CYBER ASSURED SYSTEMS ENGINEERING

AADL / ACVIP USER DAY
2 JUNE 2022

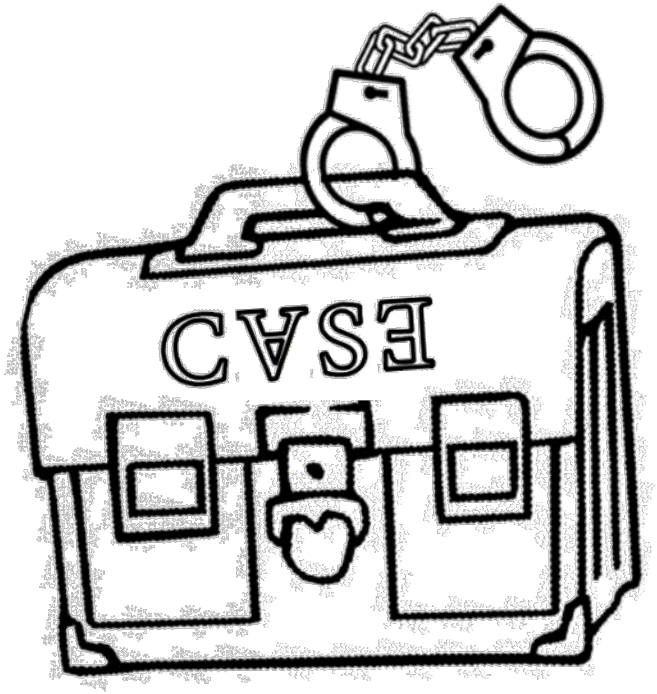
DARREN COFER / COLLINS
TODD CARPENTER / ADVENTIUM

Collins Aerospace

DISTRIBUTION STATEMENT A:Approved for public release; distribution unlimited.

OUTLINE

- CASE Overview & Applications
- BriefCASE: AADL Modeling & Analysis Tools for Cyber-Resilience (Darren)
- HAMR Code Generation Framework (Todd)



DARPA HACMS

2012-2017

Problem:

Cyber vulnerabilities are not isolated to traditional information processing platforms and infrastructure, but are also present in *embedded systems* (cyber-physical systems), including safety-critical systems

HACMS goal:

Use *formal methods* to build embedded systems that are resilient against cyber-attack because they can be *proven* not to have typical security vulnerabilities

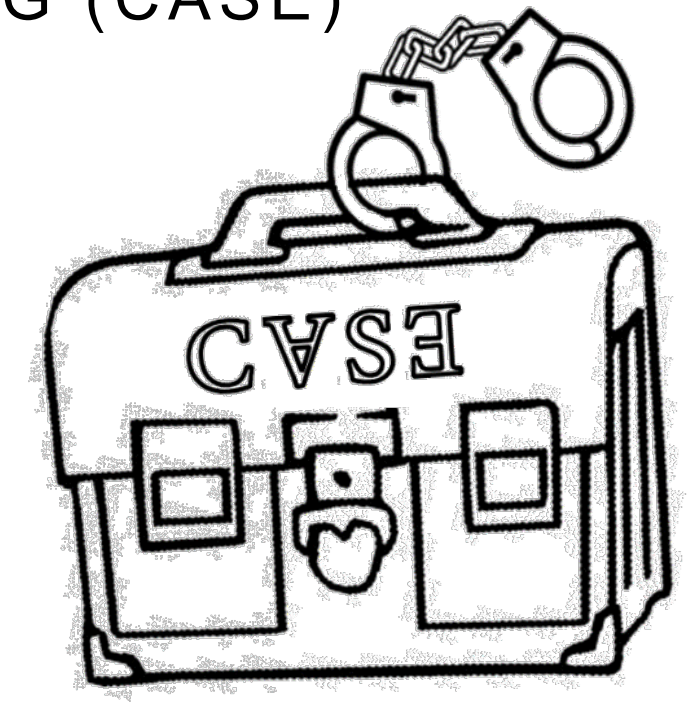
Formal Methods = Complete exploration of software/system design using mathematical analysis

Collins Aerospace | galois | seL4 | BOEING | University of Minnesota

CYBER ASSURED SYSTEMS ENGINEERING (CASE)

Develop model-based systems engineering tools and workflow to make the HACMS approach repeatable, scalable, more incremental

- **Design-in cyber-resiliency**
  - Automated architecture transforms for threat mitigation
  - High assurance components generated from specifications
  - Techniques to deal with legacy code (“cyber retrofit” using virtual machines)
- **Build what you model**
  - Build system directly from detailed, verified AADL model
  - Make the security guarantees of seL4 accessible to system developers
  - Ability to target different platforms to facilitate incremental debugging/development
- **Provide evidence**
  - Formal verification of functional and cyber-resiliency properties, information flow properties, component proofs
  - Code generation equivalence to model, seL4 build preserves properties
  - Integrate evidence as an assurance case demonstrating how/why requirements are satisfied



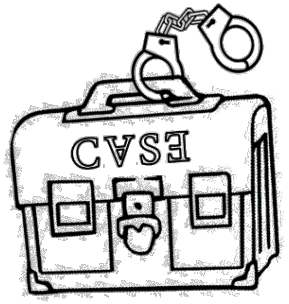
ACVIP

Collins Aerospace

CASE TEAM

TECHNICAL AREAS 2 & 5

- Collins Aerospace
  - Architectural transformations for cyber-resilience
  - Component synthesis and proofs
  - Formal analysis and assurance case
  - Tool integration
- University of New South Wales
  - seL4 formally verified secure microkernel for memory protection
  - Formally verified components (seL4, CakeML language)
- University of Kansas
  - Formally verified attestation for distributed computing platforms
- Adventium
  - Real-time scheduling
  - AADL modeling
- Kansas State University
  - Automatic code generation from architecture models with proof of equivalence
  - Information flow analysis



