# RUN-TIME ASSURANCE ARCHITECTURE FOR LEARNING-ENABLED SYSTEMS

**DR. DARREN COFER**
DARREN.COFER@COLLINS.COM
HIGH CONFIDENCE SYSTEMS & SOFTWARE
SEPTEMBER 2020

**DARPA I2O**
ASSURED AUTONOMY

Kestrel Institute
Stanford University
University of Minnesota

Collins Aerospace

# ASSURANCE CHALLENGES

## LEARNING-ENABLED COMPONENTS (LEC) IN SAFETY-CRITICAL SYSTEMS

OR "WHY THIS MIGHT BE A BAD IDEA"

**Implementation**

**Verification**

**Requirements**


https://xkcd.com/1838/

For safety-critical systems, assurance is not just showing that things work, but also showing that there are no surprises

- Absence of *unintended functionality* (DO-178C)

# DO-178C

## SOFTWARE DESIGN ASSURANCE

- Demonstrate that software implements its requirements
- ***and nothing else***

**DO-178C**

6.1 Purpose of Software Verification

…

d. The Executable Object Code satisfies the software requirements (that is, intended function), and **provides confidence in the absence of unintended functionality.**

**DO-248C**

**FAQ #43:** What is the intent of structural coverage analysis?

**Answer:** DO-178C/DO-278A sections 6.4.4.2 and 6.4.4.3 define the structural coverage analysis activities and the possible resolution for code structure that was not exercised during requirements-based testing.

…

2. **Provide a means to support demonstration of absence of unintended functions**.

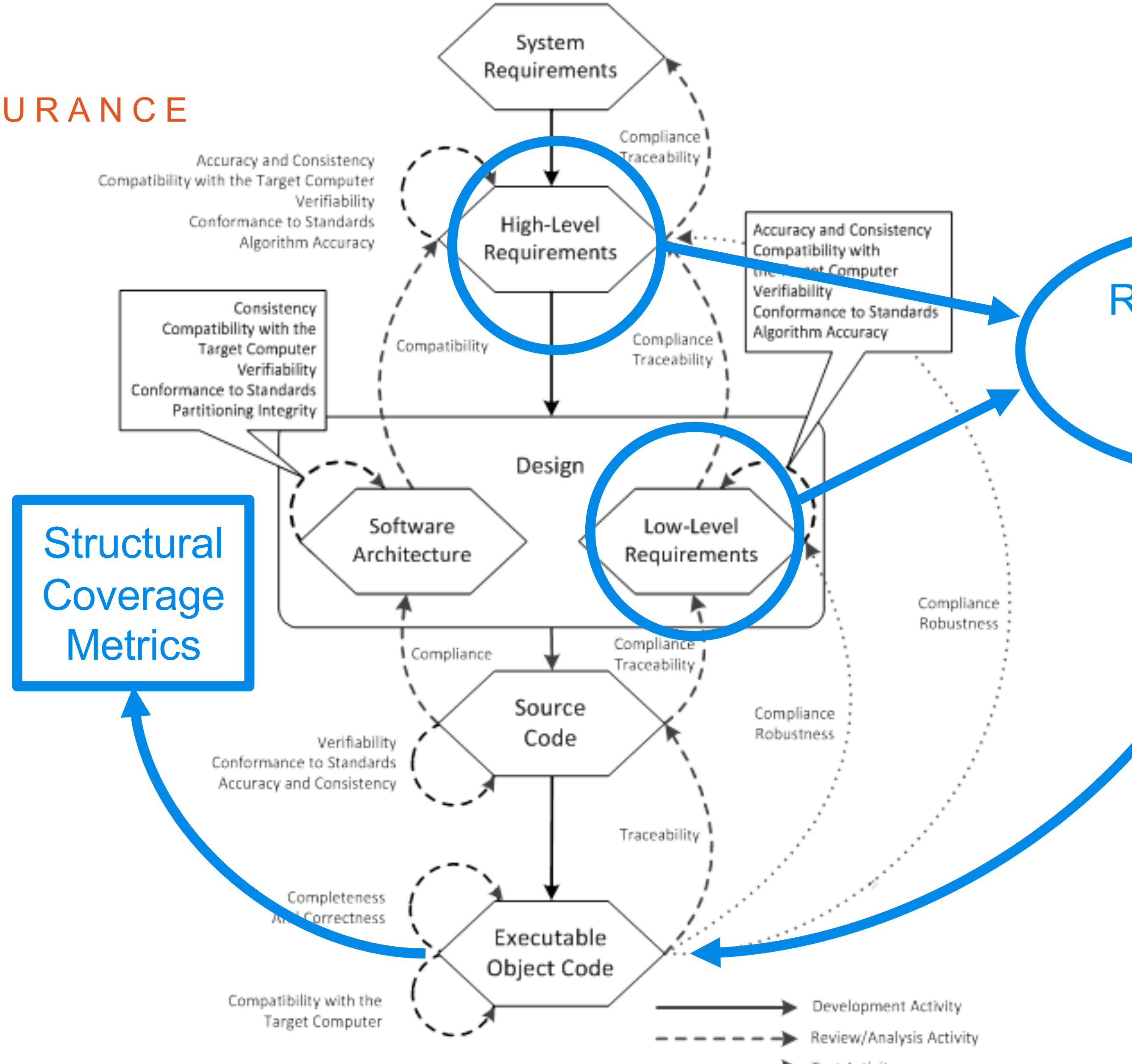



*Diagram adapted from DO-333 Formal Methods Supplement to DO-178C and DO-278A*


Collins Aerospace

# RUN-TIME ASSURANCE ARCHITECTURE

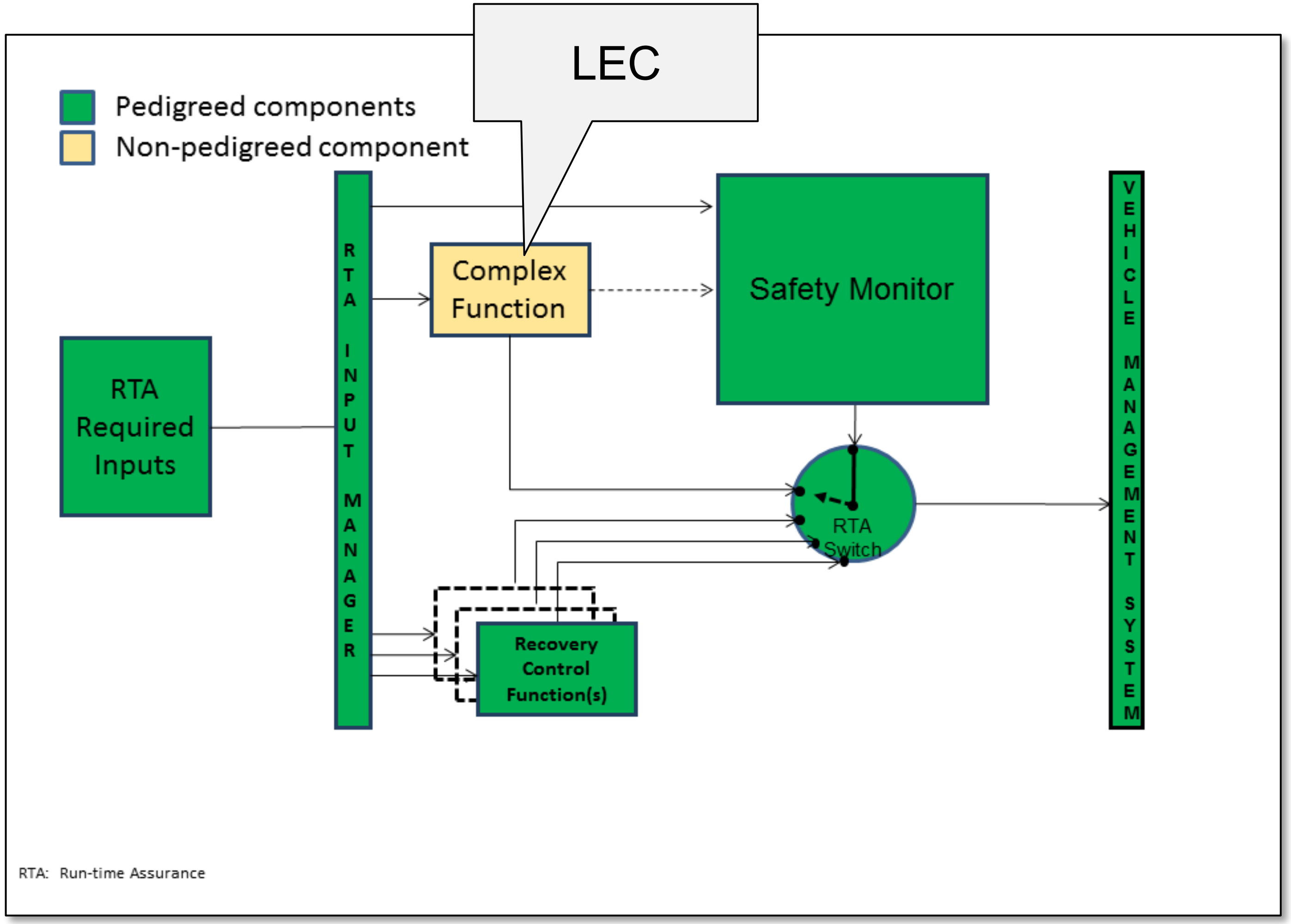
## AN APPROACH TO PREVENT UNINTENDED FUNCTIONALITY

