skilled workforce
- Vast resources
- Have control over existing processes/pipelines

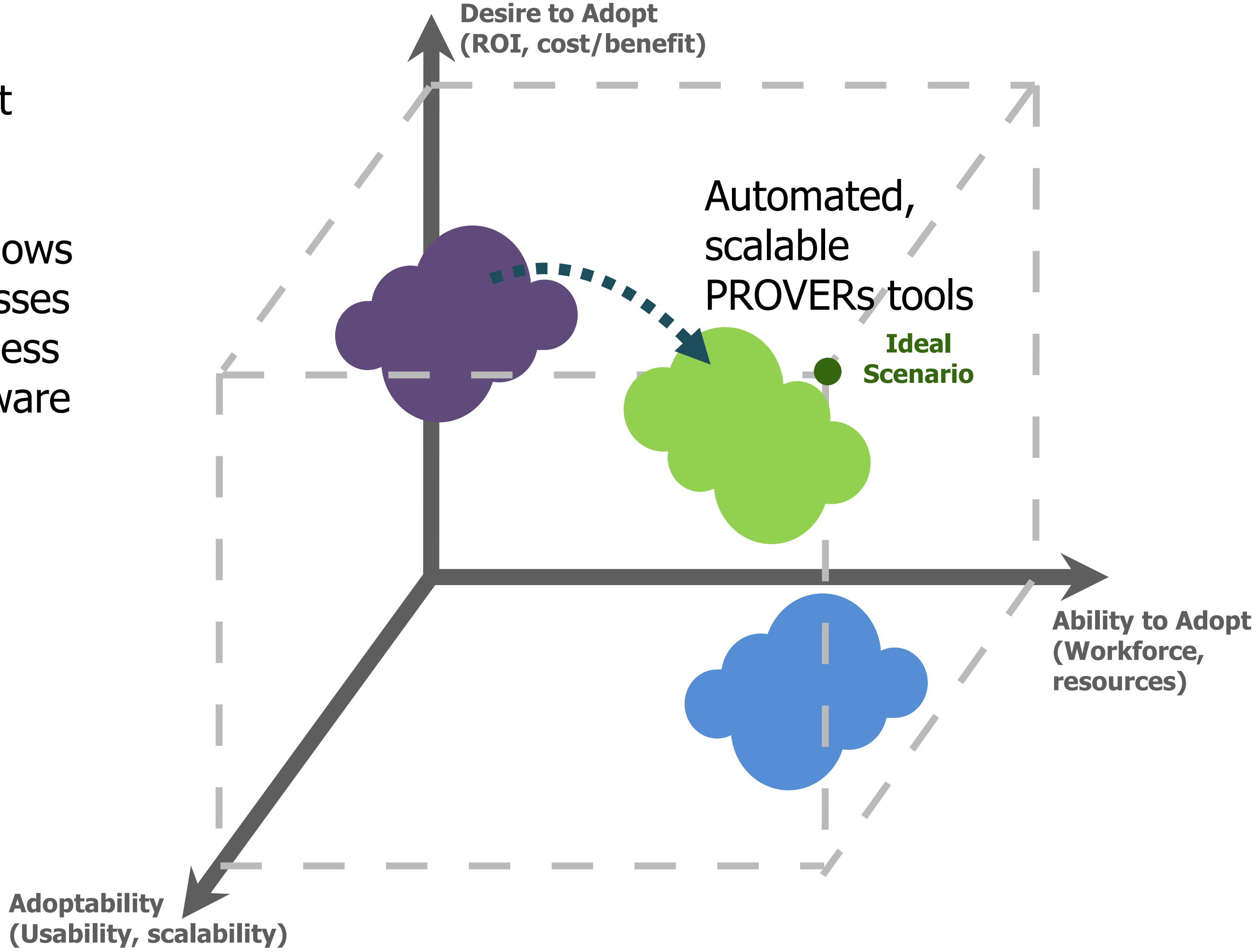


Source: I2O ISAT presentation, Addressing the Untrustworthiness of Software Supply Chains, November 7th, 2022

Tomorrow: PROVERS tools dramatically increase DIB adoption

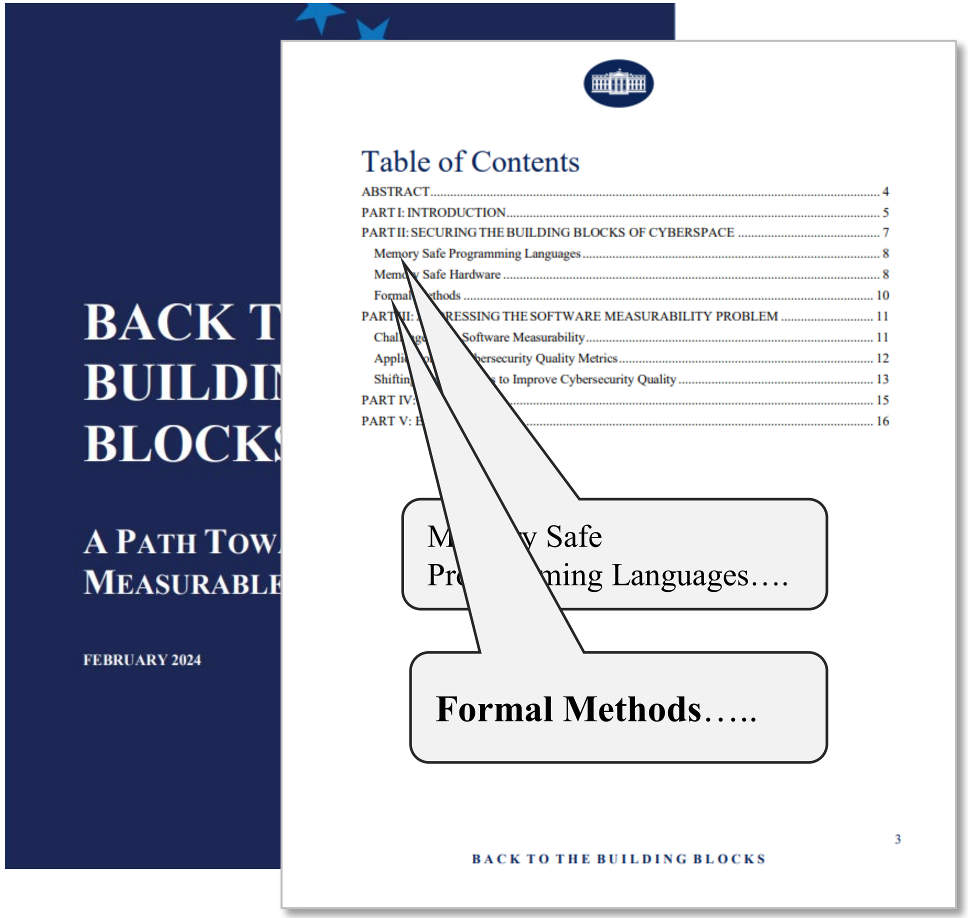
PROVERS tools:

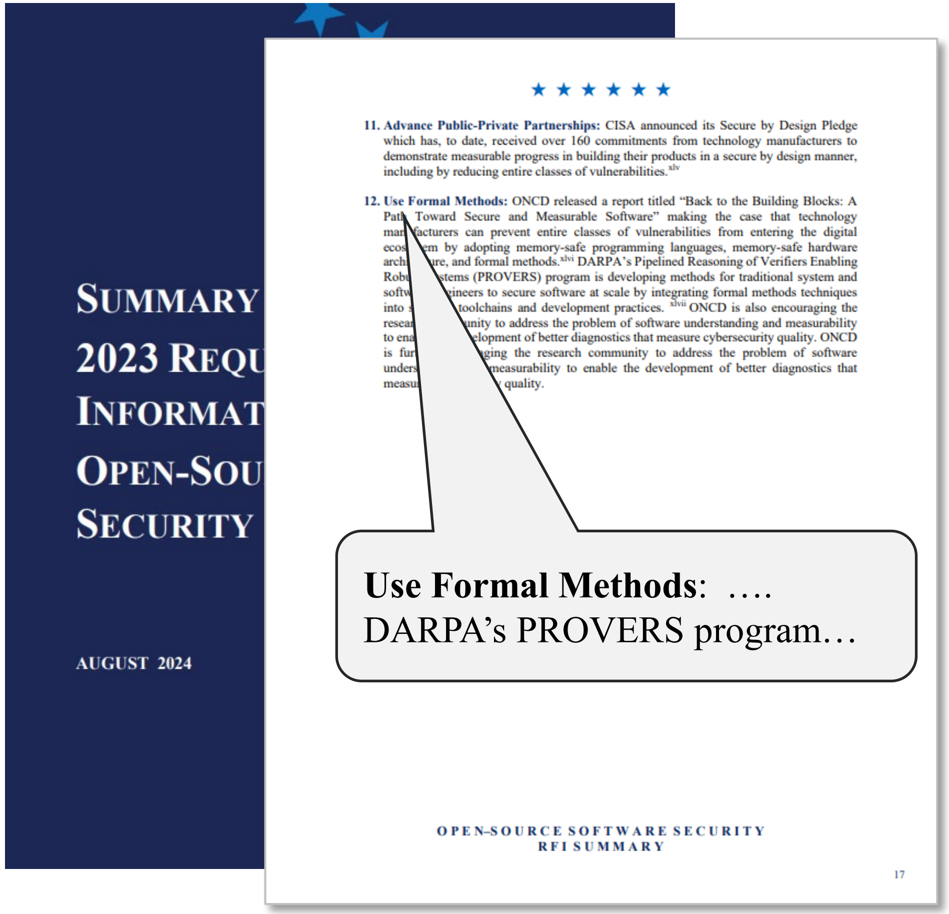
- Provide scalable automation targeted at traditional software developers (DIB workforce)
- Integrate into standard software workflows
- Enable incremental maintenance processes
- Refined via a continuous feedback process
- Create demonstrably more secure software based on red-team analysis