DARPA

Collins Aerospace

UNSW SYDNEY

seL4 FOUNDATION

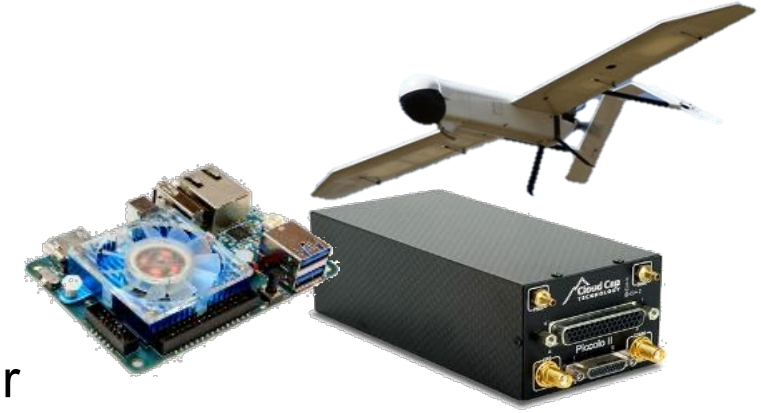
KU THE UNIVERSITY OF KANSAS

Adventium LABS

KANSAS STATE UNIVERSITY

PLATFORMS

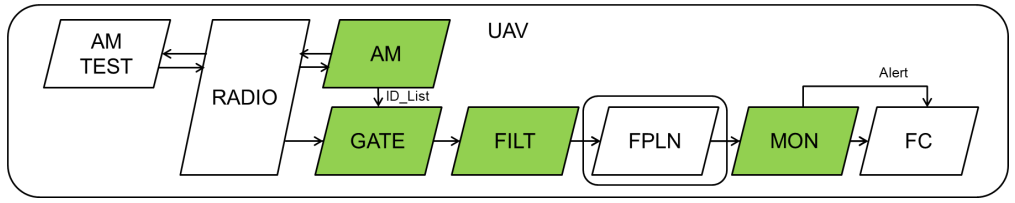
APPLICATIONS OF BRIEFCASE TOOLS

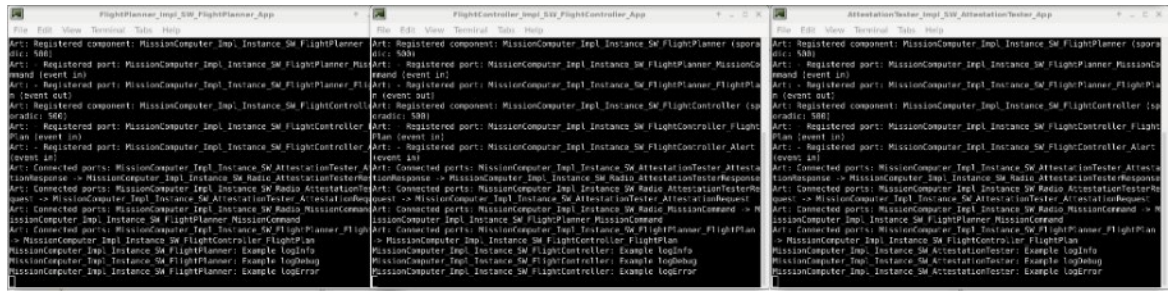


1. Experimental platform : UxAS mission planner
   - Models, demo videos

2. Demonstration platform : Collins CH-47 CAAS wireless gateway
   - Live demo in avionics lab

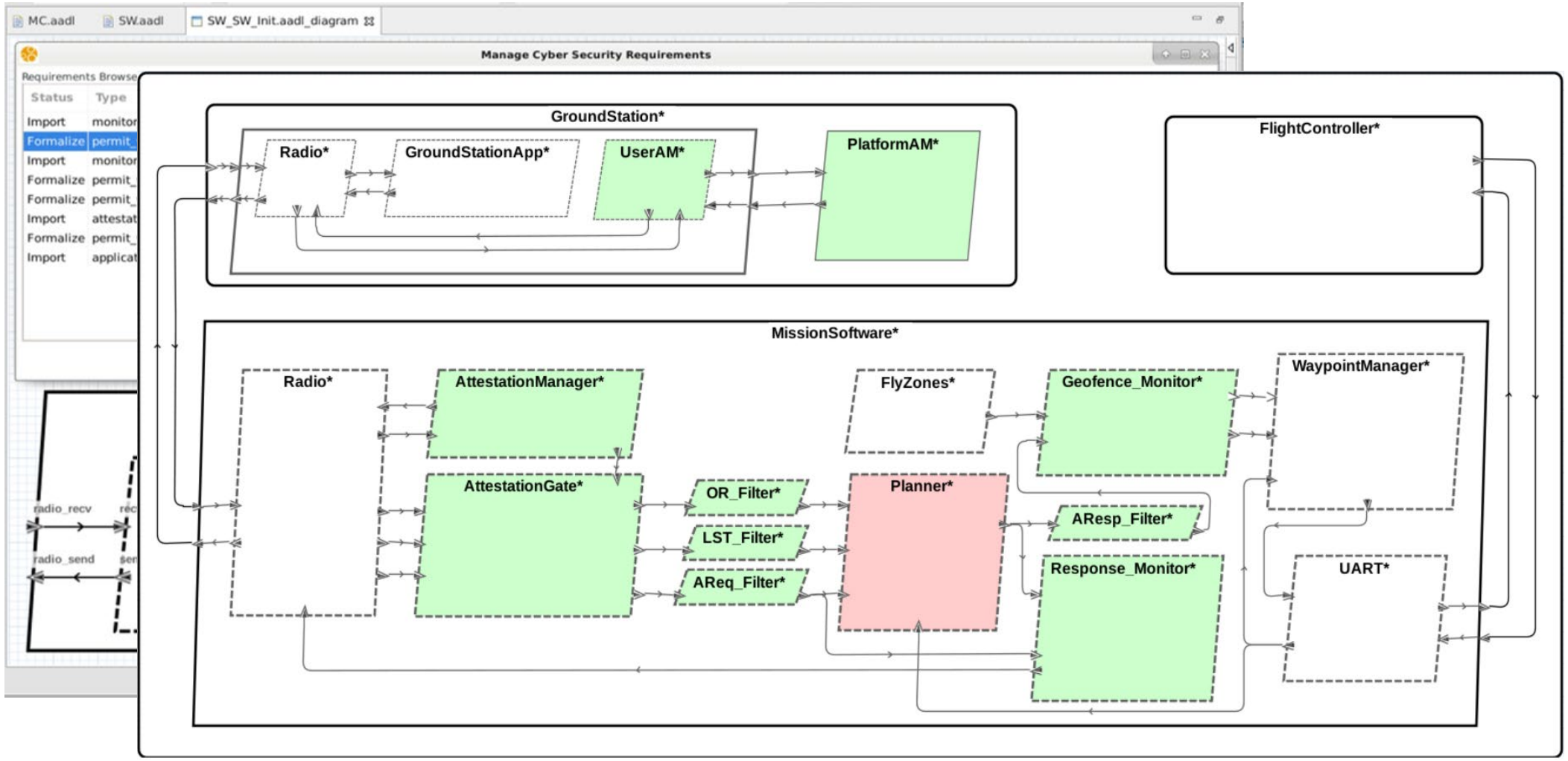
3. Self-contained workflow example : Simple UAV software on QEMU emulator
   - Tutorial, models, available on website





1. EXPERIMENTAL PLATFORM : SMALL UAV

SYSTEM ARCHITECTURE TRANSFORMATION



Collins Aerospace

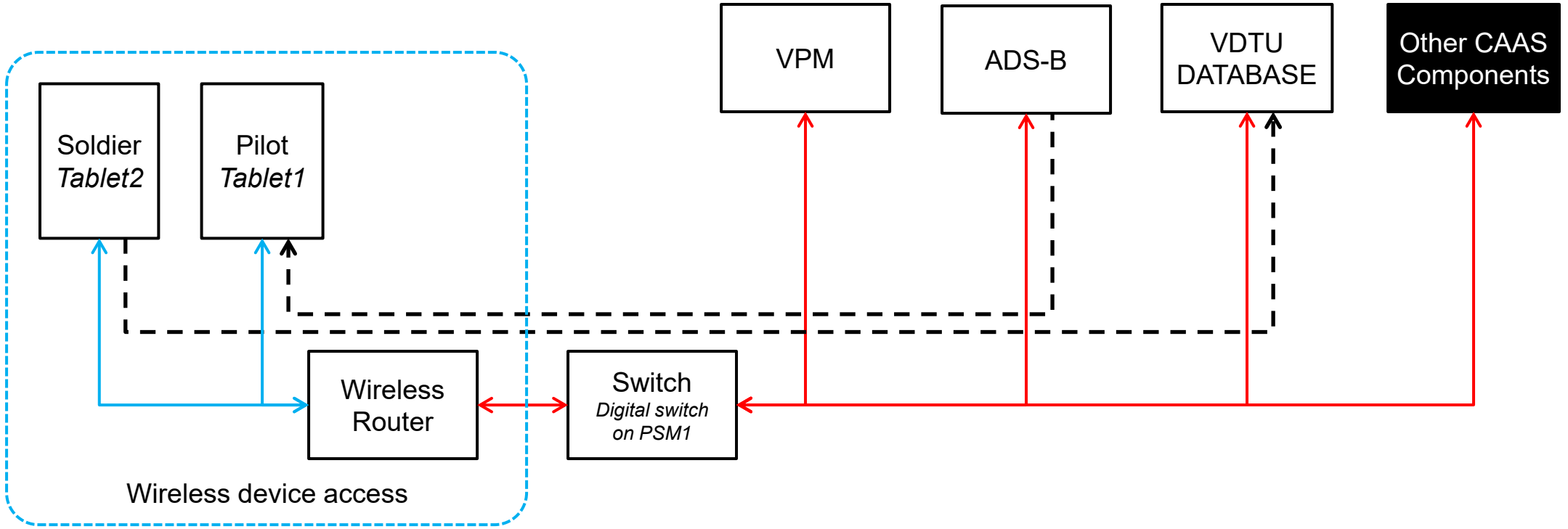
2. DEMONSTRATION PLATFORM

CAAS / CH-47 / FVL
COLLINS CUSTOMER EXPERIENCE CENTER
HUNTSVILLE AL

2. DEMO PLATFORM : BASELINE

COLLINS COMMON AVIONICS ARCHITECTURE SYSTEM (CAAS)

