• 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

**2** Architecture model is correct

**3** Software components are correct

**4** System does what the model says

**5** Software implementation corresponds to model

---

- **Assume-Guarantee Reasoning Environment (AGREE)**
  - Compositional reasoning about system behavior based on formal contracts added to AADL model elements

- **Unmanned Little Bird**
  - Assumes: "The STANAG message from the architecture component has valid data."
  - Guarantees: "If the LOI is 3, the camera state remains the same." (ex: NAV_active = true; deactivates NAV; deactivate)

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

- **Autopilot Code**
  - Assumes: "If no message is received, the camera state remains the same." (ex: NAV_active = false

- **Component contracts checked for consistency and realizability**

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

- **CMkES**
  - Build tool for sel4 systems

- **Linux**
  - VxWorks
  - eChronos

- **Formally verified from specification to binary**

---

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

---

**Open source tools and models available at Loonwerks.com**