Source: I2O ISAT presentation, Addressing the Untrustworthiness of Software Supply Chains, November 7th, 2022 (modified)

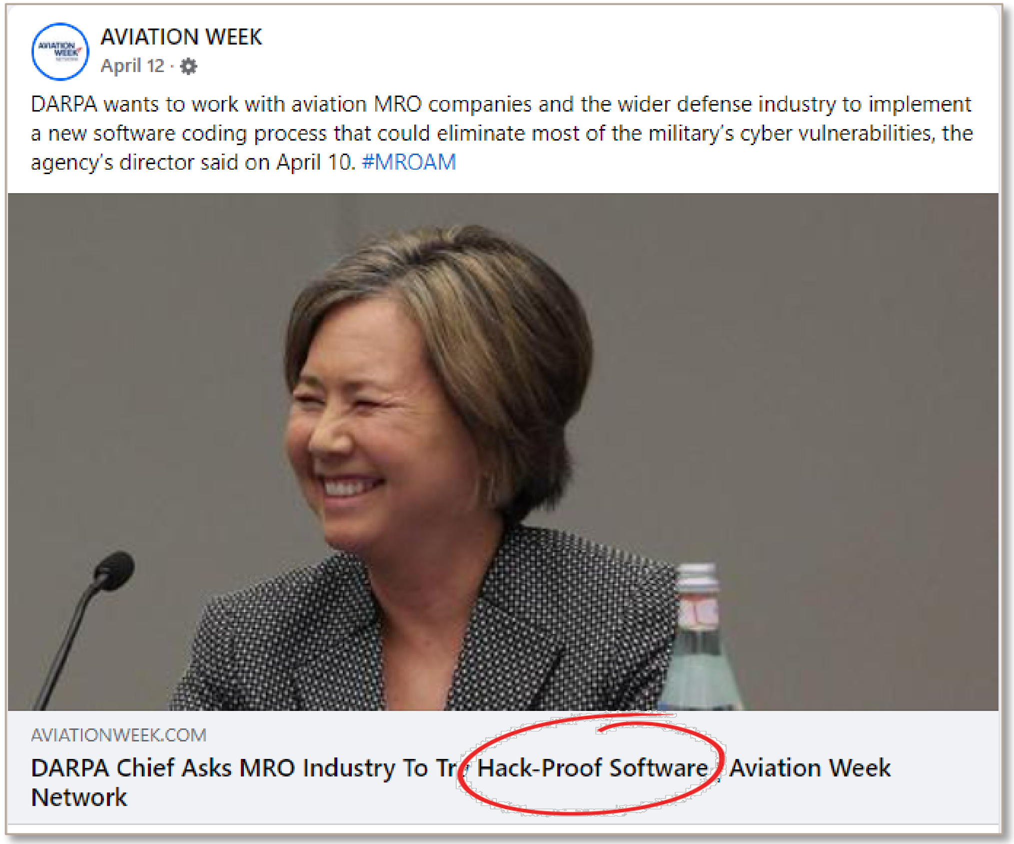
FORMAL METHODS IN THE NEWS





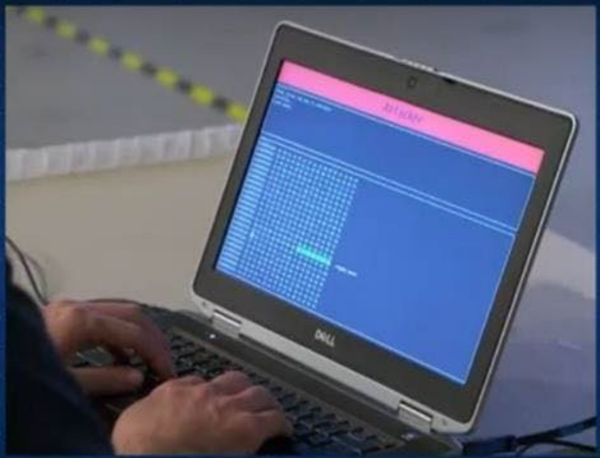
FORMAL METHODS IN THE NEWS

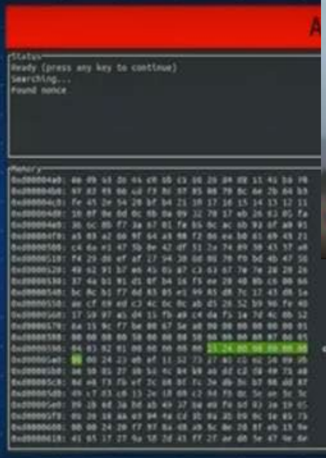
- A series of DARPA programs since 2016 have demonstrated the reliability of the “formal methods” coding approach on quadcopters and other uncrewed aircraft systems (UAS) such as the Boeing Unmanned Little Bird, **DARPA Director Stefanie Tompkins** said in a keynote speech at the Military Aircraft Logistics and Maintenance Symposium here at Aviation Week's MRO Americas conference.
- The agency also has been working on creating tools and training to implement the mathematical formal methods approach widely across government and industry.
- “We are at the point now where we have convinced ourselves that enough of this is ready for real-world use,” Tompkins said.



DARPA Cyber Resilience Strategy is closely aligned with PROVERS

HIGH ASSURANCE CYBER MILITARY SYSTEMS (HACMS)