- Goal : Extend (securely) to add wireless connectivity

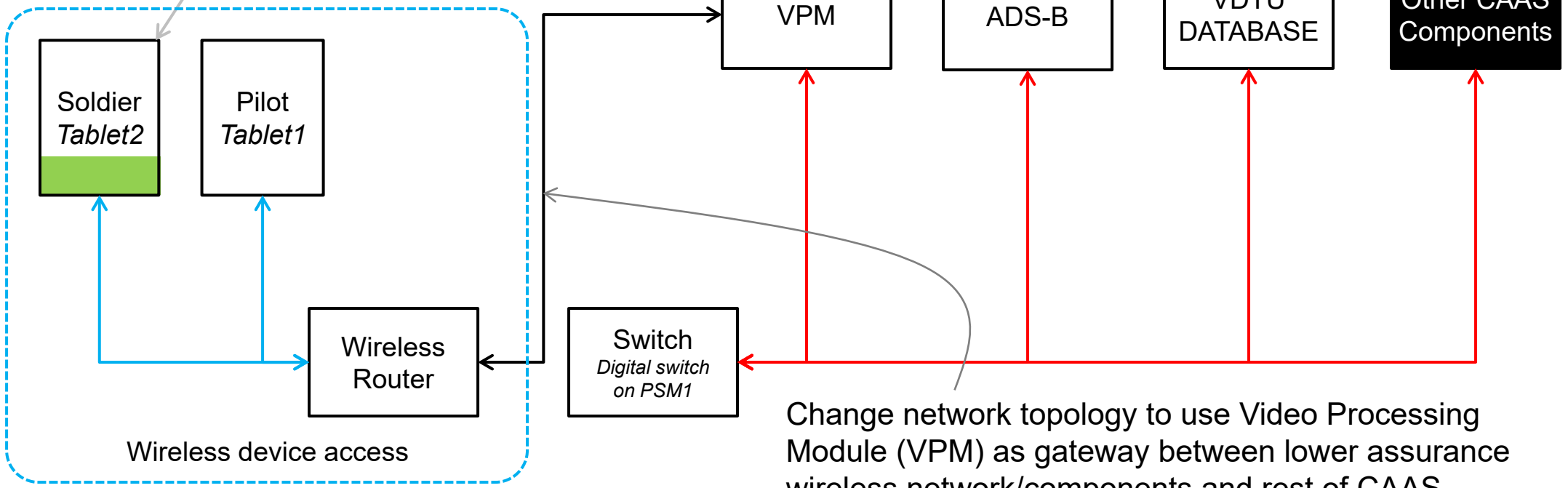


2. DEMO PLATFORM : HARDENED

COLLINS COMMON AVIONICS ARCHITECTURE SYSTEM

BriefCASE tools:
- Attestation of tablet(s)
- Filter messages to/from tablets
- Monitor ADS-B traffic for spoofing

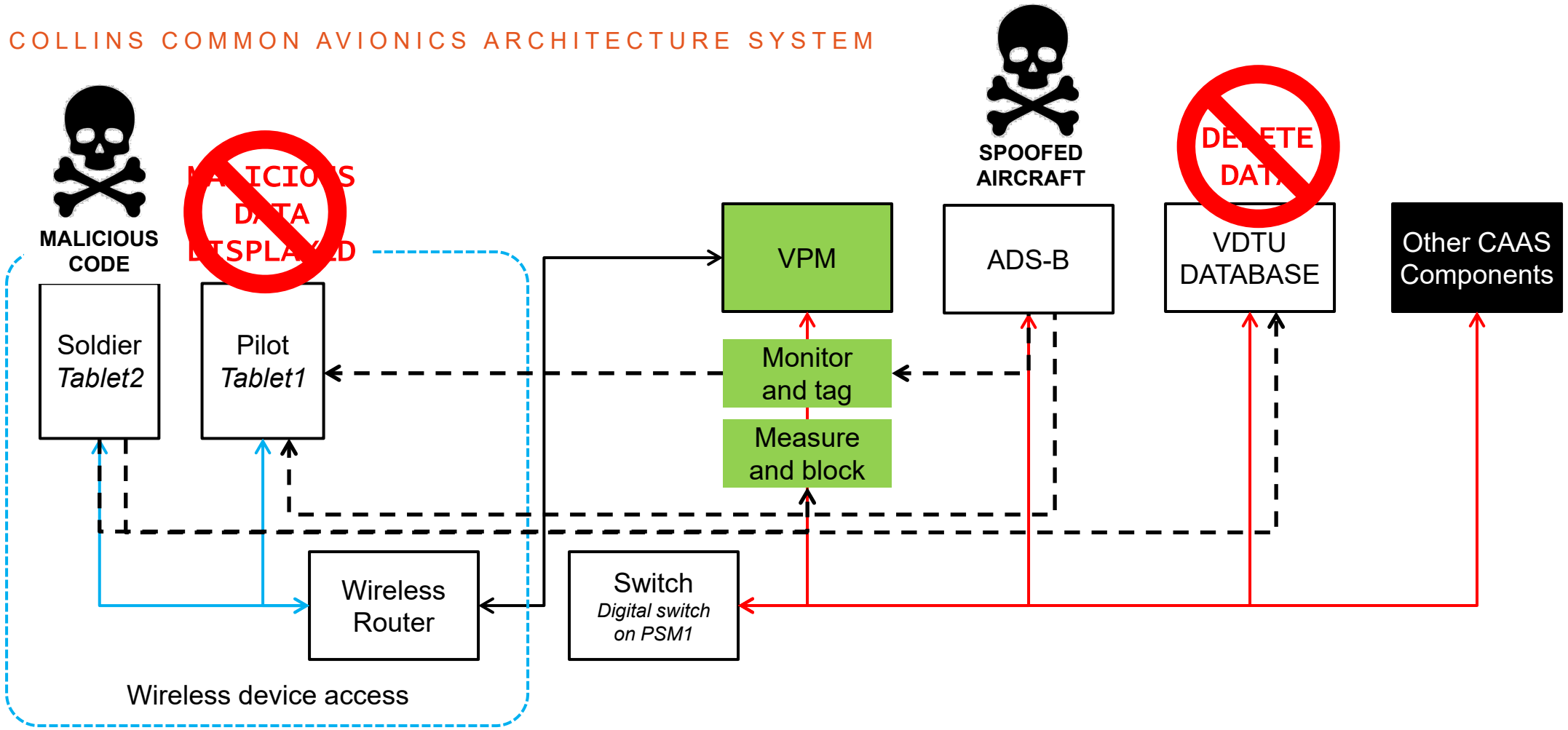
- seL4 hosting Linux
- Attestation



Change network topology to use Video Processing Module (VPM) as gateway between lower assurance wireless network/components and rest of CAAS

