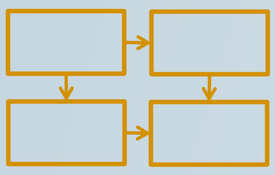
Architecture-Driven Assurance

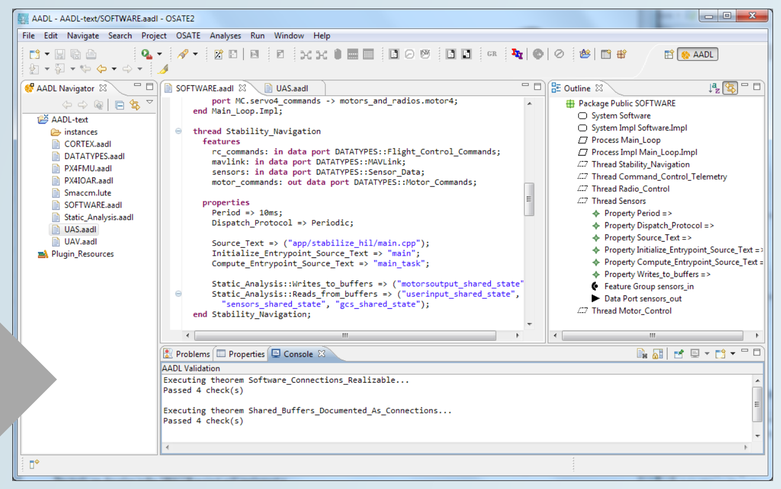
Rockwell Collins, University of Minnesota, Galois, Data61

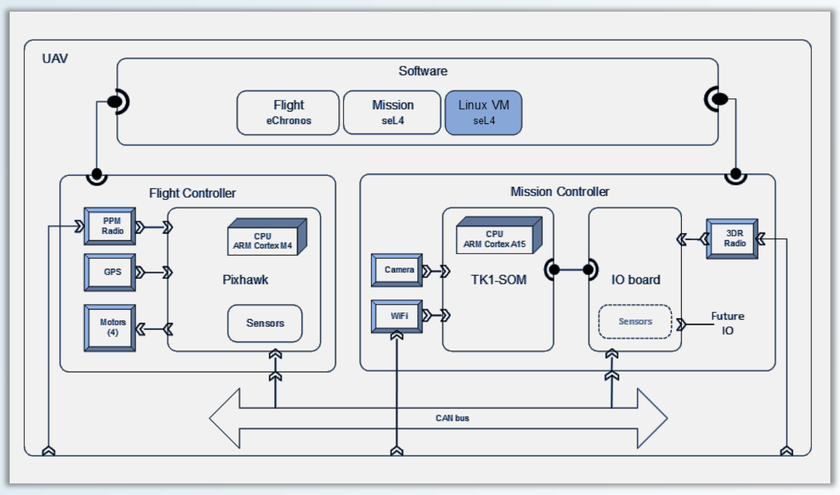
- Comprehensive use of *formal methods* throughout the development process is needed to ensure that vulnerabilities are eliminated from critical military assets.
- Integrated tools for architectural modeling, analysis, and synthesis make this approach practical and effective.