- Learning-Enabled Component (LEC) provides accuracy, performance, efficiency
  - But we are unable to establish *comprehensive* assurance needed for safety
  - **Unsafe/unexpected behavior may be triggered by new or unanticipated inputs**
- How do we guarantee absence of unintended functionality?
  - **Nothing in LEC source code can be traced to design intent (requirements)**
  - Can't rely on structural coverage (DO-178) or formal methods (yet)
- Embed LEC in **run-time assurance architecture** to guarantee that there are no surprises
  - Run-time monitors detect unsafe/unexpected behaviors
  - Switch to alternative safe behavior
  - Ideally, use formal methods to verify correctness of the architecture (limit to safe behaviors)
  - LEC may still contain surprises, but architecture ensures that there is no impact on system safety (no unintended functionality)

# ASTM F3269-17

- Standard Practice For Methods To Safely Bound Flight Behavior Of Unmanned Aircraft Systems Containing Complex Functions
- “Complex Function” = LEC
- **Monitor LEC to detect and prevent unintended functionality**

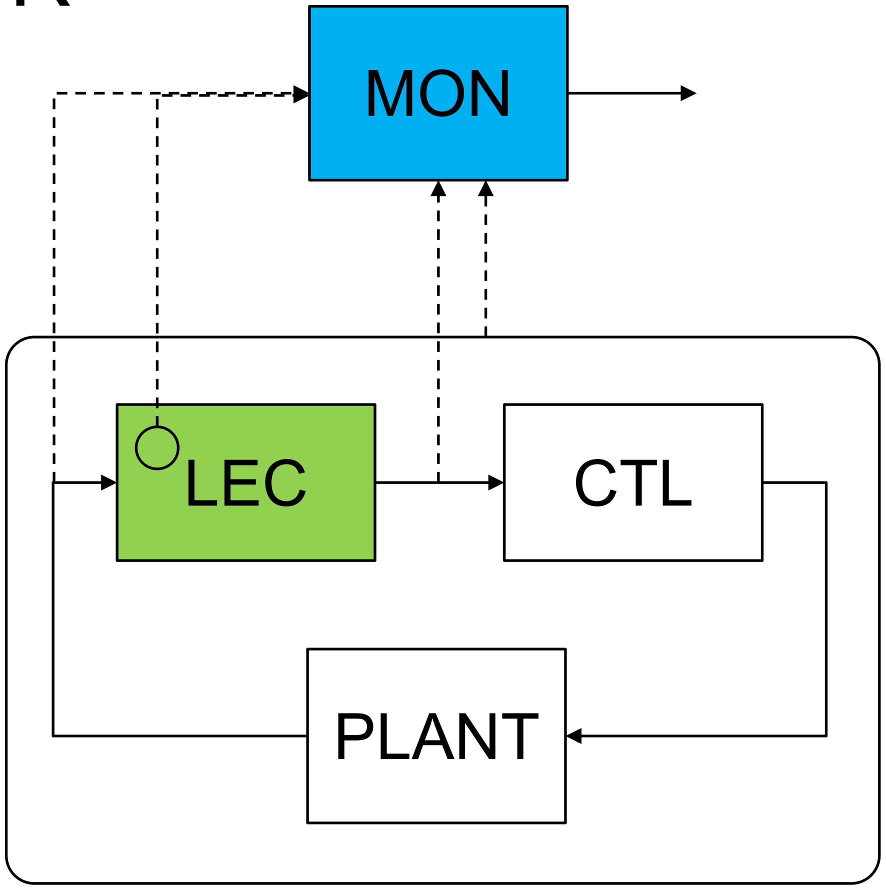
Clark, Koutsoukos, Porter, Kumar, Pappas, Sokolsky, Lee, Pike, “A Study on Run Time Assurance for Complex Cyber Physical Systems,” AFRL Report, 2013




Goal is to develop the standard to a level of capability that defines run-time monitoring (RTA) attributes to a level that the FAA will agree that monitors and architecture developed to this standard are sufficient to allow the UAS to evolve the complex function with its associated avionics equipment and sensors without requiring vehicle recertification as the CONOPS evolve after initial certification

# TYPES OF RUN-TIME MONITOR

- LEC inputs
  - Detect regions of input space where LEC is known to have poor performance or lack robustness
- LEC internal state
  - Detect activation patterns that are linked to poor performance, low confidence, or “surprise”
- LEC outputs
  - Computed outputs violate specified bounds or invariants
  - Inconsistent outputs
- System state
  - Directly monitor violations of system safety properties
  - Ex: geofence, flight envelope, position on runway