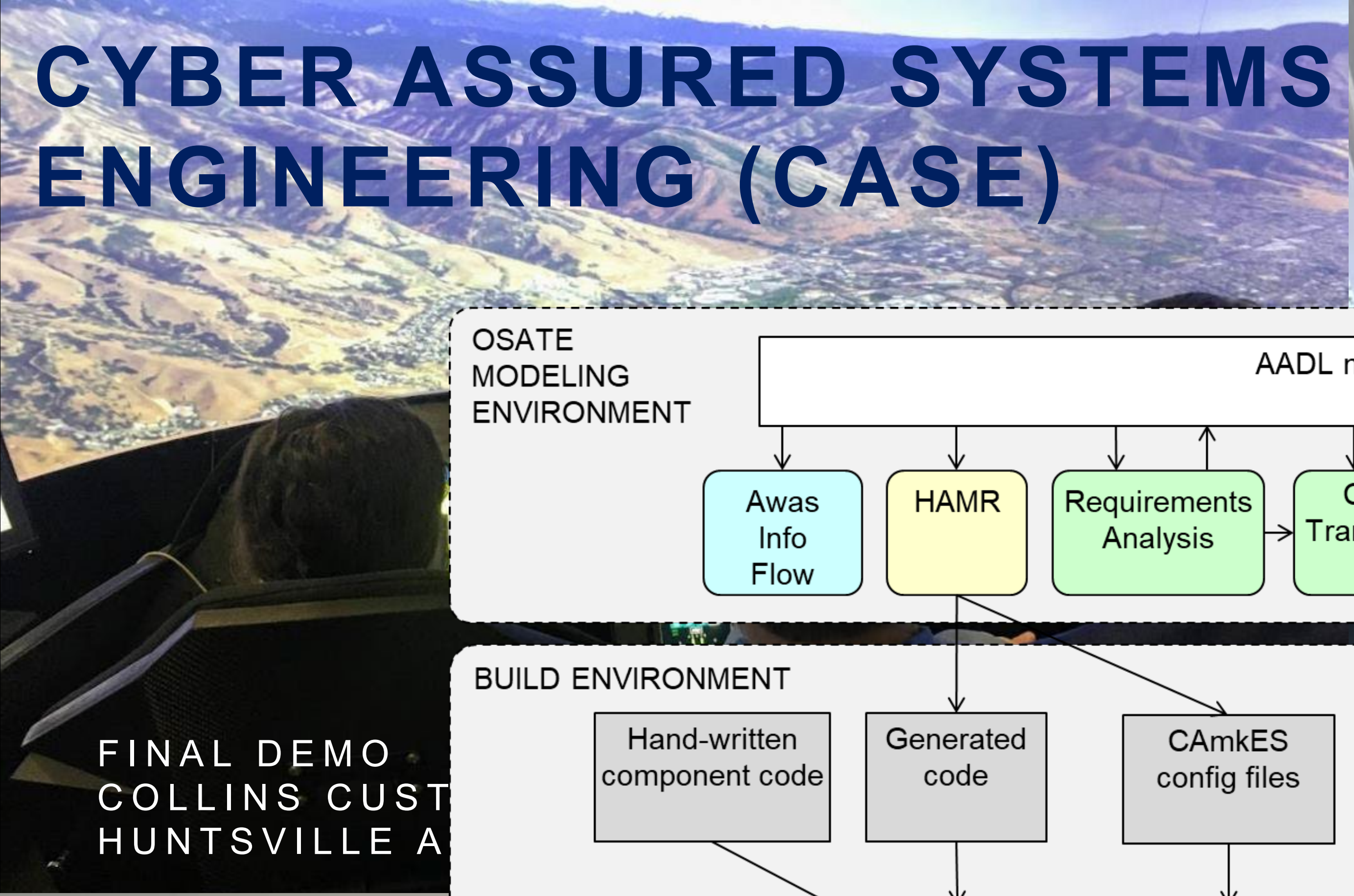
DEFCON AEROSPACE VILLAGE

9:59 / 25:07

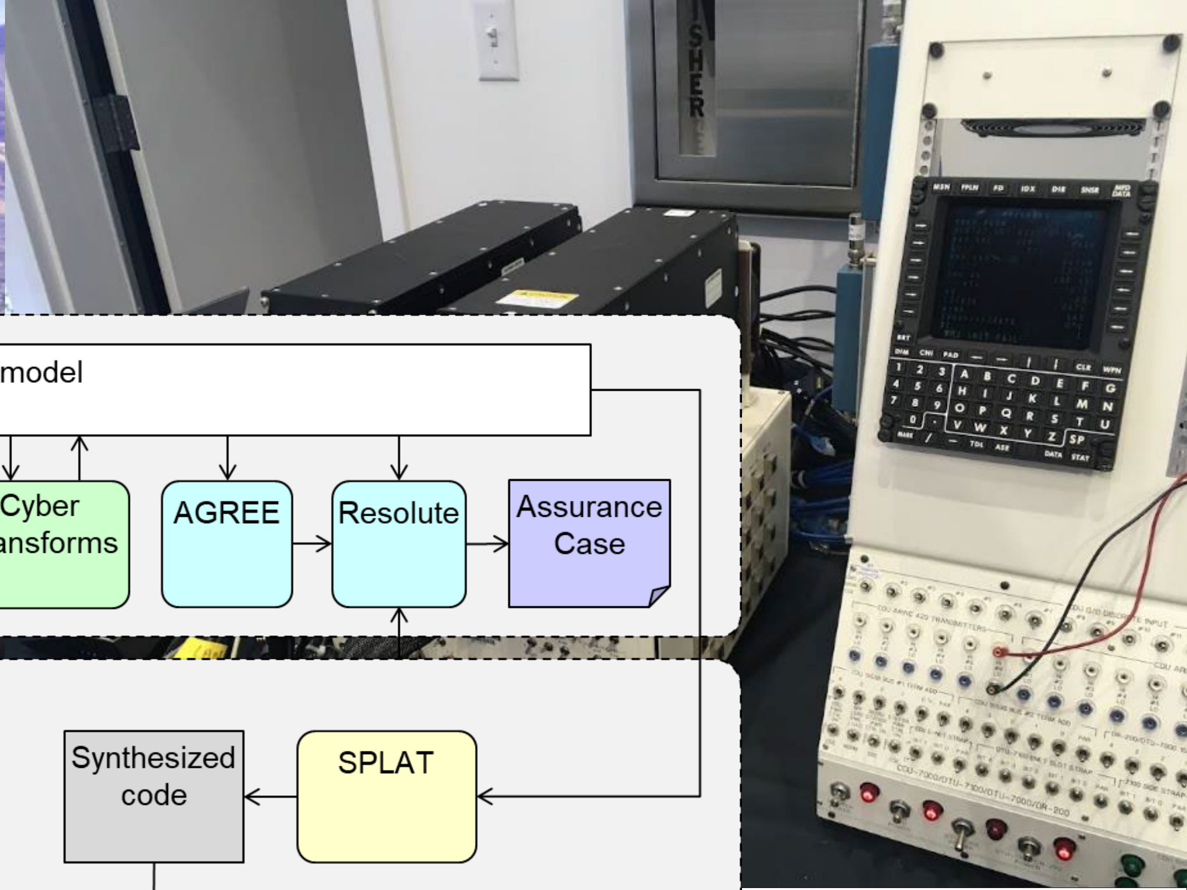
Loonwerks.com/projects/hacms

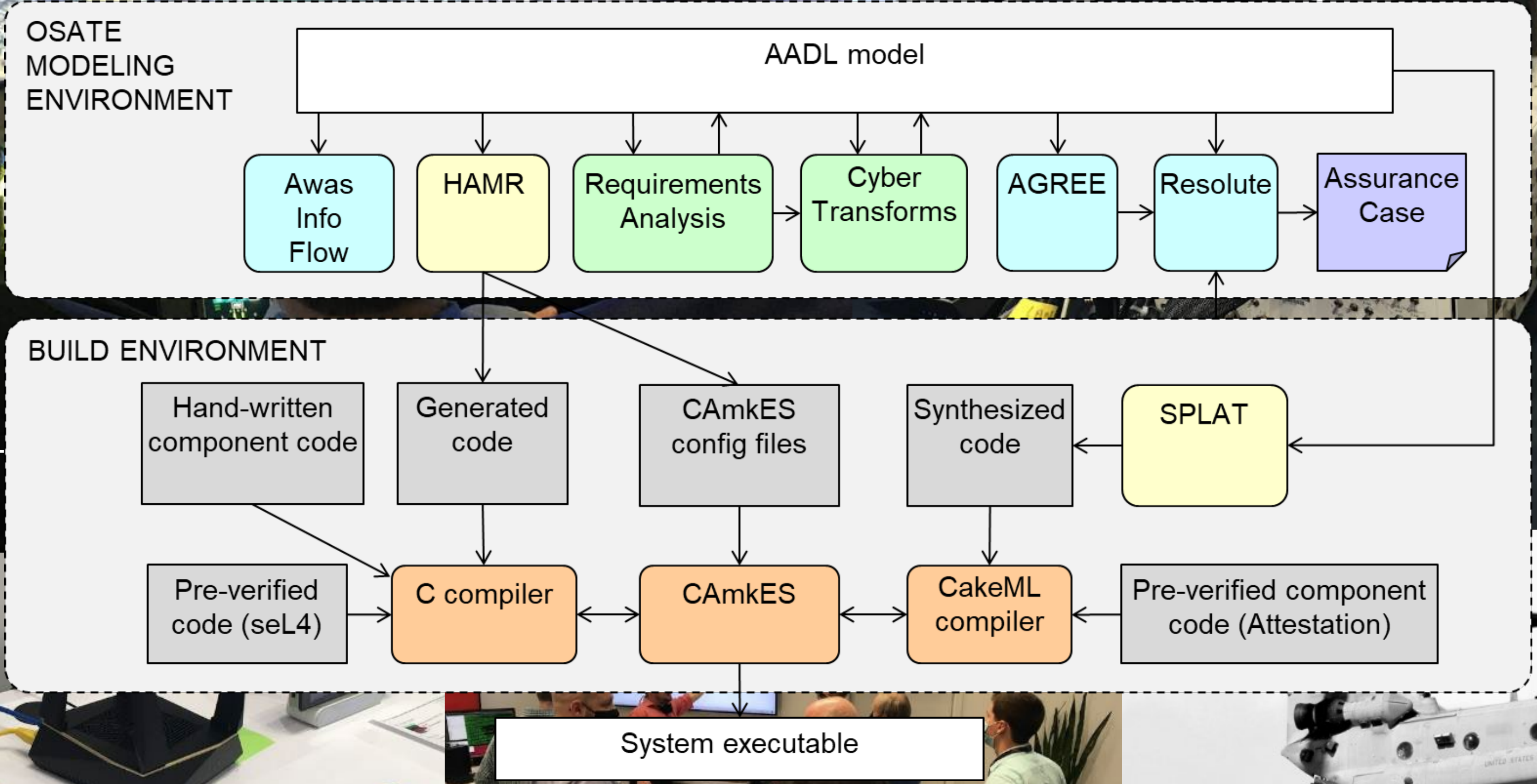
Collins Aerospace

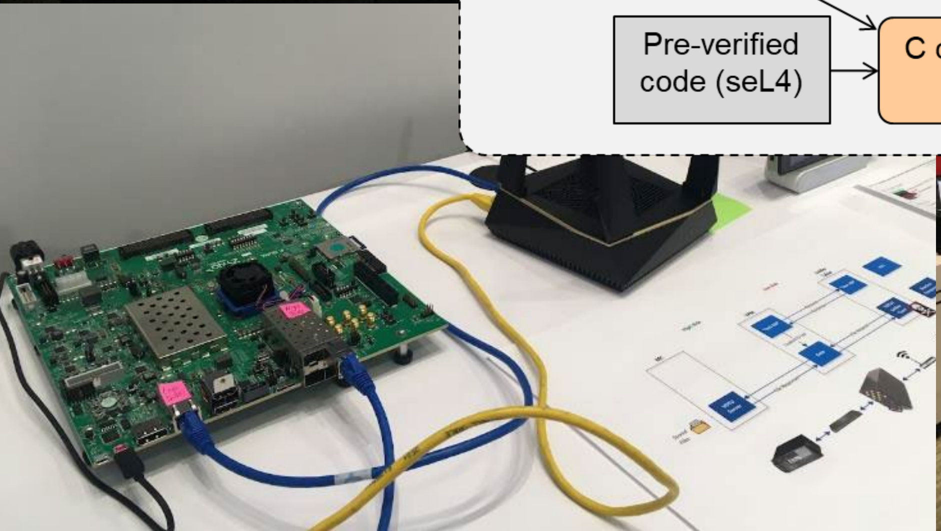
CYBER ASSURED SYSTEMS ENGINEERING (CASE)