1 System Architecture modeled in AADL

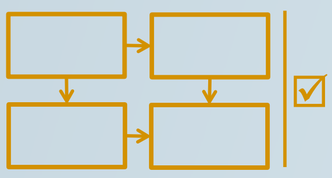


Architecture Analysis and Design Language



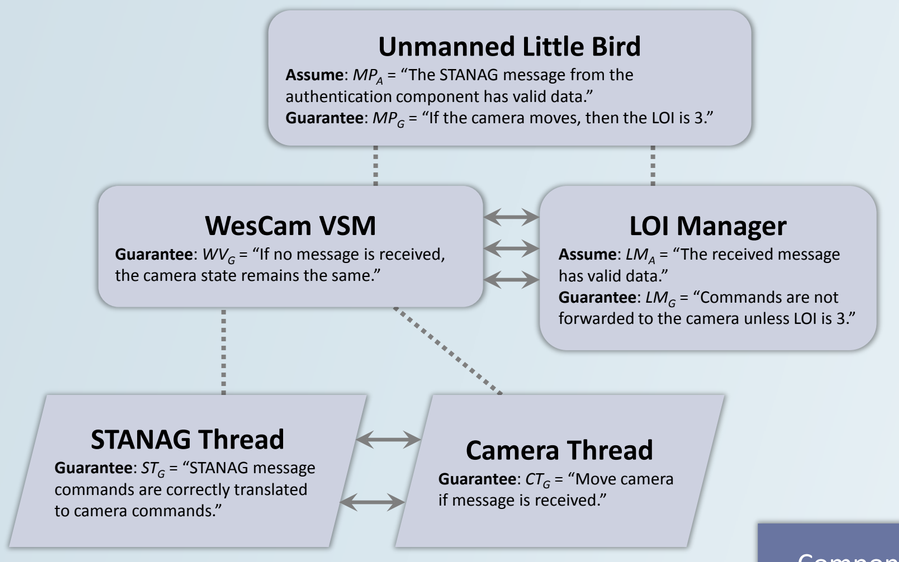


2 Architecture model is correct



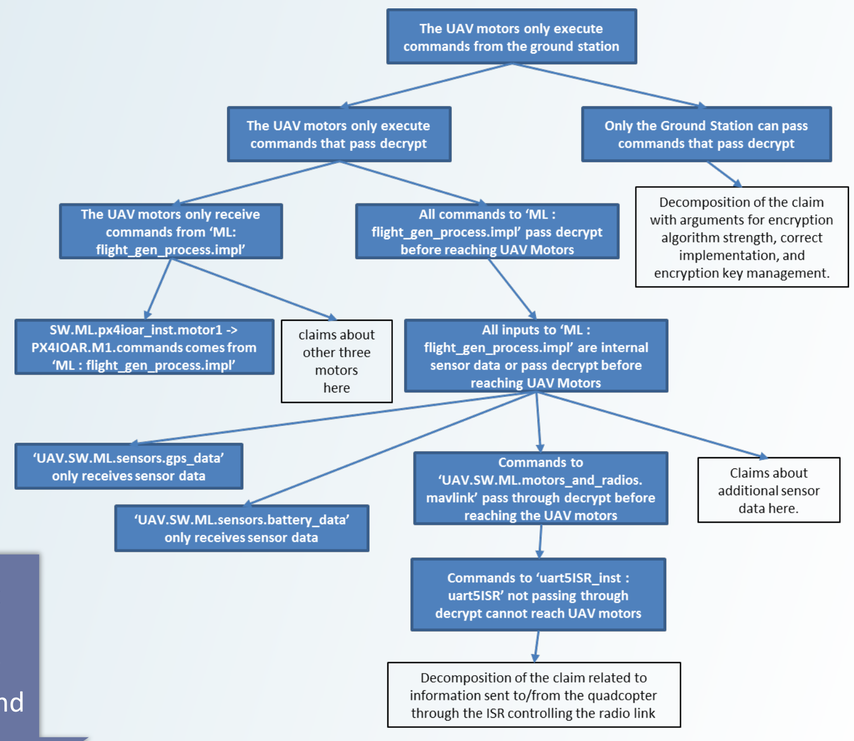
Assume-Guarantee Reasoning Environment (AGREE)

Compositional reasoning about system behavior based on formal contracts added to AADL model elements



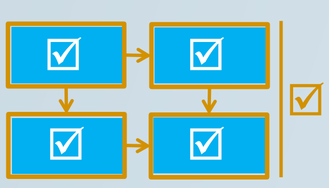
Resolute

Logic and tool for generating assurance cases from structure of AADL model and claims added to model



Component contracts checked for consistency and realizability

3 Software components are correct

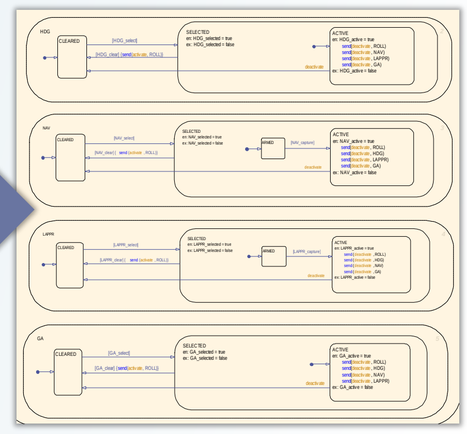


Ivory/Tower DSL

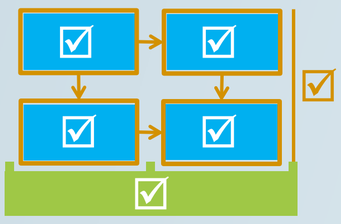
- Language prevents common C errors
- Generates memory-safe code
- Embeds checks to detect arithmetic/interface errors

Autopilot Code
*.c, *.h

AGREE contracts exported to component development environments (e.g., Simulink) for verification



4 System does what the model says

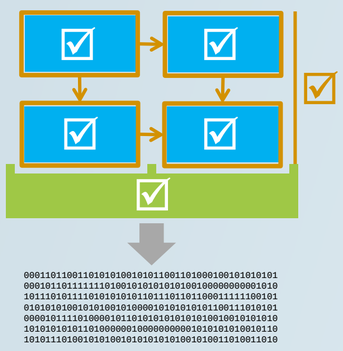


seL4

- Secure kernel guarantees isolation between components.
- No information flows other than those explicitly defined in the architecture.

Formally verified from specification to binary

5 Software implementation corresponds to model



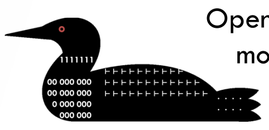
Trusted Build

Automatically generates implementation code from architecture model, component specifications, and kernel/OS build system

- **Linux**
- **VxWorks**
- **eChronos**

CAmkES
Build tool for **seL4** systems

Build for other OS/RTOS but with reduced assurance



Open source tools and models available at Loonwerks.com

www.darpa.mil

DARPA

Distribution Statement A - Approved for Public Release, Distribution Unlimited