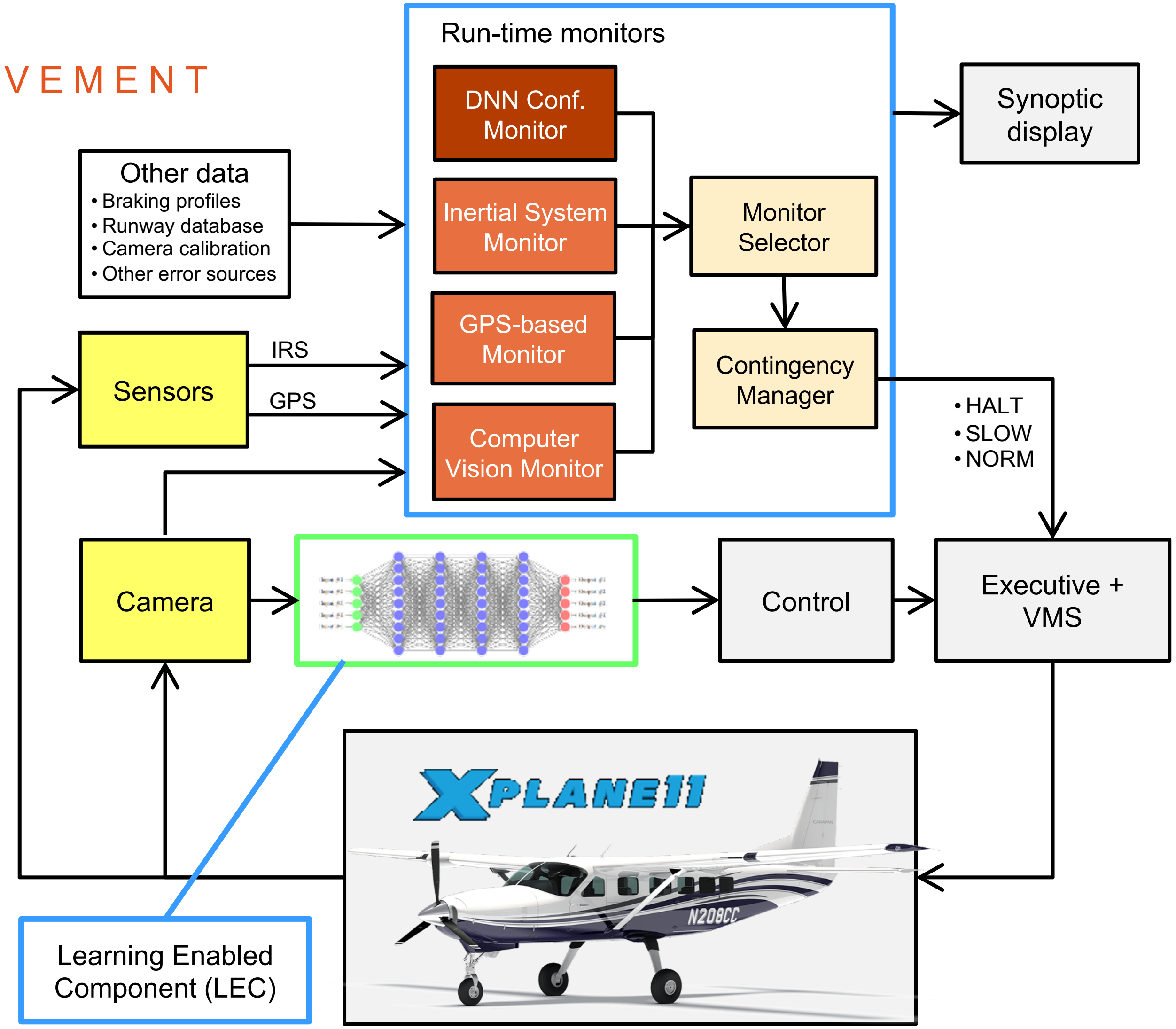

**Potetial problem:**

- **Can we actually define monitors and safety backup that are less complex (in terms of verification) than LEC?**

# DEMONSTRATION

## AUTONOMOUS AIRCRAFT SURFACE MOVEMENT

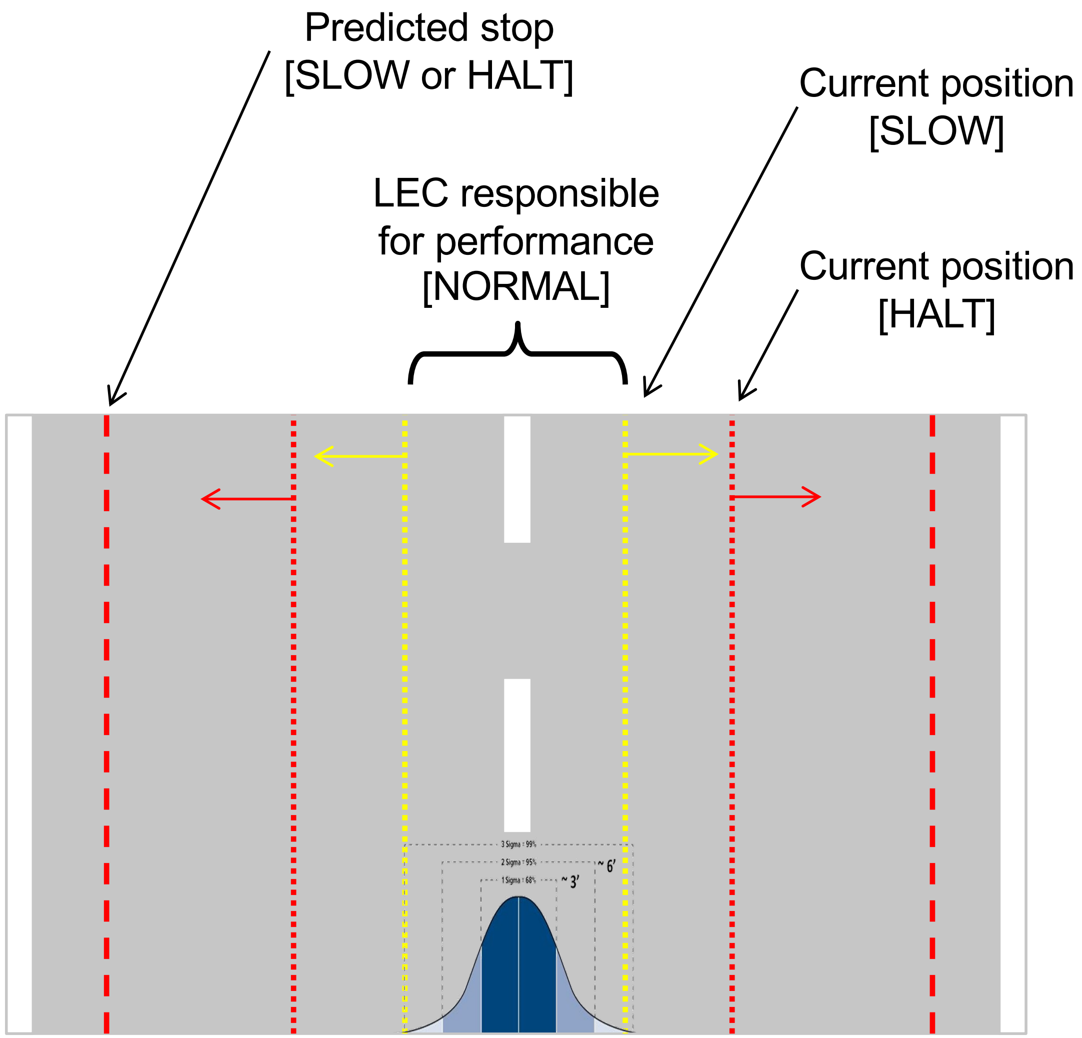
- LEC estimates runway/taxiway centerline position based on camera images to guide steering control
- Ensure that LEC does not cause violation of aircraft safety requirements
  - Keep aircraft on runway / taxiway
  - Minimize unnecessary stopping on runway
- Do so in a way that provides assurance of correctness
  - Multiple diverse monitors based on traditional verified (or verifiable) algorithms
  - Continually select monitor with highest confidence estimate
  - Synthesize monitor selector and contingency manager from formal specifications with proof of correctness