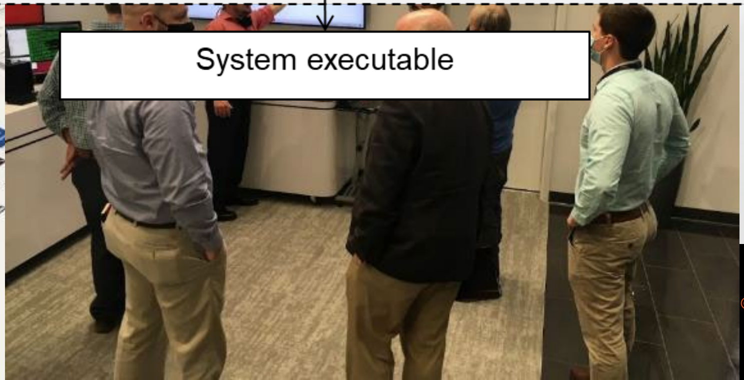


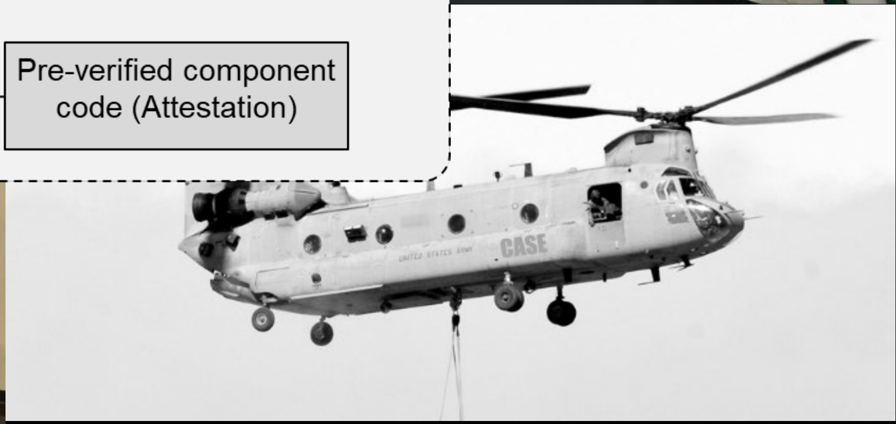
FINAL DEMO
COLLINS CUST
HUNTSVILLE A











INSPECTA : HIGHLIGHTS

- Our workflow and tools will address the **entire software development stack** from requirements and system models, to component source code, through build and deployment on the seL4 secure microkernel, linked by formal verification at each level.
- We will achieve scalability for complex defense systems through **compositional reasoning** at the system level and automated analysis of components based on powerful, cloud-based solvers.
- We will achieve the **highest levels of assurance** by building upon the best available technologies and leveraging our experience from recent research programs as a starting point.
- Our tools will be **integrated with current Collins workflow** automation processes and applied to defense and aerospace products currently under development to demonstrate their usability, practicality, and effectiveness.
- **Formal verification will be made accessible to non-formal methods experts** through automated analysis with streamlined user feedback and generalized proofs that are robust to changes, augmented by automated repair tools.
- Our **framework is adaptable and extendable** to allow incorporation of results from other researchers, including other specification languages, other source code languages, and other operating system targets.
- Our **access to critical defense and aerospace products** in both commercial and military domains served by Collins will serve as the basis for compelling demonstrations of INSPECTA technologies.

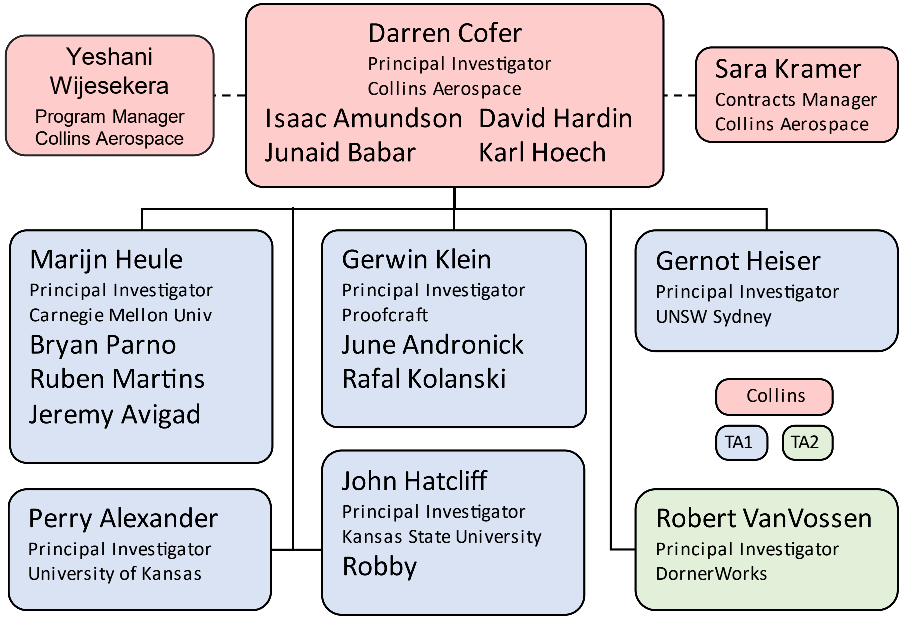
INSPECTA TEAM

Technology Area 1 – Proof Engineering

- **Collins**: System requirements, model-based compositional reasoning, workflow integration and assurance gathering, user feedback and measurement
- **CMU**: Software component analysis, scalable SAT/SMT analysis, Rust software verification environment
- **KS State Univ**: Model-based build framework, formal model of AADL, code generation for seL4 and other OS
- **Proofcraft**: Robust and generalized proofs, seL4 verification
- **UNSW Sydney**: Push-button verification of seL4 microkit, seL4 OS components
- **Univ of KS**: Component software synthesis, AI-Enhanced proof repair, lifecycle attestation for workflow

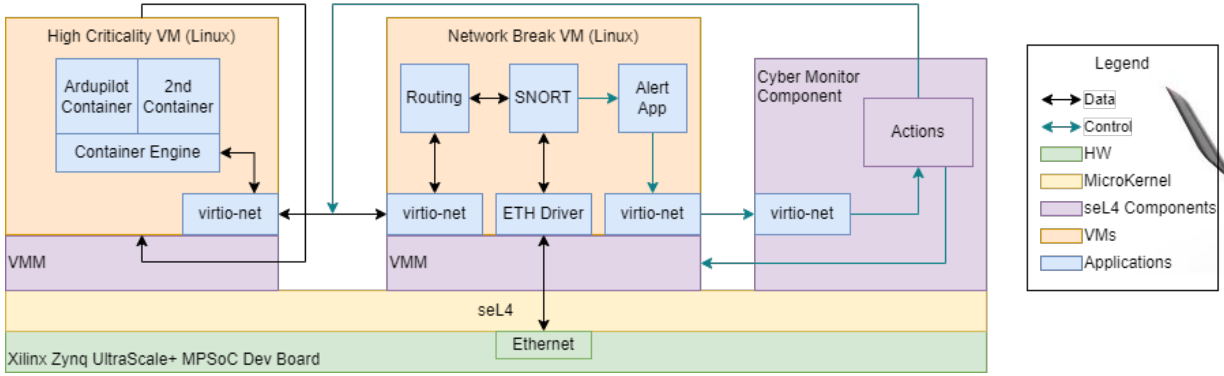
Technology Area 2 – Platform Development

- **Collins**: Provide platforms for demonstration of TA1 tools, requirements changes to evaluate tool effectiveness, including US Army (Air) Launched Effects tube-launched small UAV
- **DornerWorks**: Develop open demonstration platform based on Army SBIR with Collins, UAS mission software running on seL4



TA2 : OPEN DEMO PLATFORM

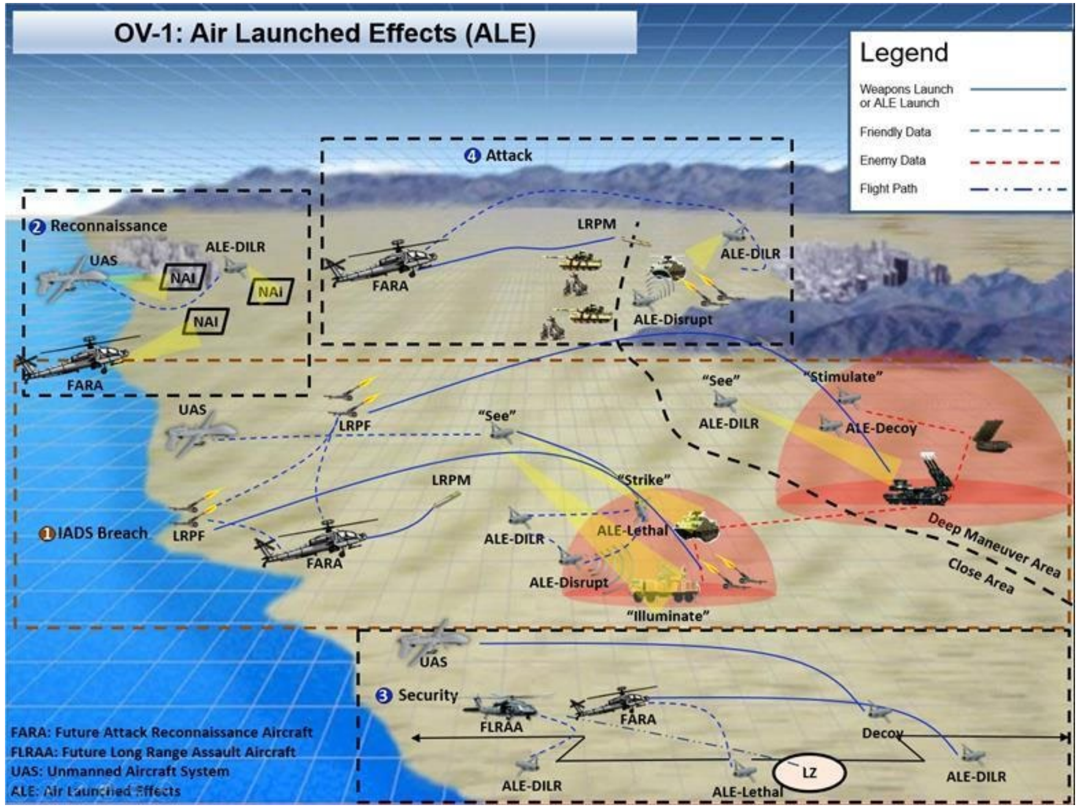
- Developed and supported by DornerWorks
- Unrestricted UAV mission software, system model with formal properties, multiple VMs, Rust software components, seL4 kernel
- Hardware
  - Xilinx Zynq UltraScale+ MPSoC-based development board (equivalent to RapidEdge)
  - Raspberry Pi Compute Module 4 (used in RapidEdge surrogate demo vehicle)