2. DEMO PLATFORM : ATTACKS

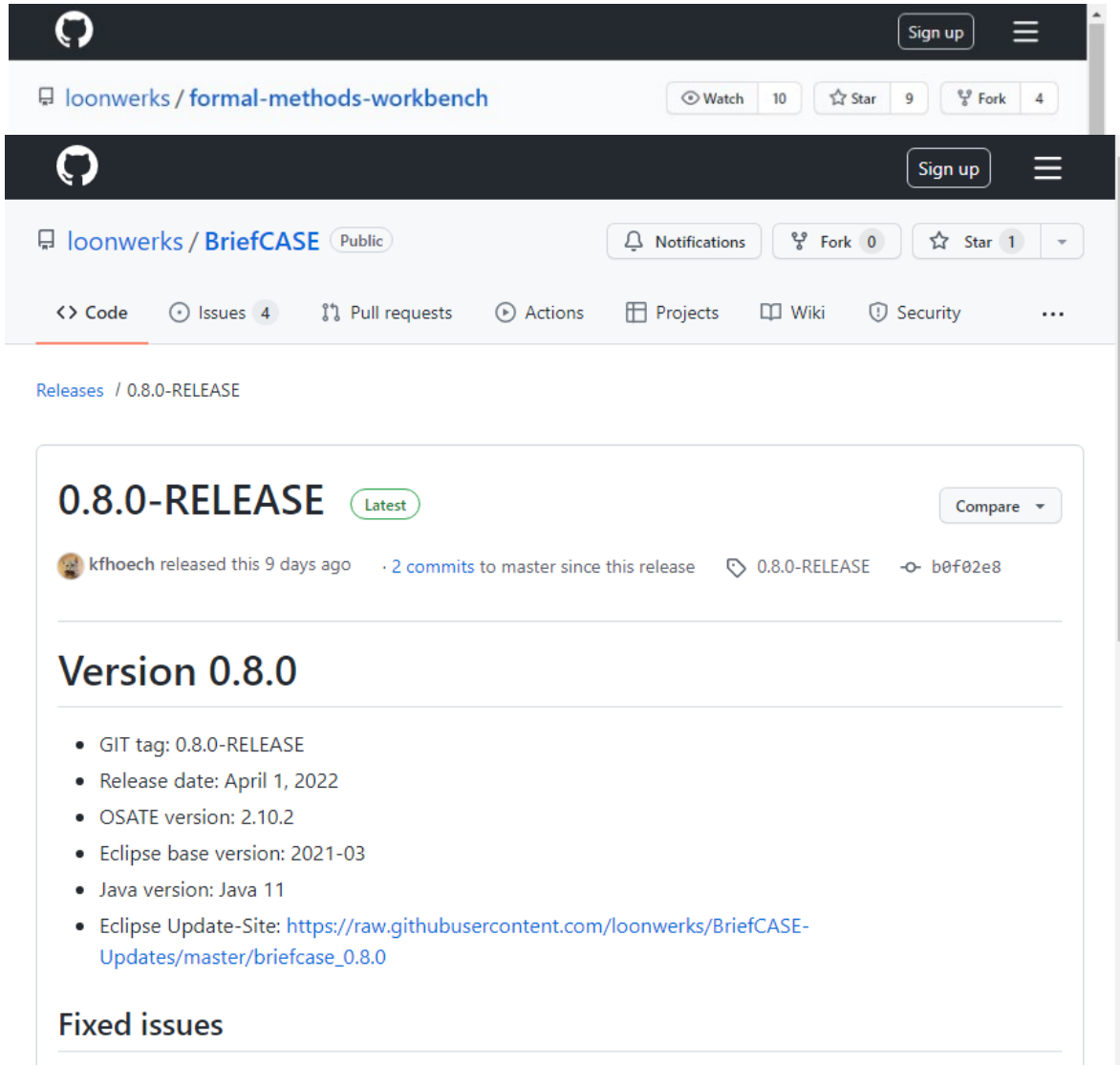
COLLINS COMMON AVIONICS ARCHITECTURE SYSTEM



OPEN-SOURCE SOFTWARE TOOL DISTRIBUTION

- Tool source code resides in several public GitHub repositories
  - https://github.com/loonwerks/CASE-Final
    also {/BriefCASE, /splat, /AGREE, /Resolute, /jkind}
  - https://github.com/ku-sldg
  - https://github.com/seL4
  - https://github.com/CakeML/cakeml
  - https://github.com/sireum
- Integrated OSATE/AADL tools and plugins
- Vagrant VM
  - Provides automatic, consistent, and reproducible provisioning of VM and native environments for developing and testing all CASE tools
- Documentation
  - Workflow example tutorial and models
  - User Guide
  - Videos, publications
- Overview
  - http://loonwerks.com/projects/case.html

3



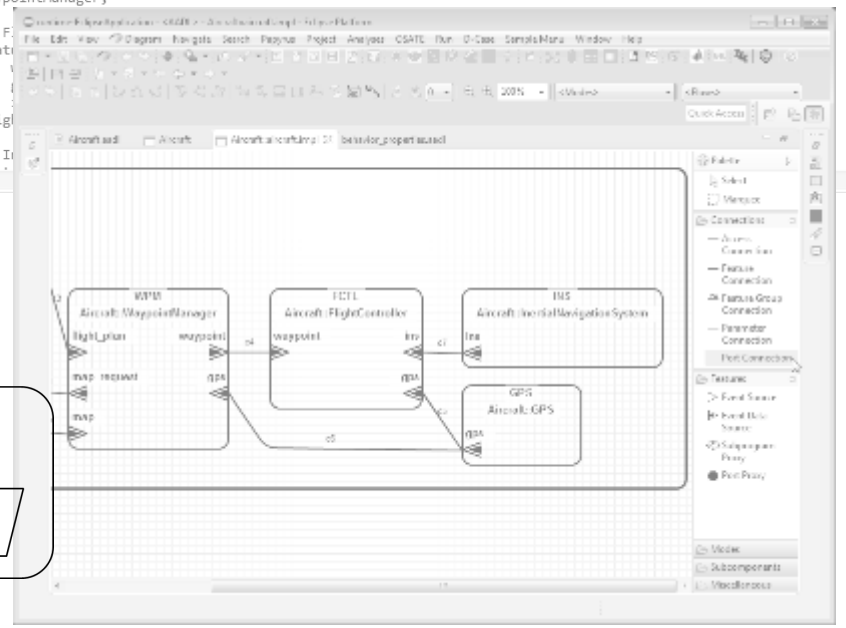
Collins Aerospace

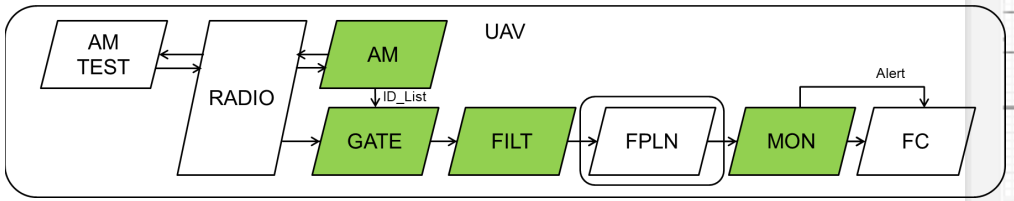
BRIEFCASE INTEGRATED WORKFLOW

WITH INTEGRATED ASSURANCE

1. Capture/import **cyber-resiliency requirements** based on initial AADL model analysis (GearCASE and DCRYPPS)
2. **Transform system architecture** model to satisfy cyber-resiliency requirements
3. Generate new **high-assurance components** from formal specifications (SPLAT) or pre-verified libraries
4. Verify system design using **formal methods** (AGREE) and information flow analysis (Awas)
5. Checks **model conformance** to standards (Resolint)
6. Generate **software integration code** (HAMR) directly from verified architecture models, targeting multiple operating systems (including seL4)
7. Document evidence/compliance with **assurance case** (Resolute)