# RTA COMMANDS

## PERFORMANCE VS. SAFETY

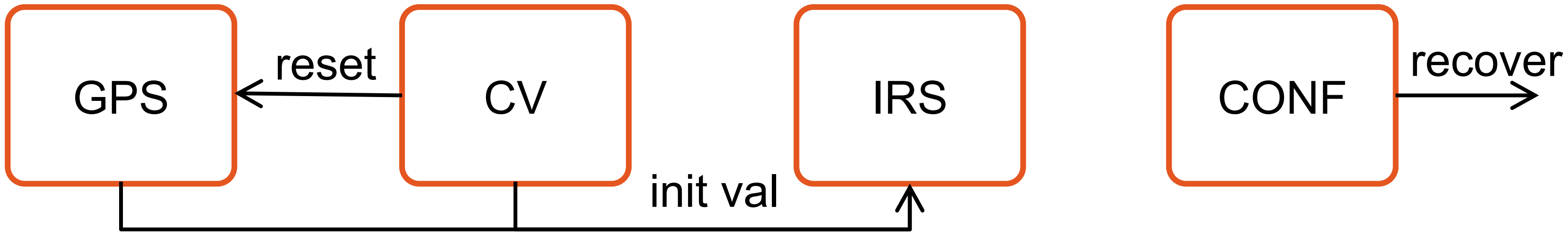
- NORMAL / SLOW / HALT
- SLOW speed command reduces stopping distance and allows more time for
  1. LEC to improve its estimate
  2. Monitor uncertainty to decrease
- Reduces unnecessary stopping on the runway



# RUN-TIME MONITORS

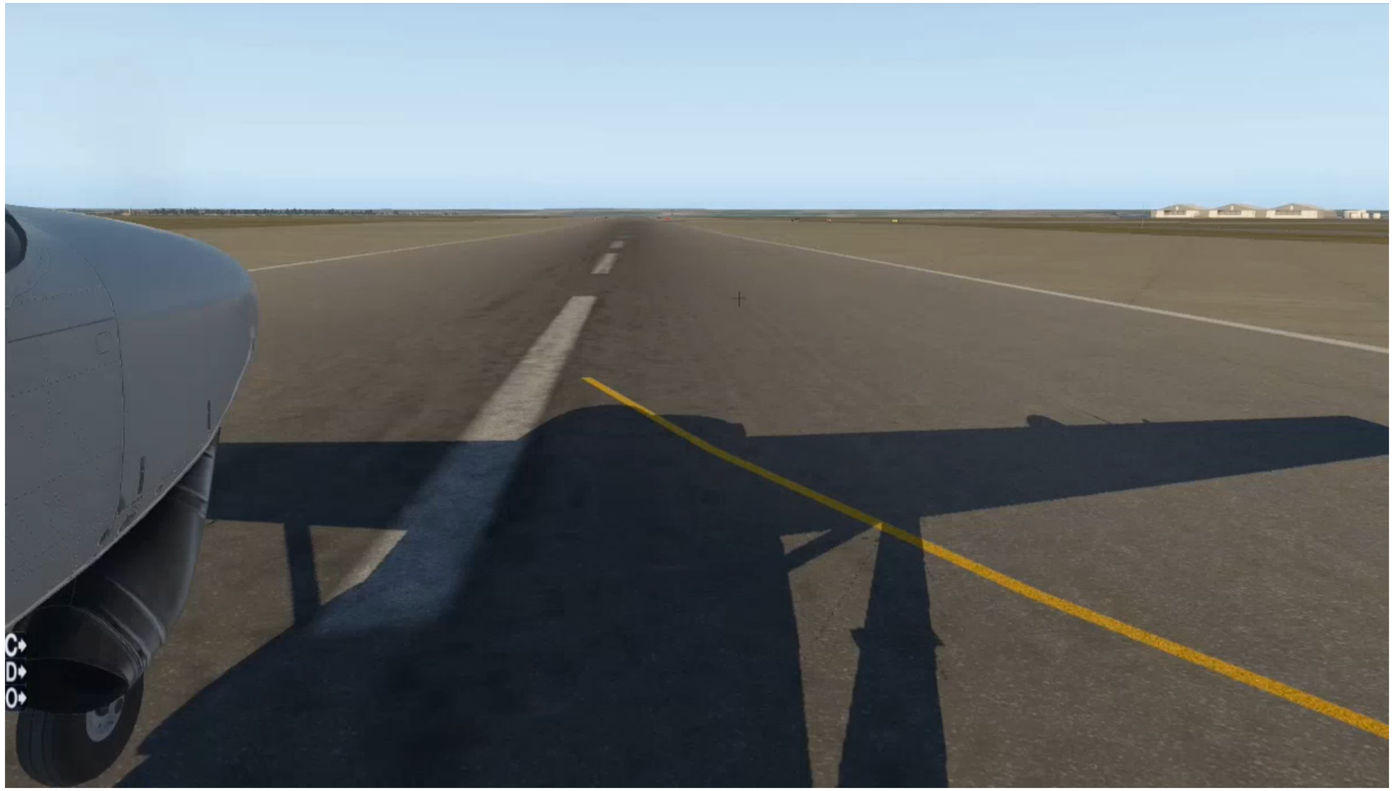
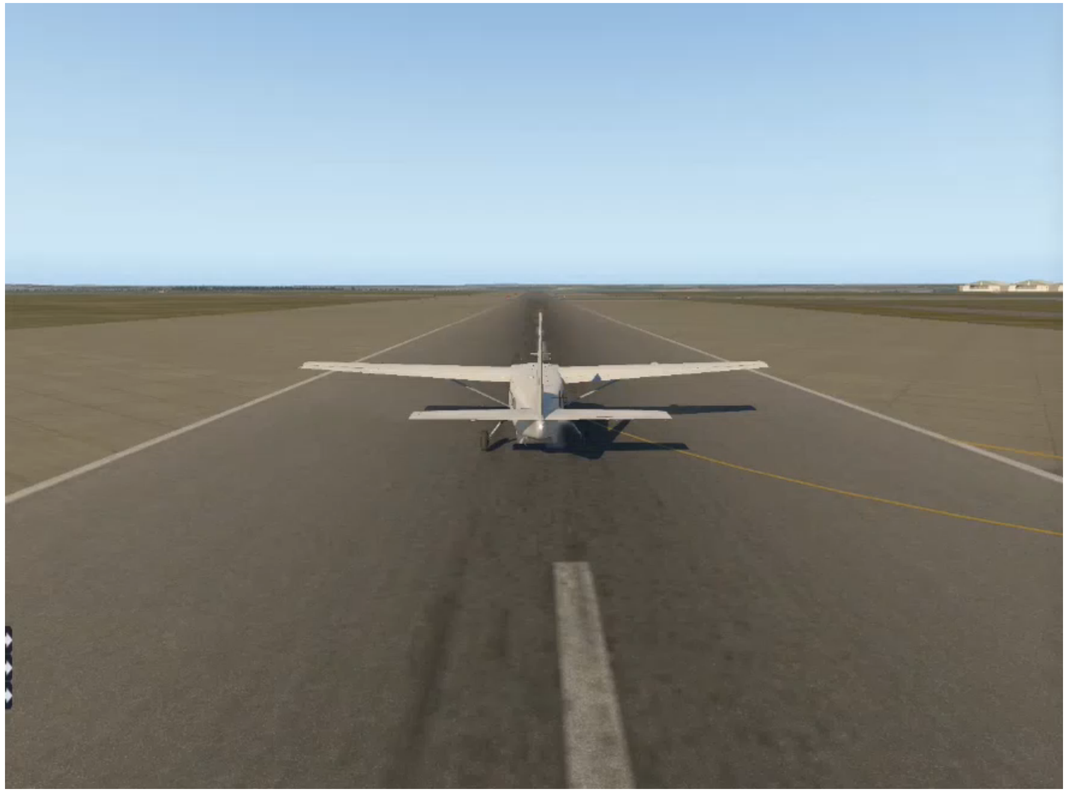
## DIVERSE MONITORS TO MAXIMIZE AVAILABILITY