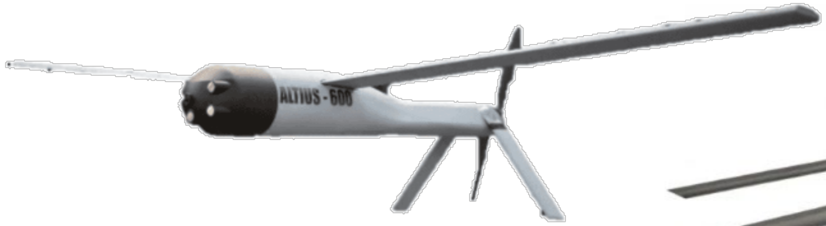
TA2 : TRANSITION PLATFORM

U.S. Army (Air) Launched Effects (LE)

- Family of Systems (FoS) consisting of a tube-launched air vehicle, payload(s), mission system applications, and associated support equipment to autonomously or semi-autonomously deliver effects as a single agent or as a member of a team
- LE extends tactical and operational reach and lethality of manned assets, allowing them to remain outside of the range of enemy sensors and weapon systems while delivering kinetic and non-kinetic, lethal and non-lethal mission effects
- Relatively low-cost systems, attritable or optionally recoverable, allow rapid integration of new technologies.



https://www.flightglobal.com/military-uavs/us-army-outlines-recon-and-electronic-warfare-missions-for-air-launched-effects/139780.article



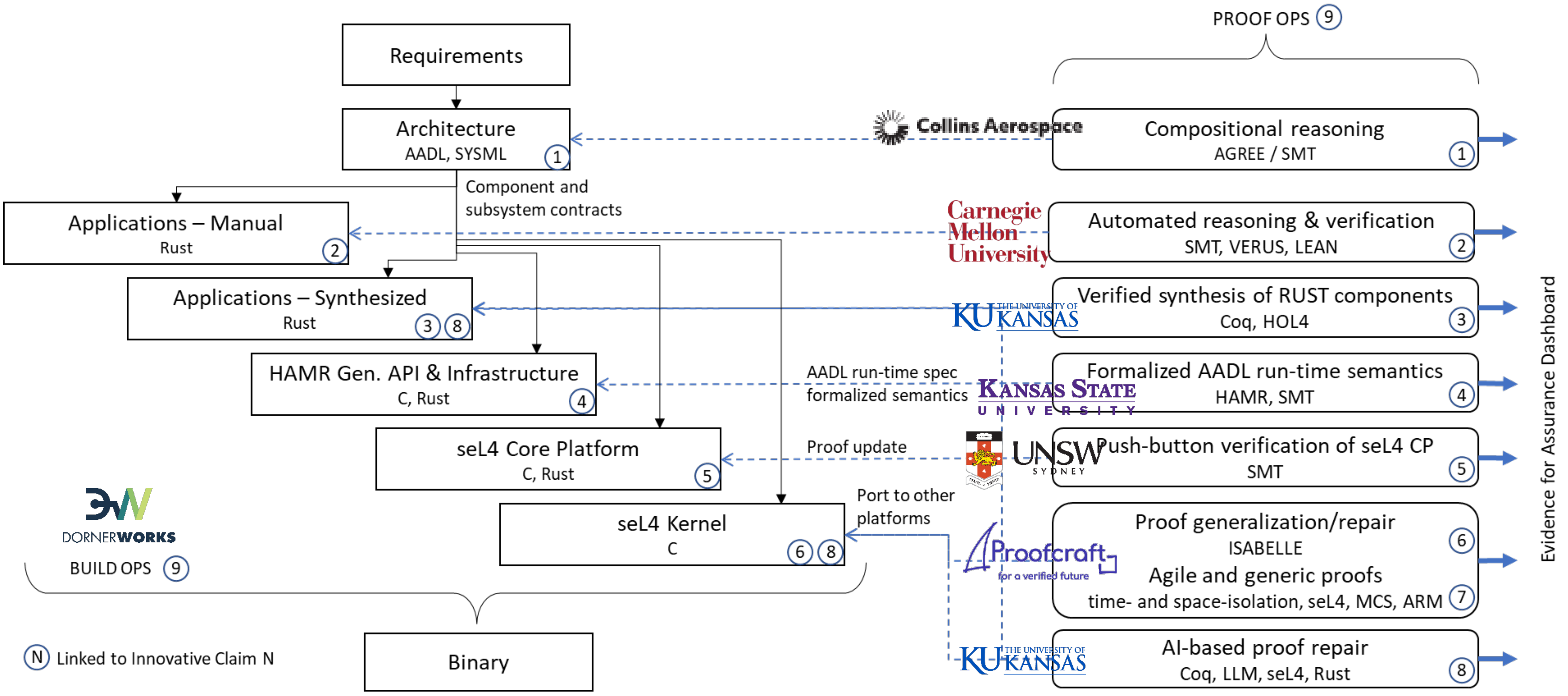
(AIR) LAUNCHED EFFECTS

https://youtu.be/SpnGE2CCx2w

https://youtu.be/OosofUsbaRc

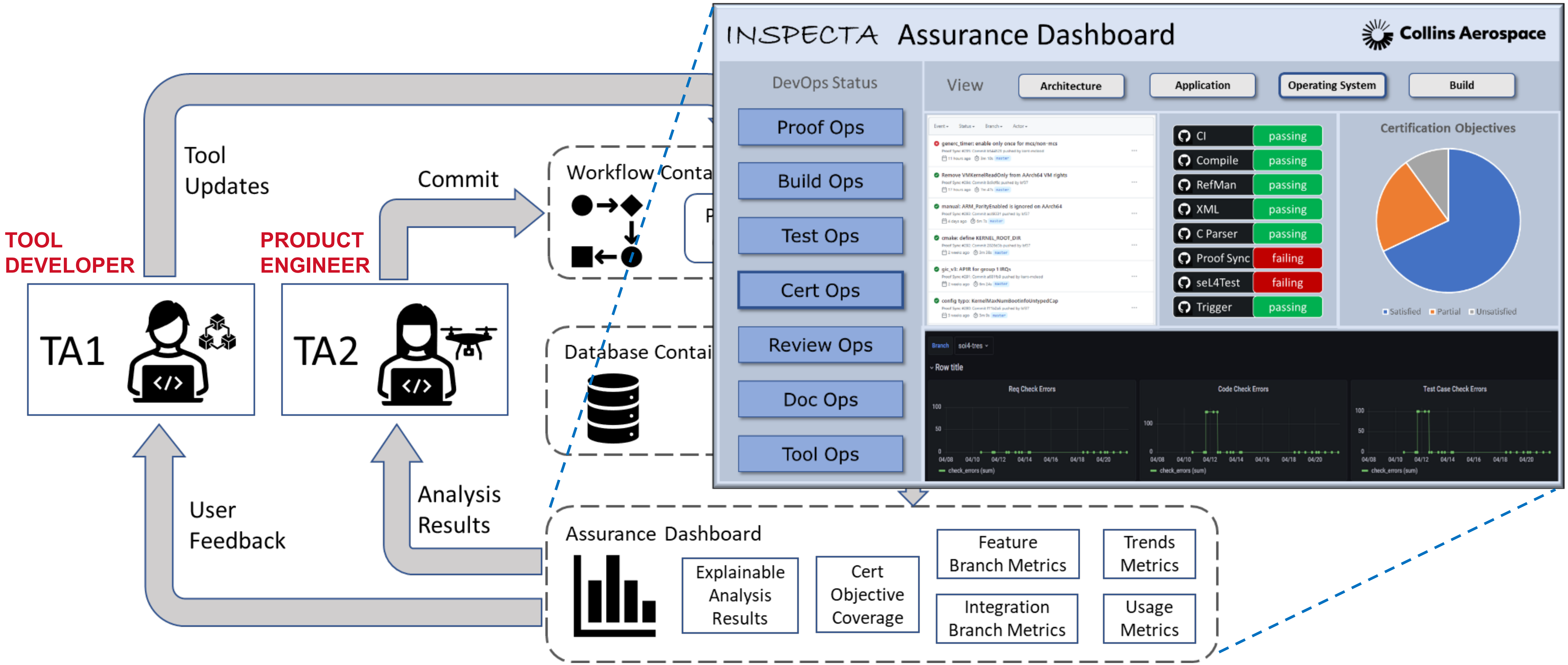
Both air and ground launched options supporting a wide variety of missions