SAE AS5506 STANDARD

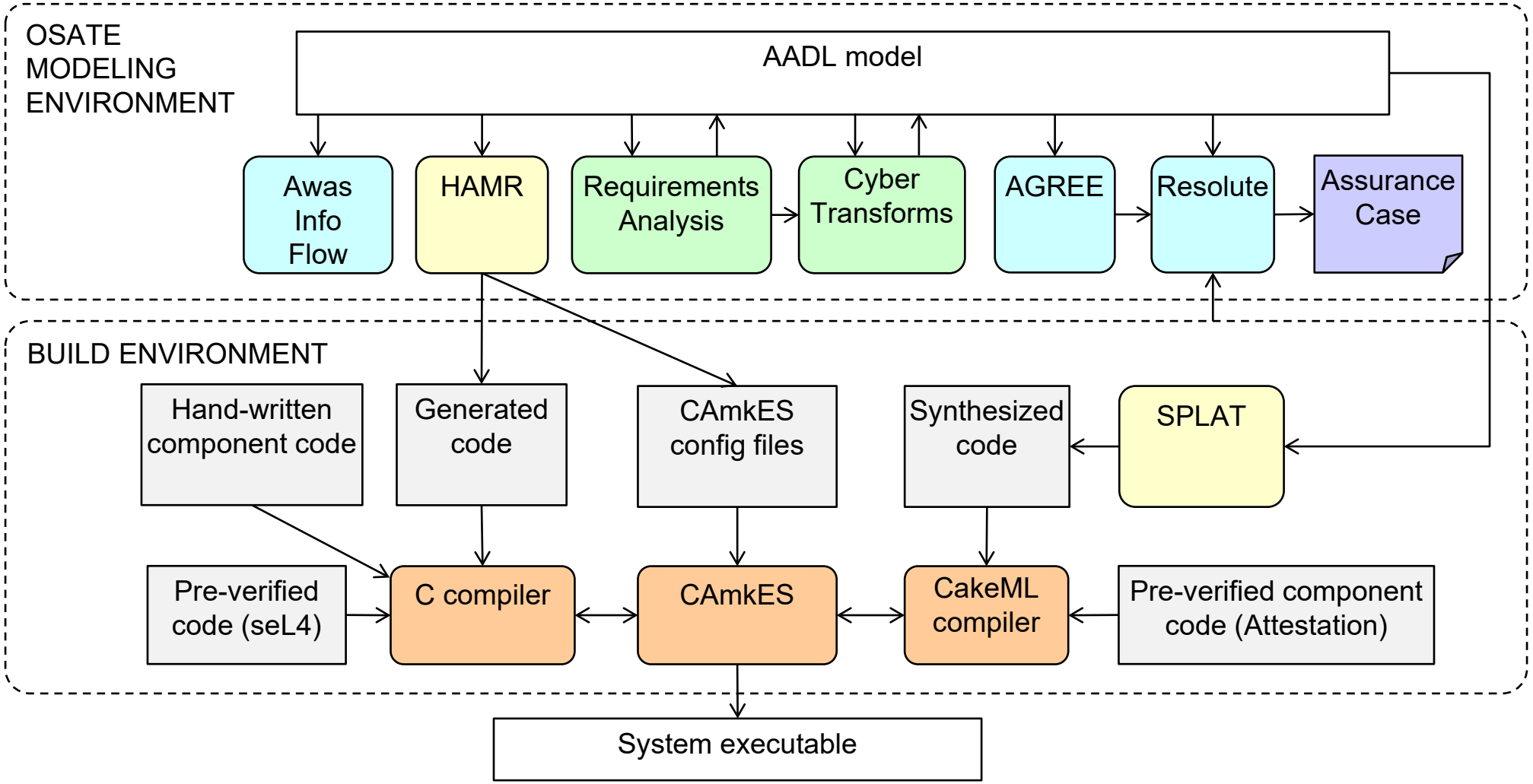




Cyber Assured Systems Engineering at Scale
IEEE Security & Privacy
May 2022

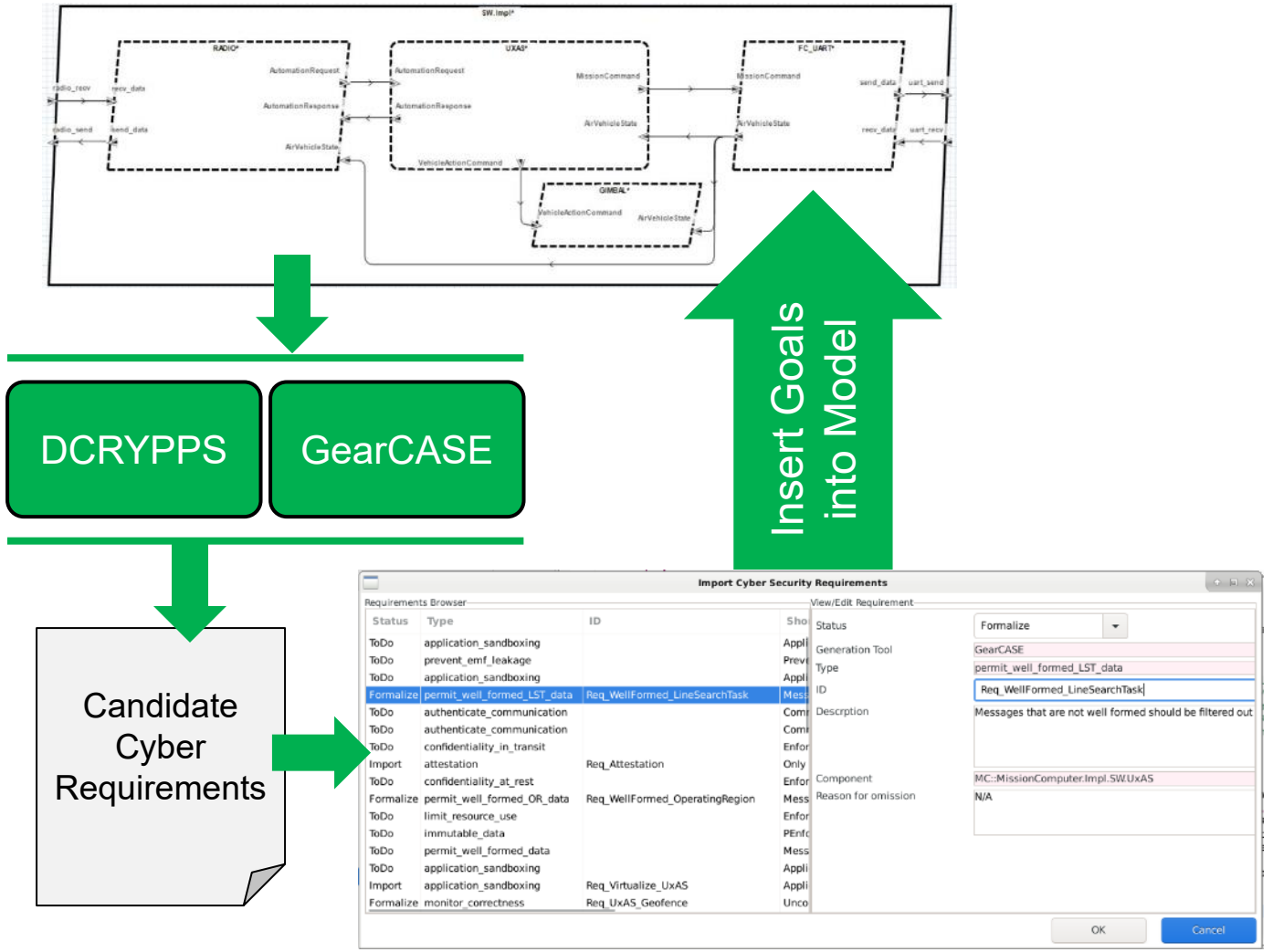
Collins Aerospace

BRIEFCASE TOOL ARCHITECTURE



1. GENERATE / IMPORT CYBER REQUIREMENTS

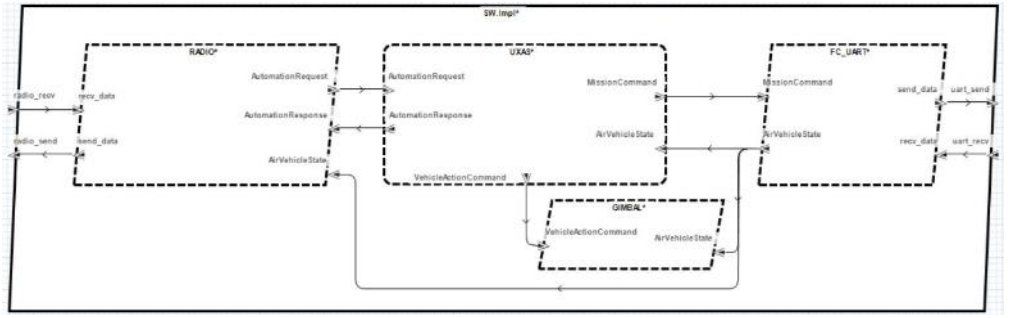
- Choose one of the Cyber Requirements generation tools
  - CRA GearCASE plugin
  - Vanderbilt/DOLL DCRYPPS plugin
- Initial model data is exported to selected tool
- Requirements import wizard manages the generated requirements
  - Select action
  - Naming/tagging
  - Associate with formal properties
- **Requirements inserted into model as Resolute goals (GSN)**
  - We will build an assurance argument to satisfy these goals

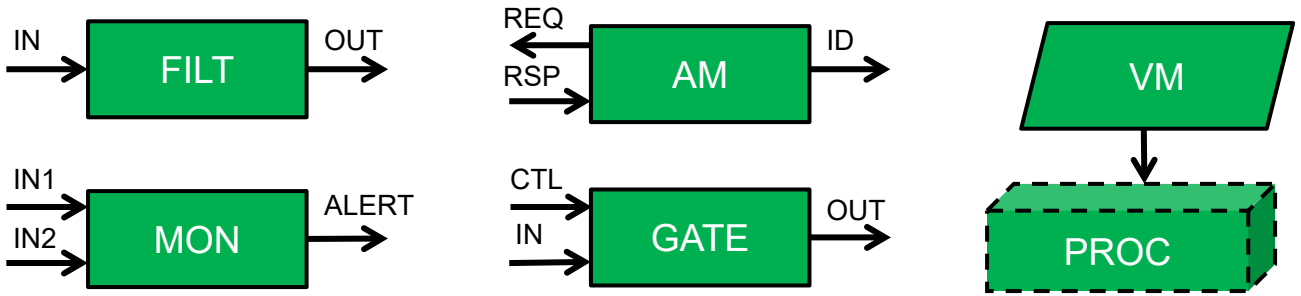


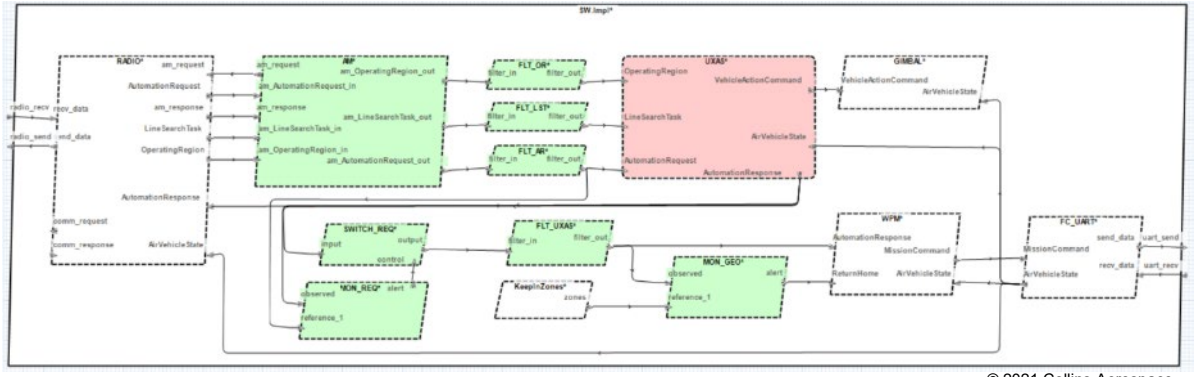
2. APPLY CYBER TRANSFORMATIONS

- Cyber requirements tools provide model context and sometimes suggested mitigation
- System engineer selects from available cyber-resiliency transformations
  - Filter
  - Monitor
  - Gate (controlled by monitor)
  - Attestation
  - Virtualization
  - seL4 build prep
- Wizard interface collects needed configuration data
- **Tool automatically transforms AADL model**

- Also adds Resolute assurance case strategy to show how the associated goal (requirement) is satisfied



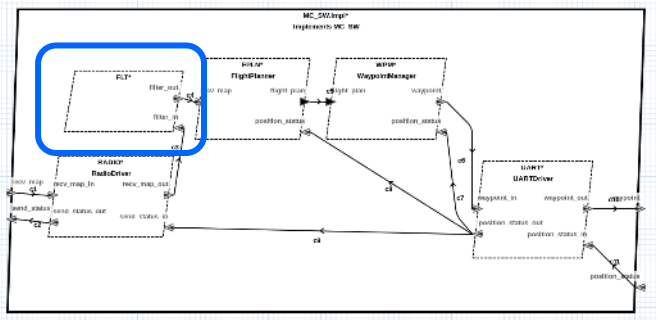




Collins Aerospace

2A. INSERT ASSURANCE CASE STRATEGY

- **Resolute links cyber transform to goal as a *strategy* in GSN**
- Checks for violations/changes that impact correctness
- Collects evidence and generates assurance case



Goal Structuring Notation

```
thread Filter
    features
        filter_in: in event data port
        filter_out: out event data po
    properties
        CASE::COMP_TYPE => FILTER;