- **GPS monitor:** Estimate Cross-Track Error (CTE) by integrating GPS velocity signal
  - High performance, preferred estimate
- **Computer Vision (CV) monitor:** Estimate CTE by detecting center line (edge/pattern detection)
  - Use if GPS unavailable or if GPS error > CV error
  - Use CV CTE estimate to reset GPS position
- **IRS monitor:** Estimate CTE by integrating acceleration measurements
  - Use if both GPS and CV monitors are unavailable
  - Initialize with best CTE estimate from GPS or CV
- **LEC confidence monitor:** Is LEC input representative of training data?
  - Use to allow recovering from temporary SLOW or HALT interventions



# BEFORE

## MORNING-ONLY TRAINING DATA / TIME = 1600 (AFTERNOON)

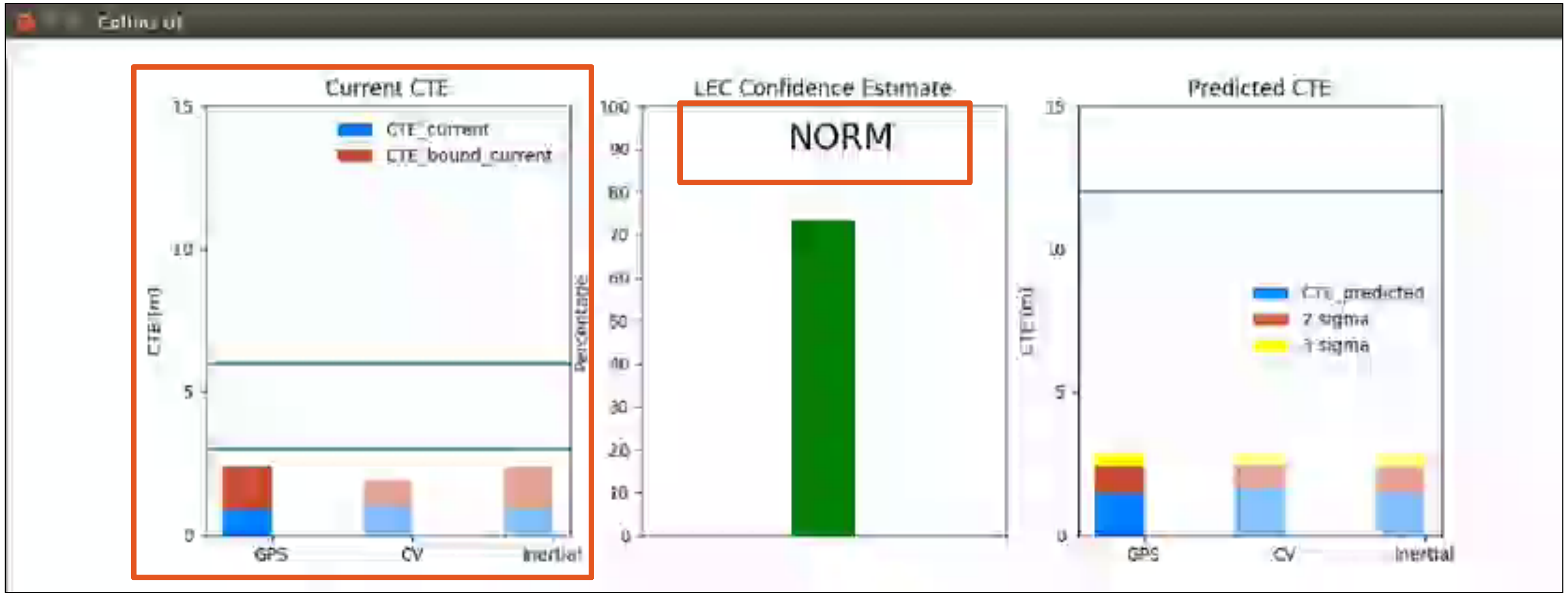



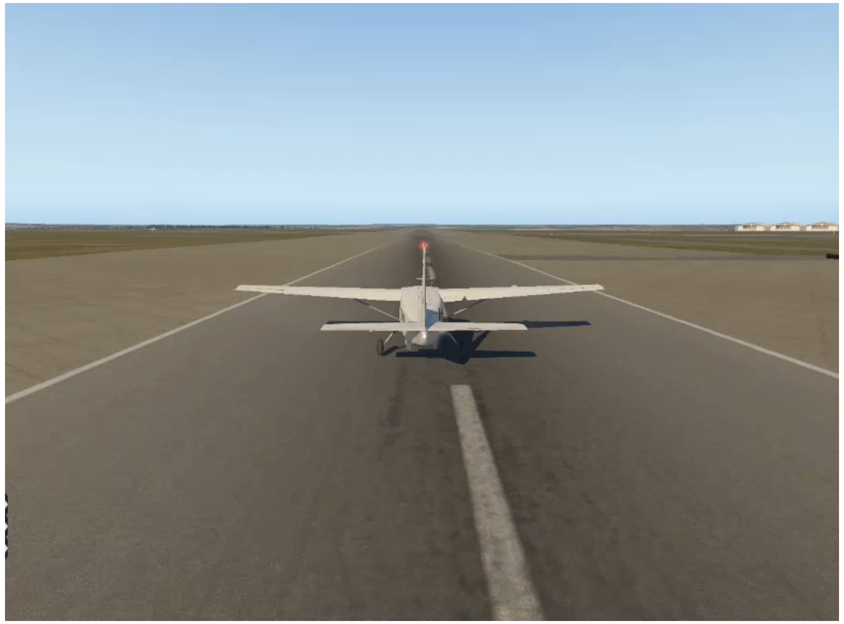
- Intentionally use poorly trained LEC to simulate unsafe/unexpected behaviors