INSPECTA : TA1 TOOLS



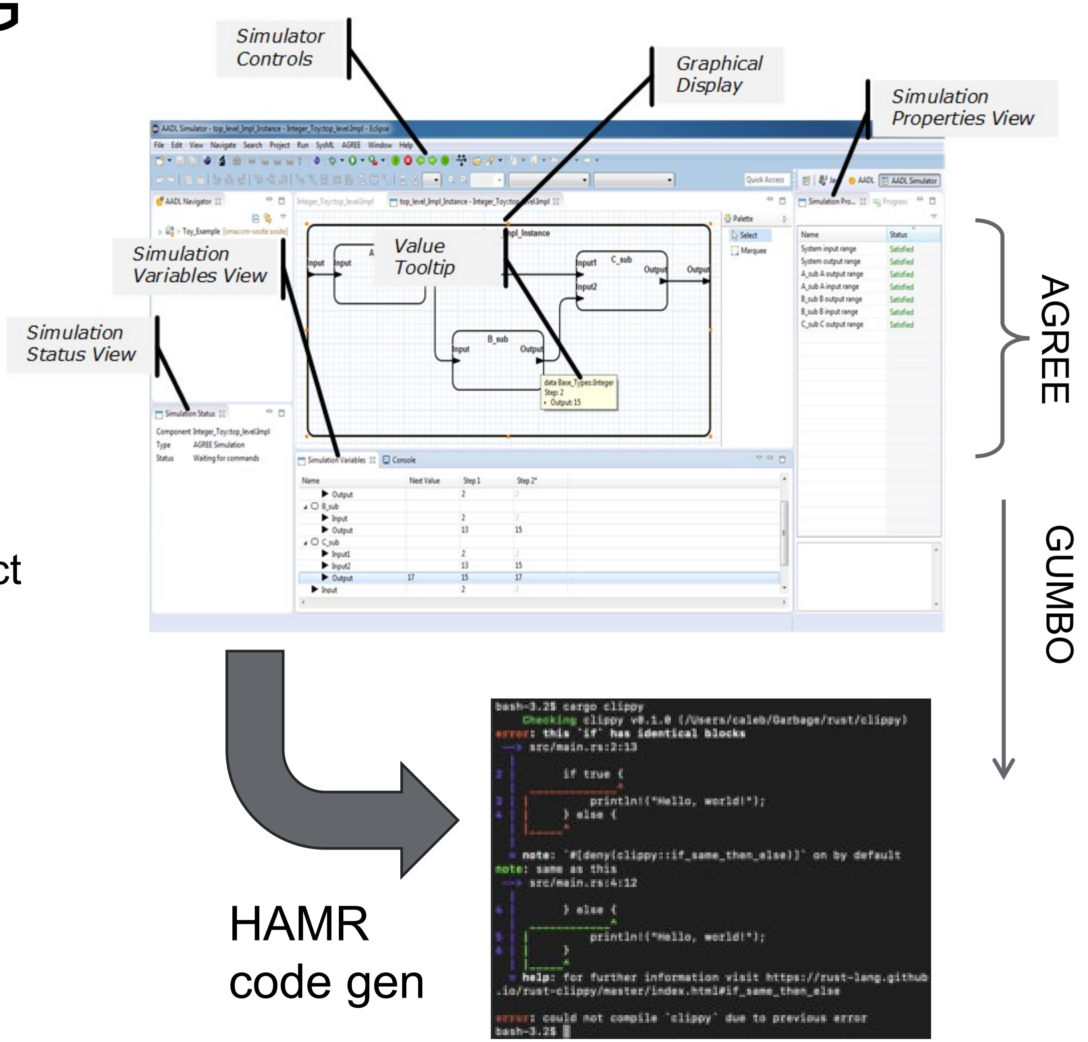
Collins Aerospace

INSPECTA : TA1 WORKFLOW



COLLINS : ARCHITECTURE MODELING AND COMPOSITIONAL REASONING

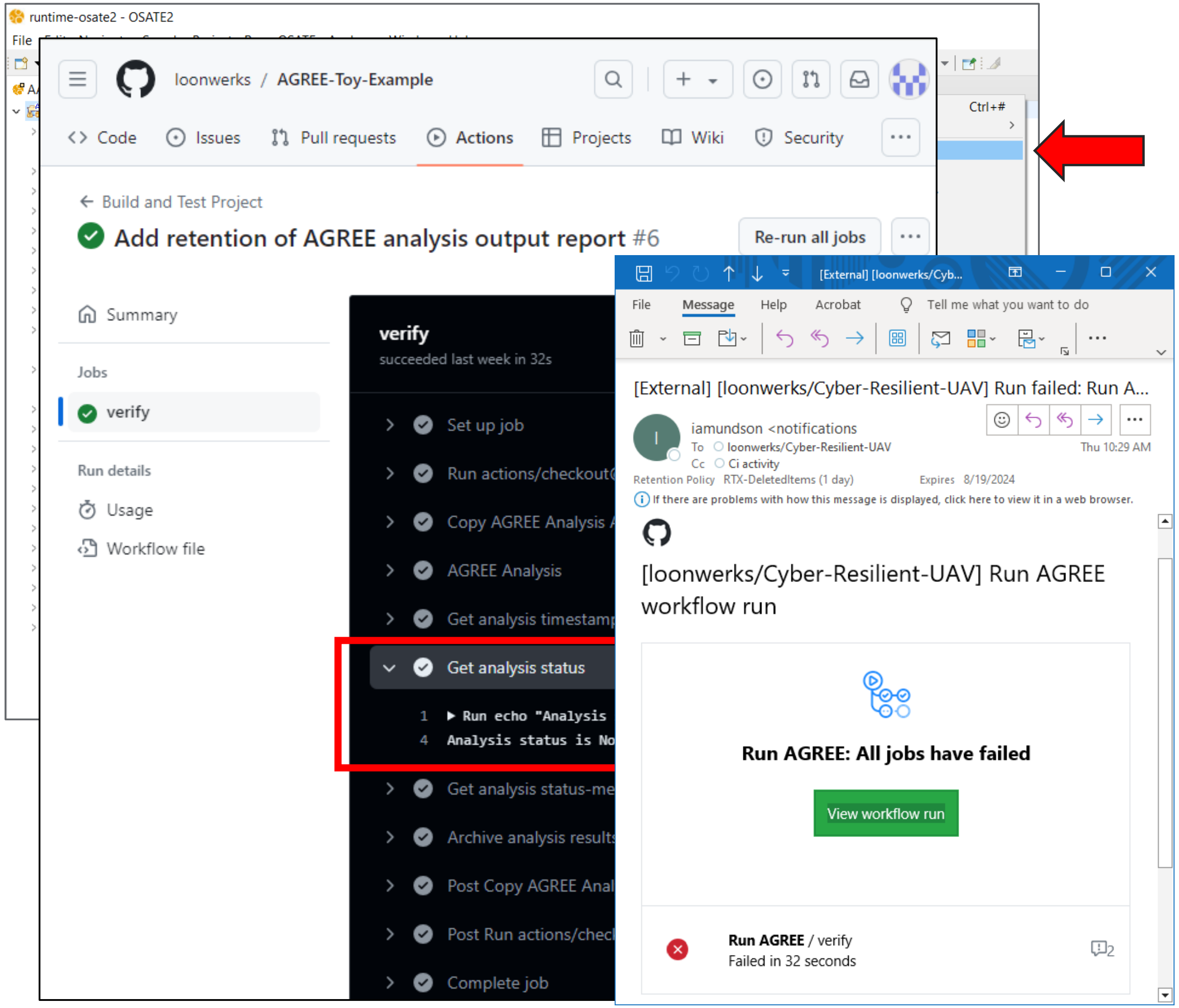
- Develop language abstractions to simplify contract specification in AGREE
- Enhance graphical interface for AGREE that enables engineers to walk through generated counterexamples more intuitively
- Establish traceability to proof obligations at the source code level
  - Achieved through tighter coupling with KSU's GUMBO contract language
- Integrate AGREE into DevOps workflow
- Compositional reasoning for SysMLv2
  - OMG Real-Time Embedded Safety Critical Systems working group