        CASE::COMP_IMPL => "CakeML";
        CASE::COMP_SPEC => "(\\i{-90,
    annex agree {**
        guarantee "The Flight Planner
```

```
package CASE_Requirements
private

    annex Resolute {**

        goal Req_WellFormed(comp_
            ** "[permit_well_form
            context Generated_By
            context Generated_On
            context Req_Component
            context Formalized :
            agree_property_checke
```

```
annex Resolute {**

    ---------------------------
    -- MODEL TRANSFORMATIONS --
    ---------------------------

    -- Top-level claim for proper insertion of a filter
    goal add_filter(comp_context : component, filter : component, conn : connection, msg_type : data) <=
        ** "Filter " filter " is properly added to component " comp_context **
        filter_exists(filter, comp_context, conn) and component_not_bypassed(filter, comp_context, msg_type) and component_implemented(filter)

    -- Top-level claim for proper insertion of attestation manager
    goal add_attestation_manager(comm_driver : component, attestation_manager : component, attestation_gate : component) <=
        ** "Attestation Manager added for communications driver " comm_driver **
        attestation_manager_exists(comm_driver, attestation_manager) and attestation_manager_not_bypassed(comm_driver, attestation_manager, attes
```

Problems | Properties | AADL Property Values | AGREE Results | Console | Progress | Assurance Case

- well_formed(FPLN : SW::FlightPlanner, "good_gs_command")
  - FPLN : SW::FlightPlanner only receives well-formed messages
    - A filter exists on the communication pathway immediately before FPLN : SW::FlightPlanner
    - Filter cannot be bypassed
    - Filter property implemented by CakeML
    - AGREE property passed: [good_gs_command]