# AFTER

MORNING ONLY
TRAINING DATA
TIME = 1600

Monitor intervenes to maintain safety



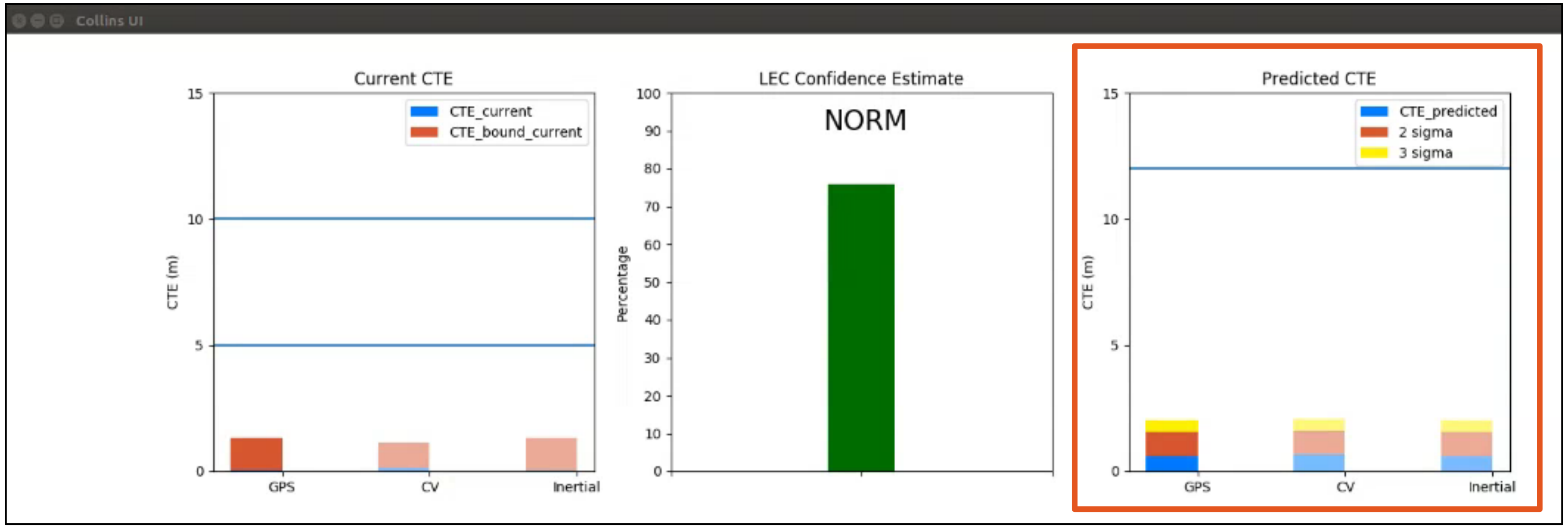



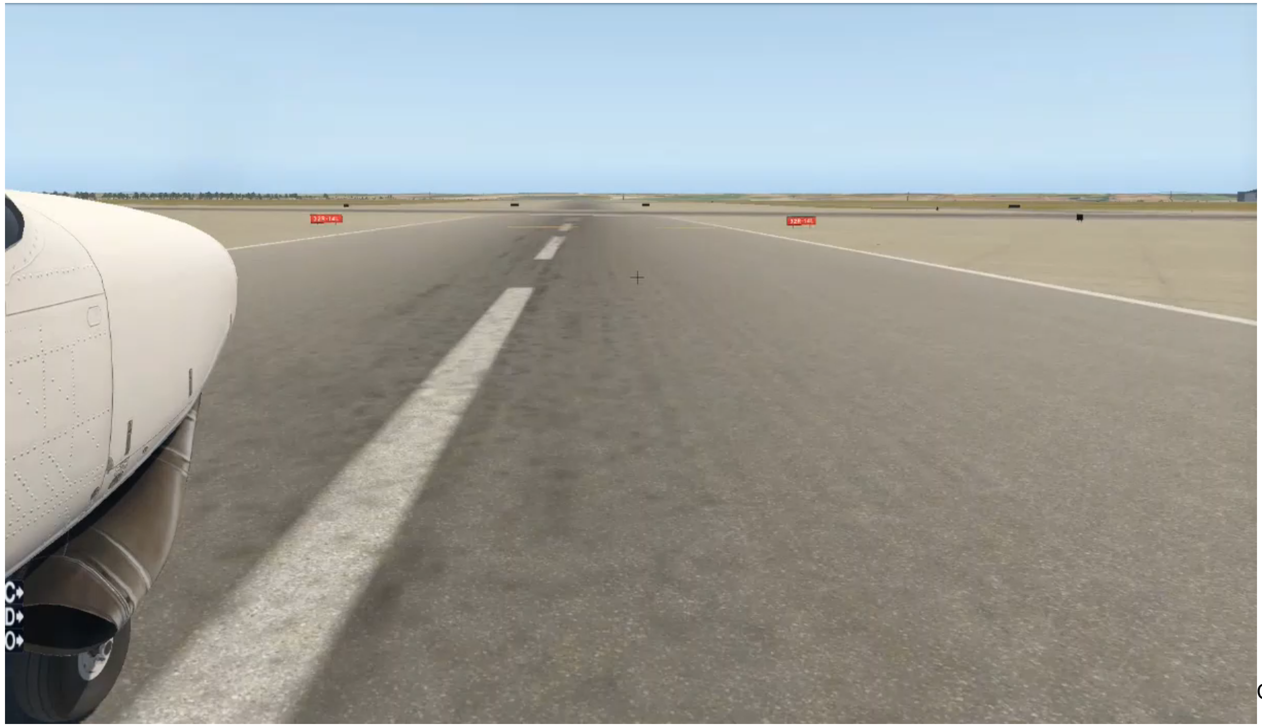
# MORE...