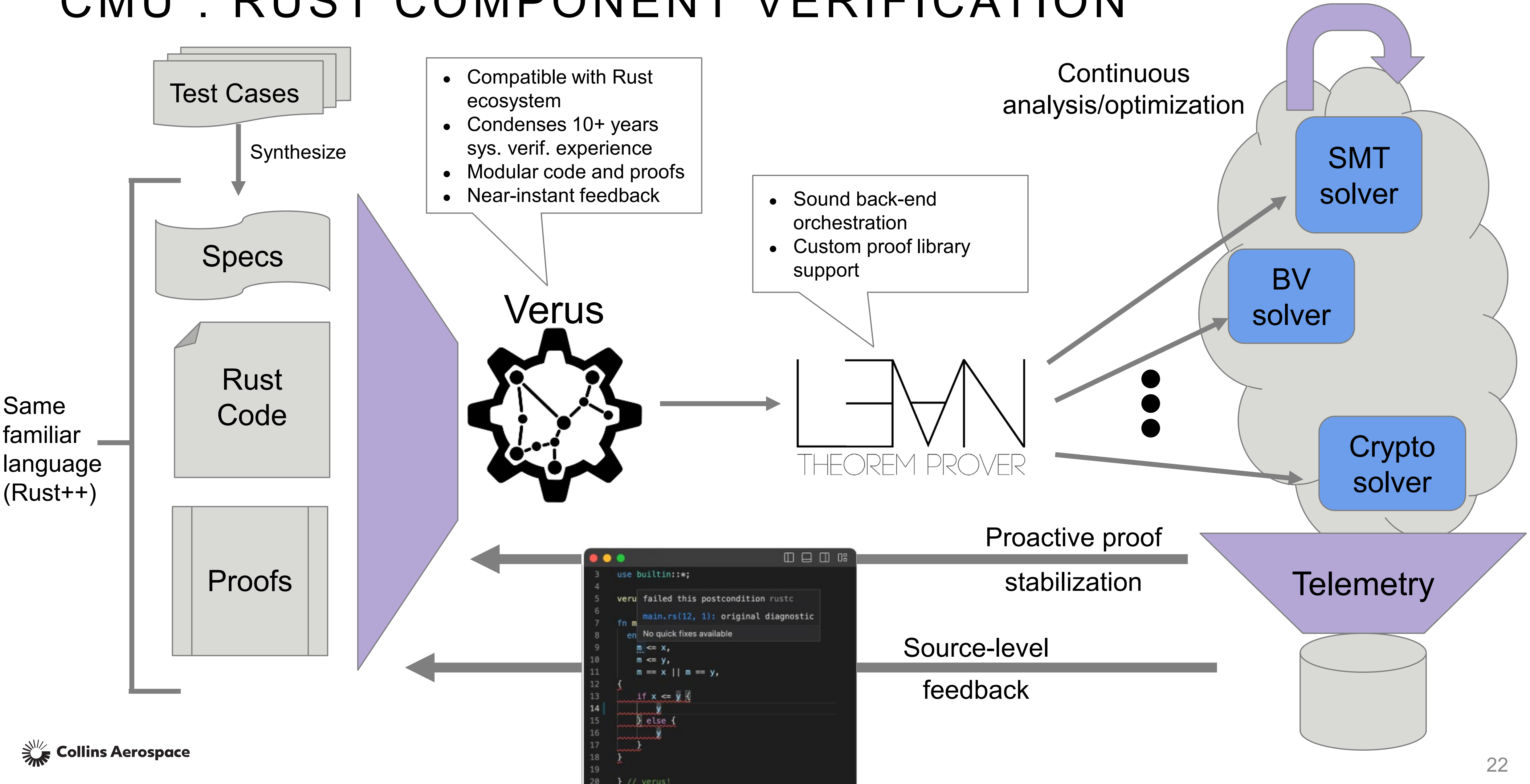
Collins Aerospace

COLLINS : INTEGRATION INTO CI/CD WORKFLOW

- Automatically formally verify architecture models upon check-in
- Workflow automation handled by Github actions
- Users are alerted to verification failures via email
- AGREE output in json format
- Requires command line version of AGREE
  - Available at github.com/loonwerks/agree



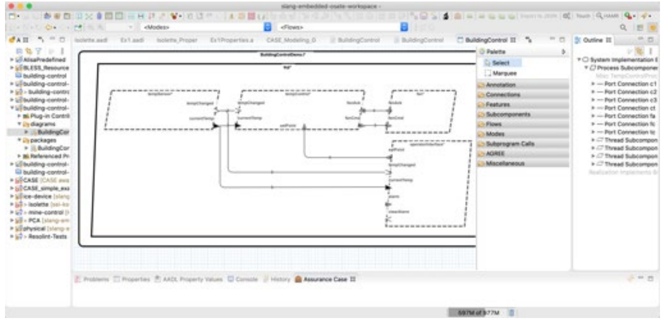
CMU : RUST COMPONENT VERIFICATION



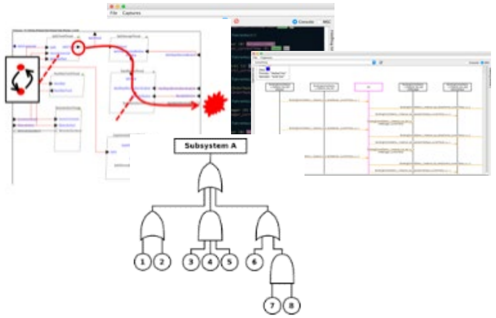
KANSAS STATE UNIV (KSU) : HAMR

HAMR – tool chain for [H]igh [A]ssurance [M]odeling and [R]apid engineering for embedded systems (developed by Kansas State University and Galois)

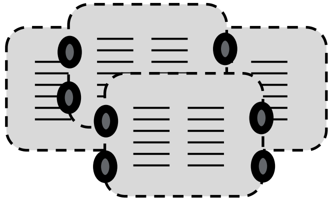
Modeling, analysis, and verification in the **AADL** modeling language



Leveraging analyses from AADL community



Component development and verification in multiple languages



- C
- Slang (developed at Kansas State)
    - high integrity subset of Scala
    - contract verification framework
    - translates to C

Deployments aligned with AADL run-time on multiple platforms

JVM Deployment

JVM (Java Virtual Machine)

Linux Deployment

seL4 Deployment

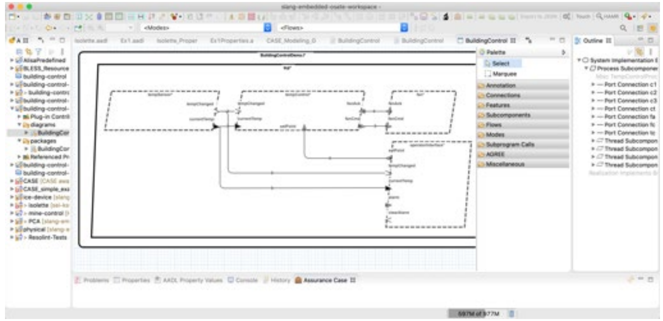
seL4 Security. Performance. Proof.

verified micro-kernel

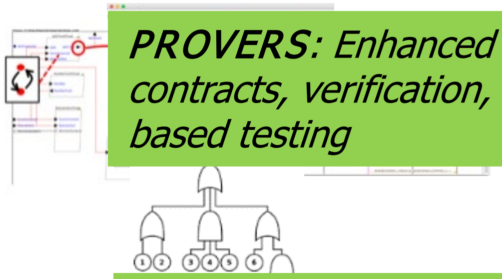
KANSAS STATE UNIV (KSU) : HAMR

HAMR – tool chain for [H]igh [A]ssurance [M]odeling and [R]apid engineering for embedded systems (developed by Kansas State University and Galois)

Modeling, analysis, and verification in the **AADL** modeling language



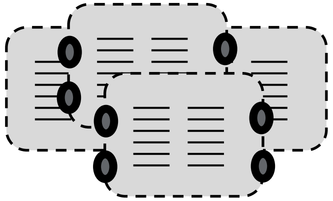
Levera
commu



PROVERS: Add SysMLv2 prototype

PROVERS: Enhanced support for contracts, verification, property-based testing

Component development and verification in multiple languages



- C
- Slang (developed
  - high integrit
  - contract verification framework
  - translates to C

PROVERS: Add code- and contract-generation, and property-based testing for Rust

Deployments aligned with AADL run-time on multiple platforms

JVM Deployment

JVM (Java Virtual Machine)