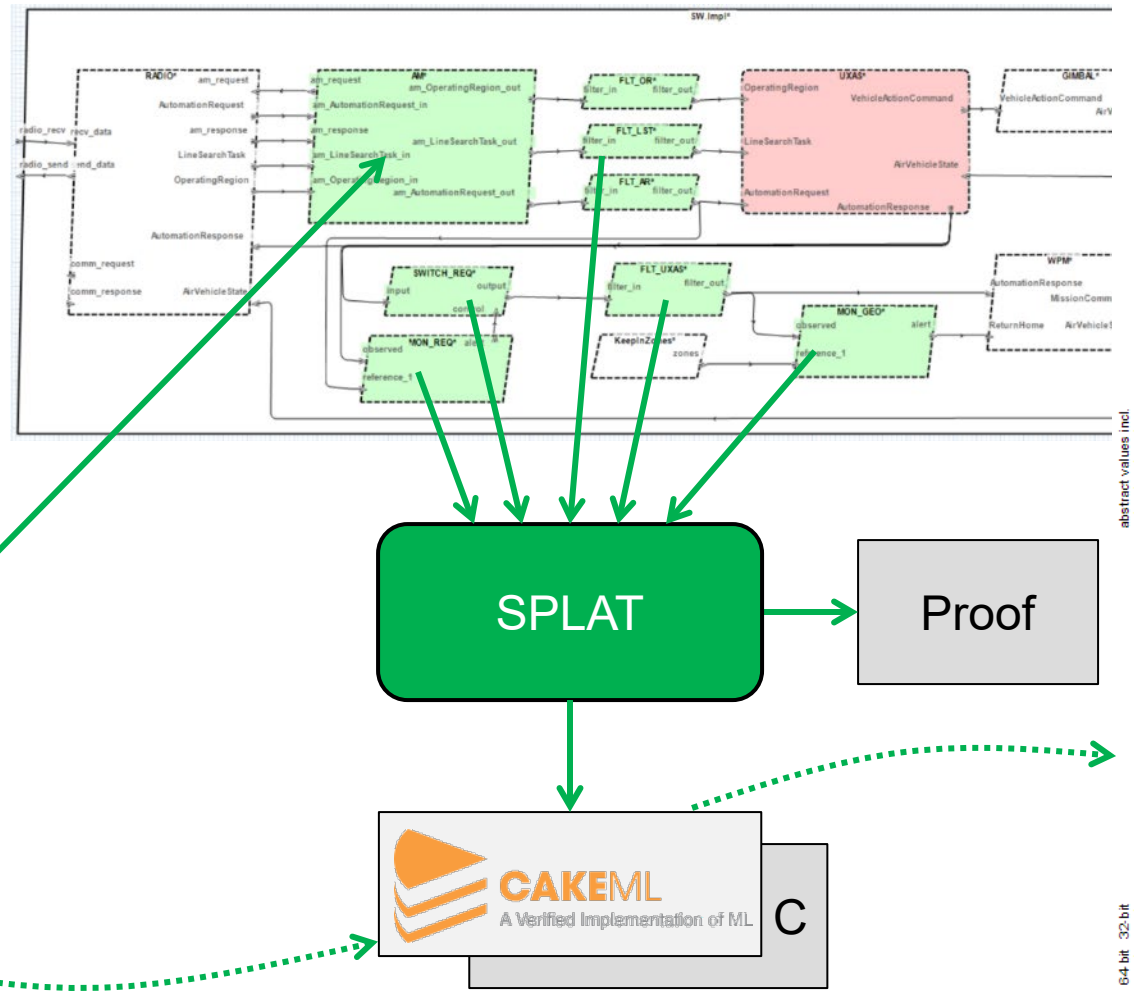
Collins Aerospace

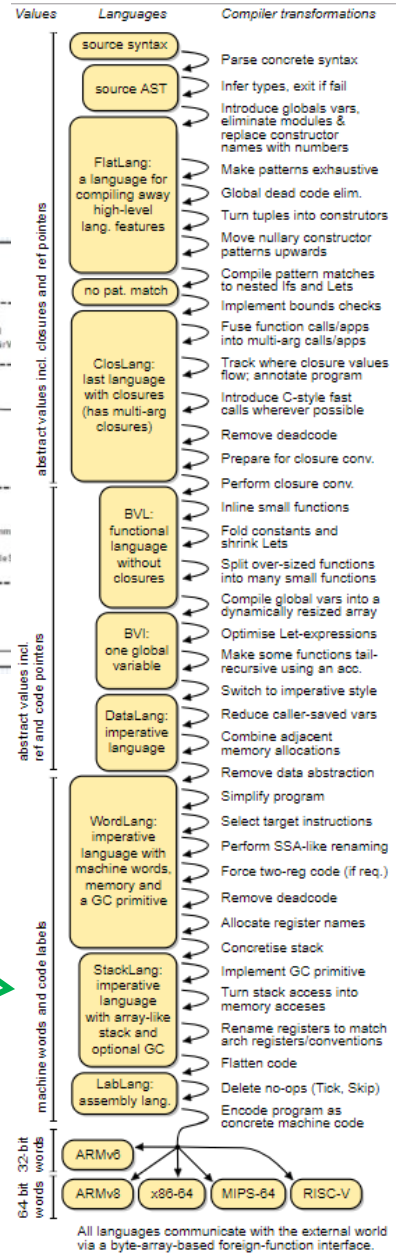
3. GENERATE HIGH ASSURANCE COMPONENTS

- Some of the cyber transforms insert new high-assurance components into the model
- The behavior of the component (its contract) is specified in AGREE
- **SPLAT generates component implementations from their specifications**
- SPLAT also generates a proof showing that the component implements its specification

- Other components (such as the Attestation Manager) are pre-built pre-verified libraries
- Their implementations are essentially library functions that are added to the build, possibly with some configuration data from the model