## AFTERNOON ONLY TRAINING DATA TIME = 1200

HALT / SLOW based on *predicted* CTE

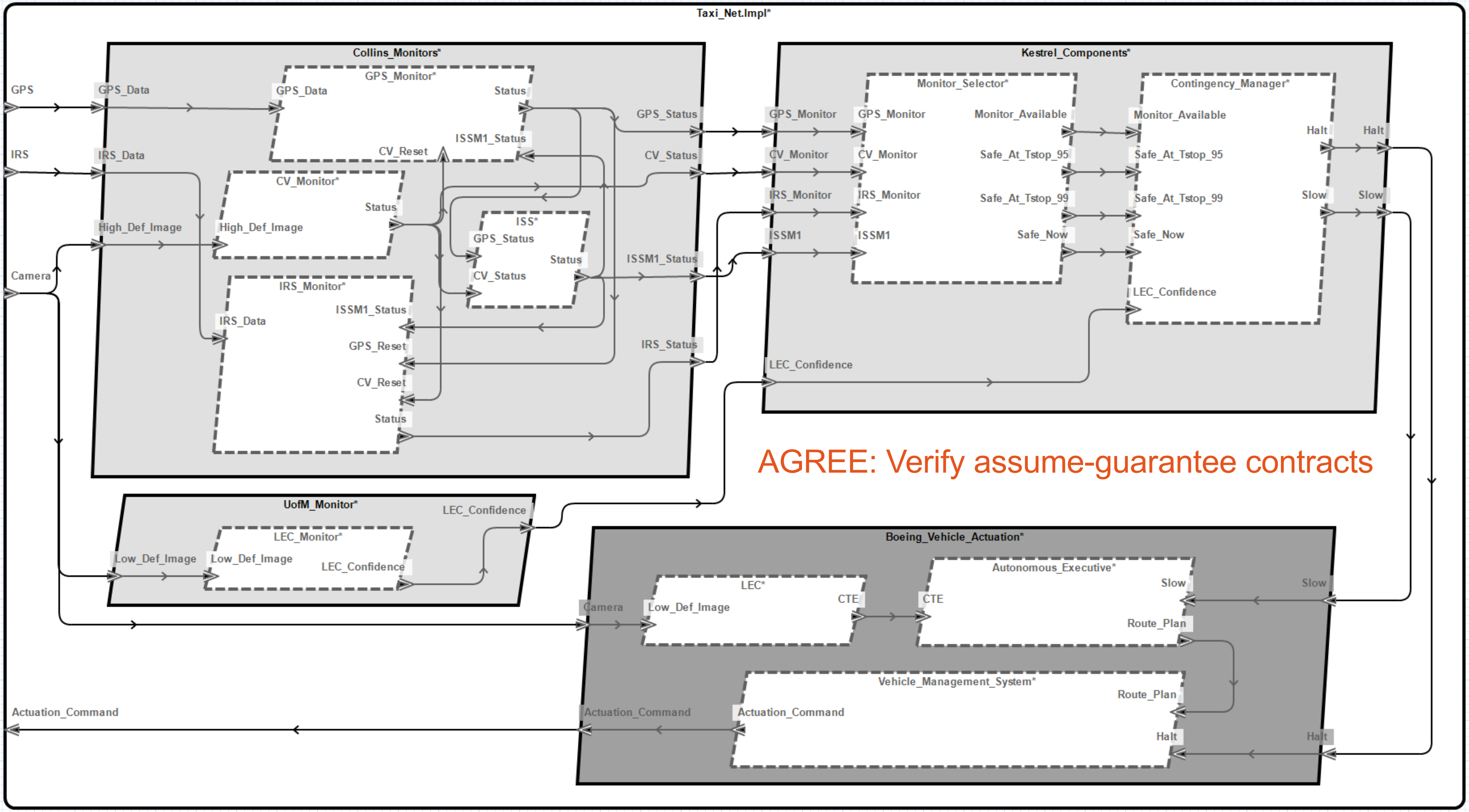
Monitor intervenes to maintain safety



Collins Aerospace

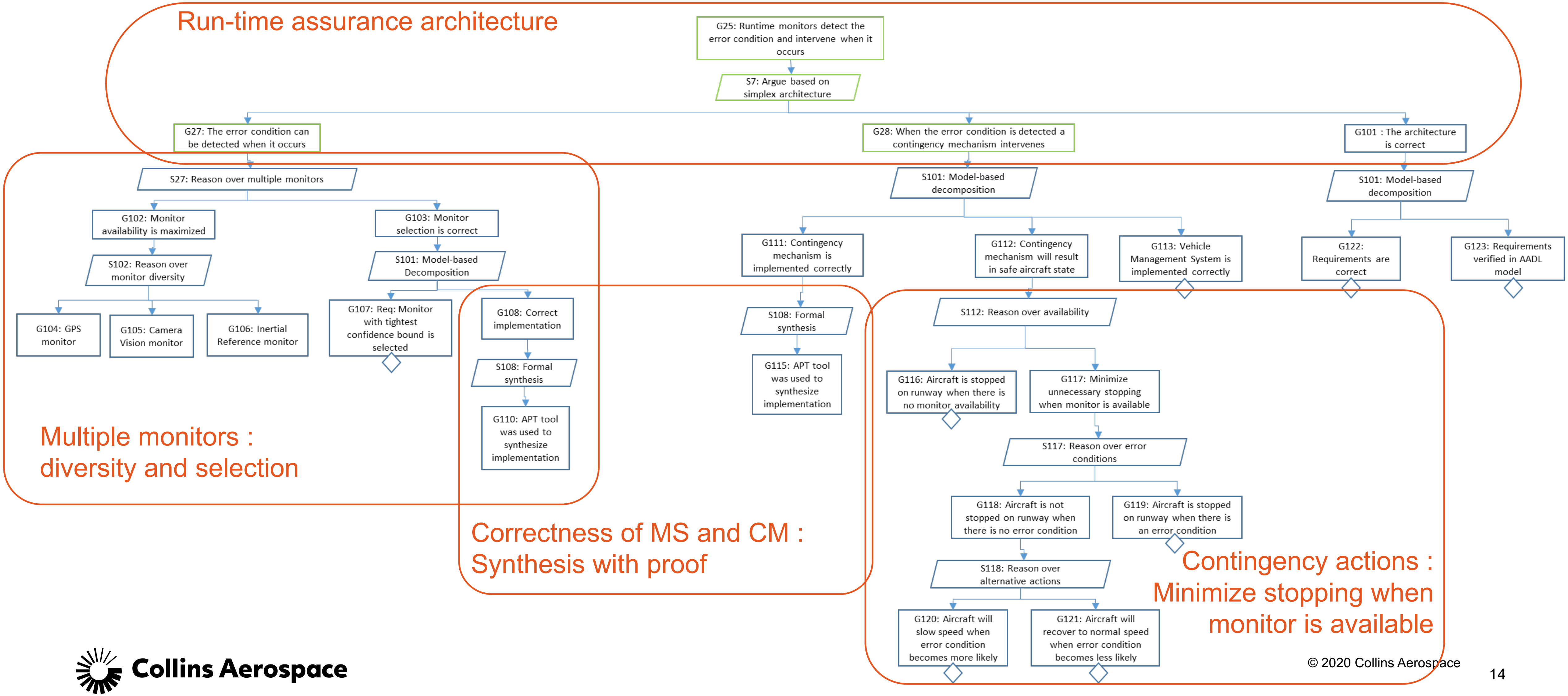
# AADL ARCHITECTURE FOR RUN-TIME ASSURANCE




AGREE: Verify assume-guarantee contracts

# ASSURANCE ARGUMENT

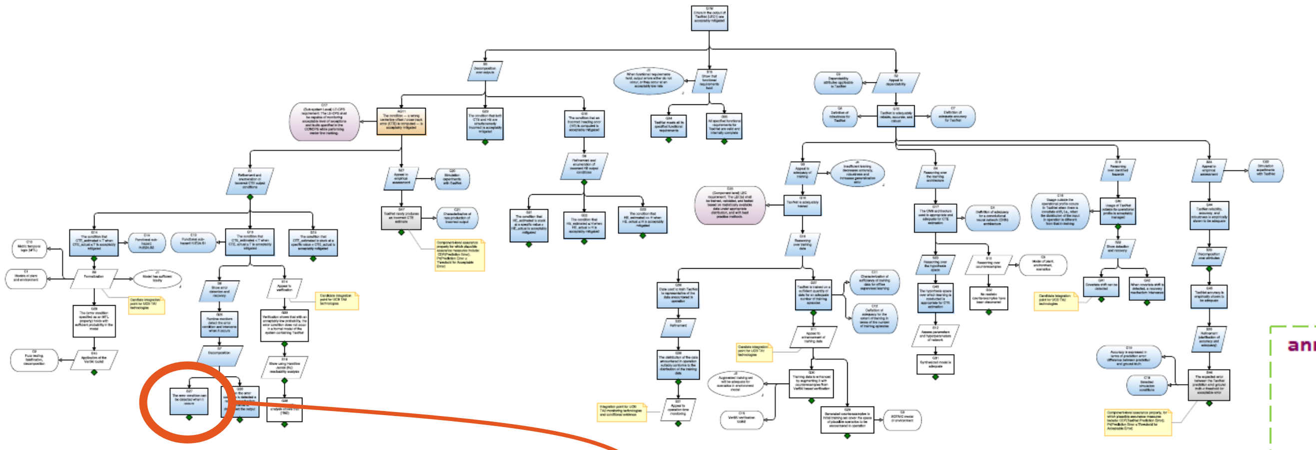
## RESOLUTE TOOL FOR ARCHITECTURAL ASSURANCE CASE