Linux Deployment

seL4 Deploymen

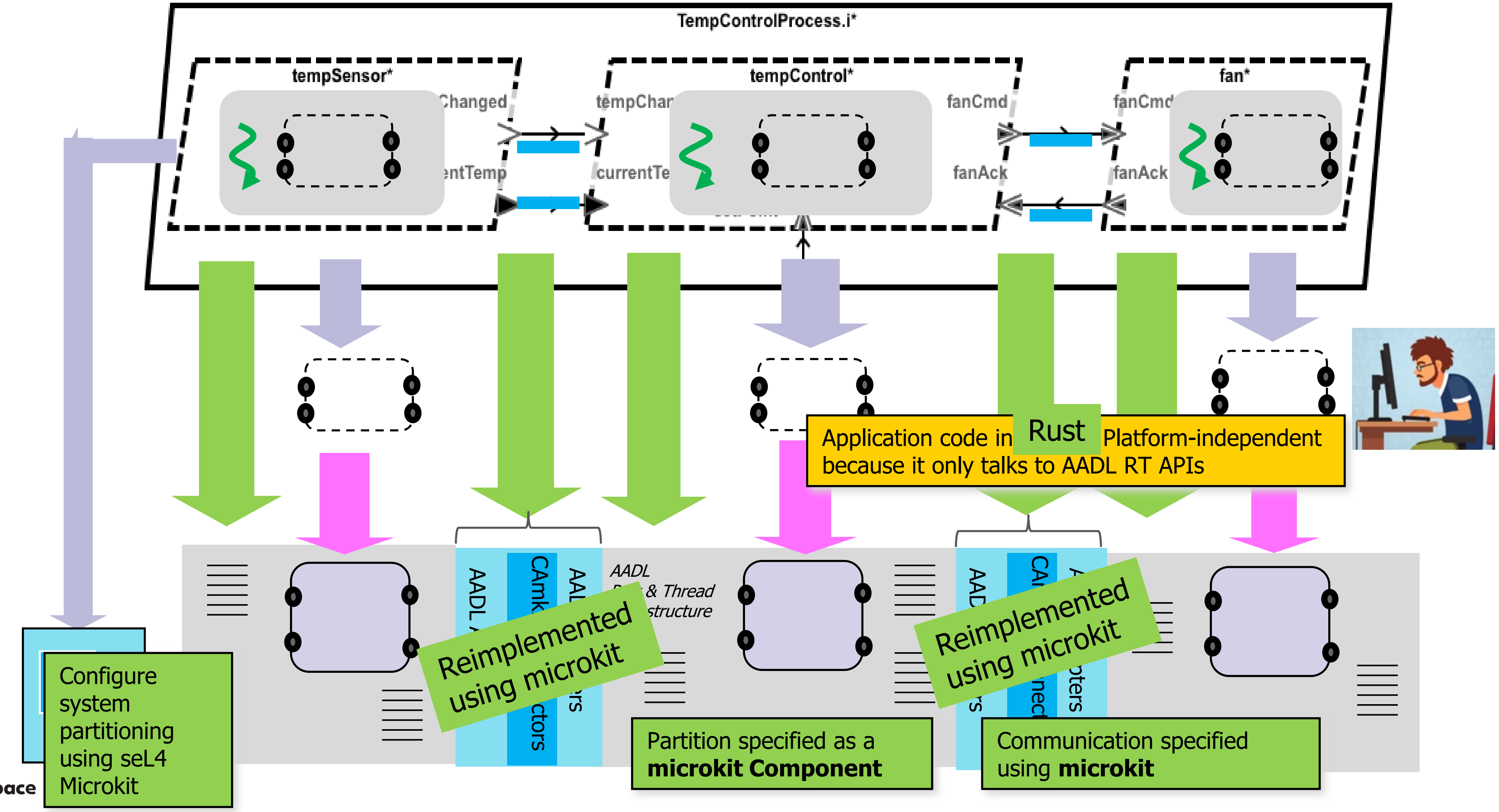
PROVERS: Retarget to seL4 micro-kit (Core Platform)

seL4 Security. Performance. Proof.

verified micro-kernel

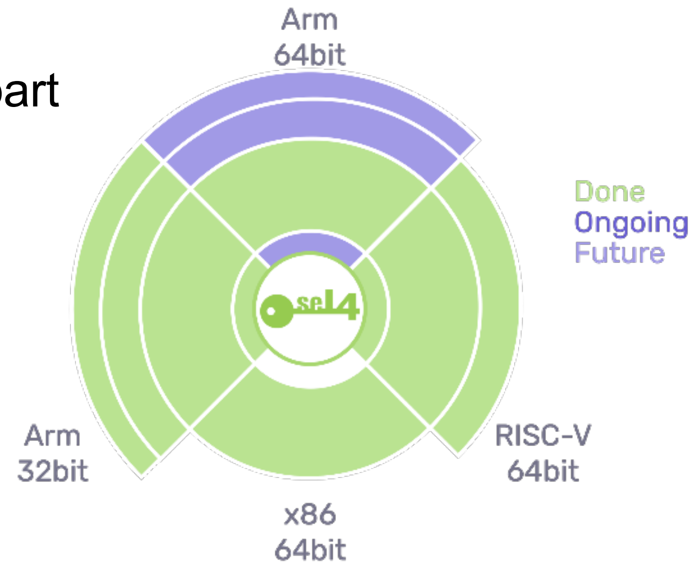
KSU : HAMR CODE GENERATION

HAMR instantiation for Rust based development on **seL4 microkernel** using seL4 microkit



PROOFCRAFT : SEL4 KERNEL PROOF GENERALIZATION AND REPAIR

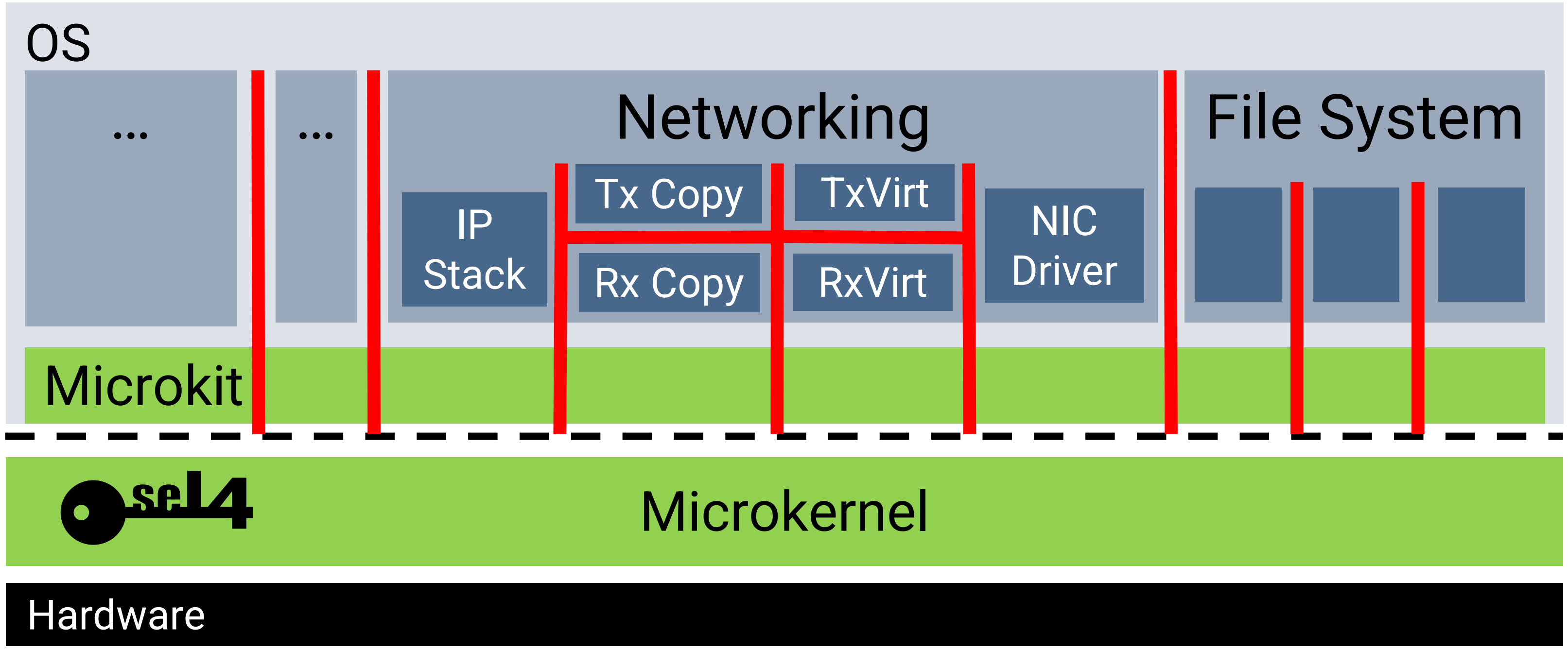
- Goal: make seL4 proofs less dependent on experts for maintenance and extensions
  - Automated Verification for Platform Ports
    - → Proof parametrization, proved once against sufficient conditions
    - → For each new platform: automatic extraction of configuration parameters & check against conditions
  - More Agile and Generic Proofs
    - → Split generic architecture-independent part from architecture-dependent part
    - → Extend verification to latest major feature: MCS seL4
- Impact: scalable access to formal methods
    - → Reduced cost and reliance on experts for maintenance and extensions
    - → Increased assurance robustness against anticipated change
    - → Increased features for verified foundation



seL4 verified on more platforms, with more features, for less cost and less expertise

UNSW SYDNEY : LIONS OPERATING SYSTEM & COMPONENT VERIFICATION

- Lions OS: new seL4-based OS developed from scratch at UNSW
- Highly-componentized, verification-friendly, yet high-performance
- Simplicity & adaptability by use-case-specific, swappable policies



UNIV OF KANSAS (KU) : PROOF EVIDENCE, AI-BASED PROOF REPAIR, COMPONENT SYNTHESIS

- ML-Enhanced Proof Repair
  - Maintain evidence over design, requirements and environmental changes
  - Update and replay proofs, retake measurements, replay testing
- Evidence Protocols
  - Update and generalize Copland attestation protocols for general-purpose evidence gathering
  - Develop canonical techniques for parametric adaptation, refinement and abstraction, protocol synthesis
  - Reuse MAESTRO attestation environment for general evidence gathering
- Verified Synthesis
  - Enable working at the requirements level
  - Synthesis of Rust from requirements language retargeting Coq / CakeML synthesis from CASE

SUMMARY

- Workflow and tools address the entire software development stack
- Building upon the best available technologies and leveraging our experience from recent research programs as a starting point
- Integrate new formal methods tools with Collins workflow automation processes
- Apply tools to Collins Launched Effects mission computer to enhance cyber-resiliency and demonstrate usability, practicality, and effectiveness
- Formal verification will be made accessible to non-formal methods experts through automated analysis with streamlined user feedback and generalized proofs that are robust to changes, augmented by automated repair tools

END

https://loonwerks.com/projects/INSPECTA