- Code can be generated in the CakeML language which has a verified compiler

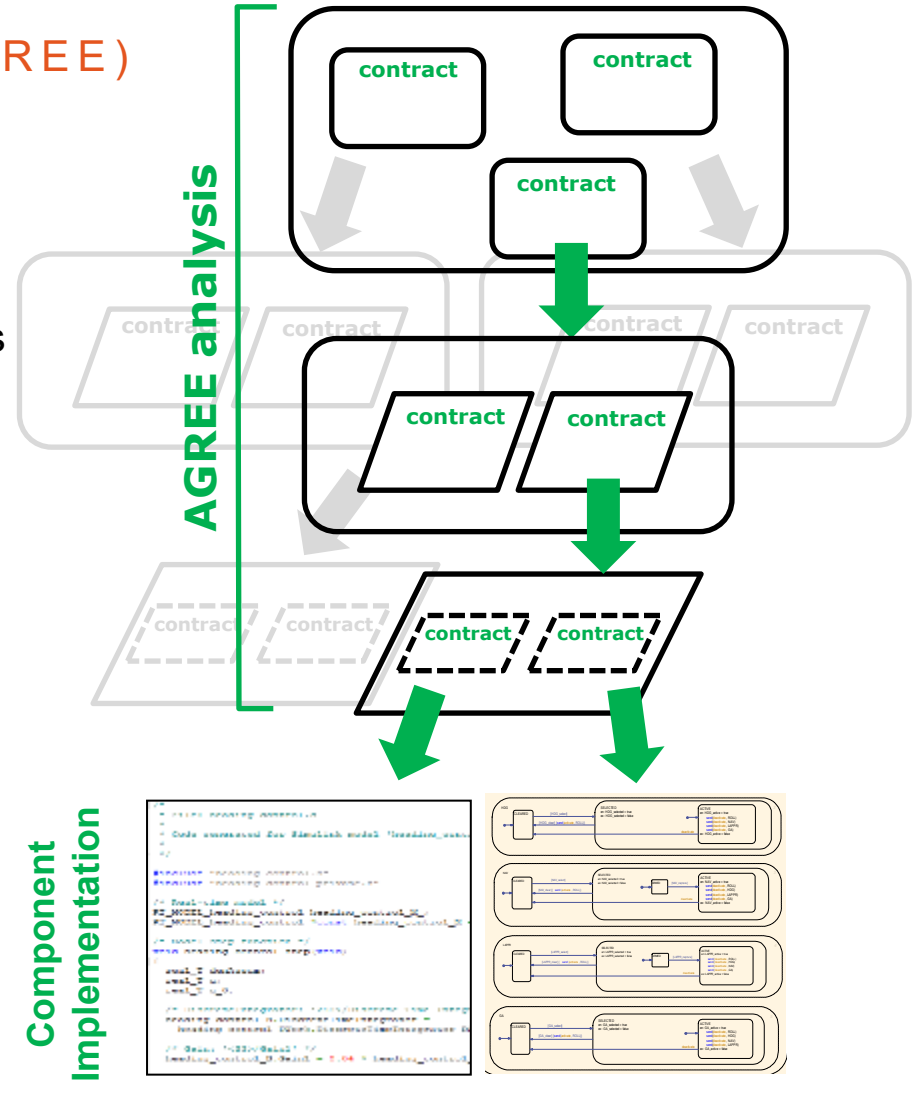




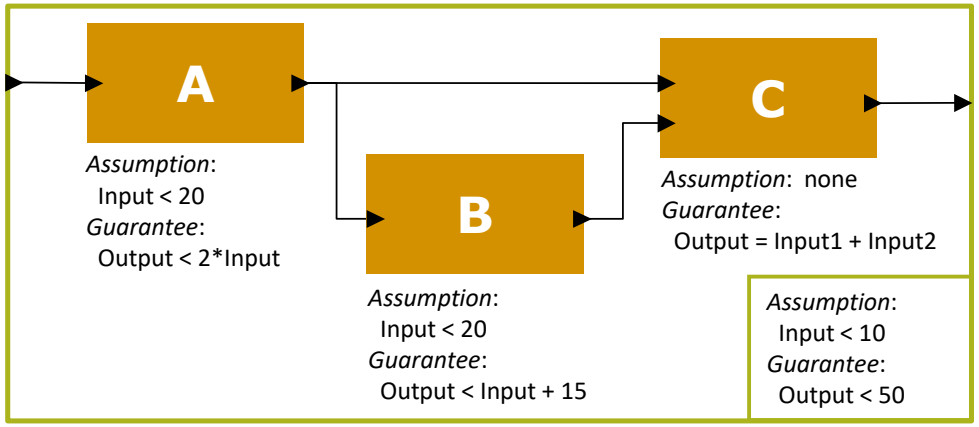
4. ANALYZE SYSTEM BEHAVIOR

ASSUME GUARANTEE REASONING ENVIRONMENT (AGREE)

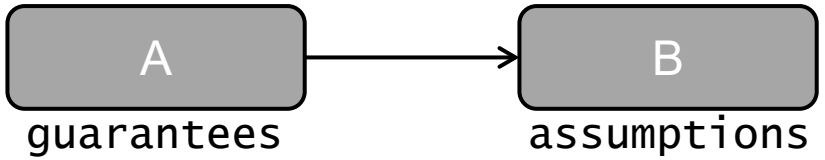
- Contract-based *compositional reasoning* provides **scalability**
- Each component has a *contract* consisting of assumptions and guarantees
- The contract of a component abstracts the behavior of its implementation
- Contracts at each layer must be satisfied by contracts of its subcomponents
- Leaf component contracts must be satisfied by implementation



Composition



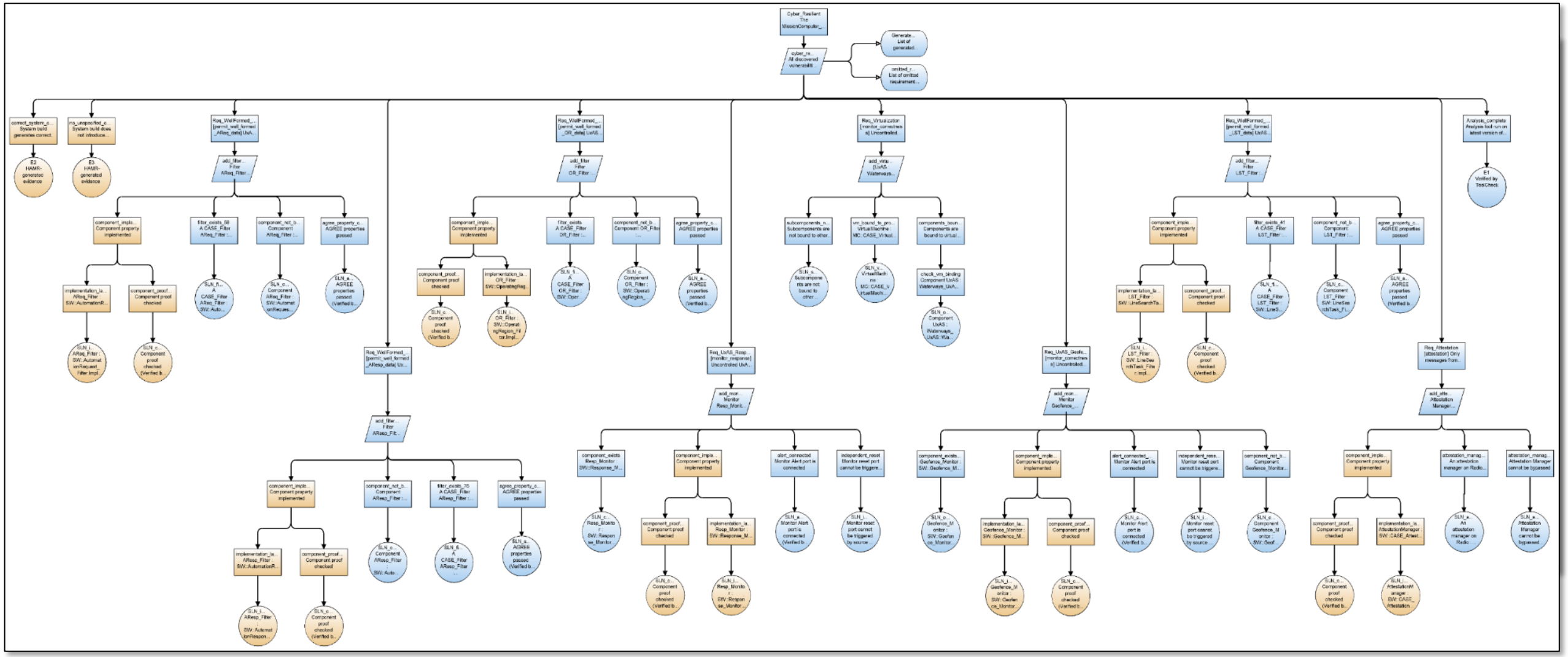
Modularity



Collins Aerospace

ASSURANCE ARGUMENT

INTEGRATE AND DOCUMENT EVIDENCE OF CORRECTNESS AND COMPLIANCE



HAMR

CODE GENERATION FRAMEWORK

TODD CARPENTER : ADVENTIUM

CONCLUSION

- **Design-in cyber-resiliency**
  - Automated architecture transforms for threat mitigation
  - High assurance components generated from specifications
  - Techniques to deal with legacy code (cyber retrofit)
- **Build what you model**
  - Build system directly from detailed, verified AADL model
  - Makes the security guarantees of seL4 accessible to system developers
  - Ability to target different platforms to facilitate incremental debugging/development
- **Provide evidence**
  - Formal verification of functional and cyber-resiliency properties, information flow properties, component proofs
  - Code generation equivalence to model, seL4 build preserves properties
  - Integrate evidence as an assurance case demonstrating how/why requirements are satisfied