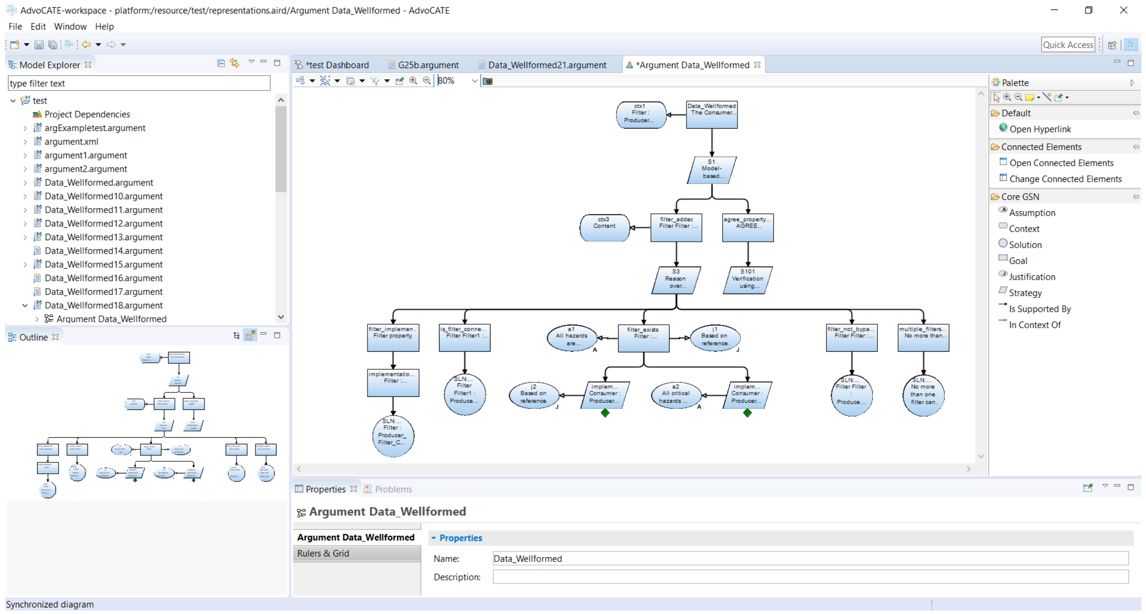



Collins Aerospace

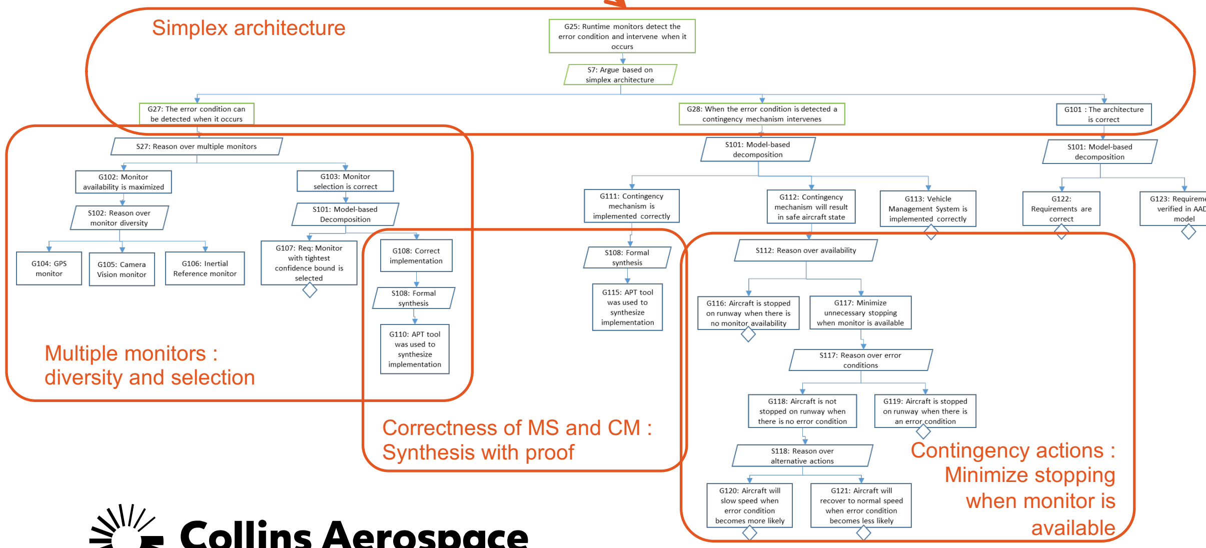
# RESOLUTE APPLICATION

## BOEING CHALLENGE PROBLEM 1.1
## EXPORT TO NASA/SGT ADVOCATE TOOL






```
annex resolute {**

  -- Top-level claim: Runtime monitors detect the error condition and intervene when it occurs
  goal G25(Collins_Monitors : component, Monitor_Selector : component, Contingency_Manager : co
      ** "Runtime monitors detect the error condition and intervene when it occurs" **
      S7(Collins_Monitors, Monitor_Selector, Contingency_Manager, Vehicle_Management_System)

  strategy S7(Collins_Monitors : component, Monitor_Selector : component, Contingency_Manager :
      ** "Argue based on simplex architecture" **
      G27(Collins_Monitors, Monitor_Selector) and G28(Contingency_Manager, Vehicle_Management_S

  goal G27(Collins_Monitors : component, Monitor_Selector : component) <=
      ** "The error condition can be detected when it occurs" **
      S27(Collins_Monitors, Monitor_Selector)

  strategy S27(Collins_Monitors : component, Monitor_Selector : component) <=
      ** "Reason over multiple monitors" **
      G102(Collins_Monitors) and G103(Monitor_Selector)

  goal G28(Contingency_Manager : component, Vehicle_Management_System : component) <=
      ** "When the error condition is detected a recovery mechanism intervenes" **
      strategy S101: "Model-based decomposition";
      G111(Contingency_Manager) and G112() and G113(Vehicle_Management_System)
```

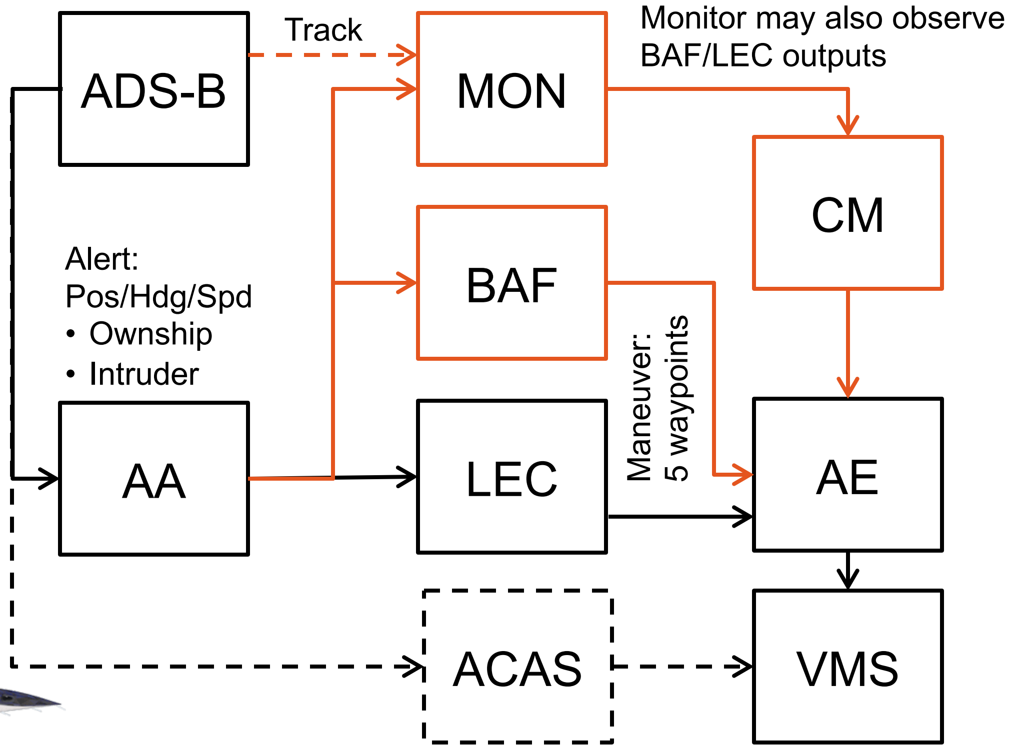



Collins Aerospace

# COLLISION AVOIDANCE

## ASSURED AUTONOMY PHASE 2 CHALLENGE PROBLEM

- Assurance goal:
  - Ensure required separation (“stay well clear”) given assumptions about traffic behavior
- Develop RTA architecture and system verification
- Generate LEC test cases based on sequential inputs
- Verify LEC properties, closed-loop safety
- Assurance case integrating static and dynamic evidence




Additional challenge problems:
- Landing / go around decision
- Take-off / reject decision

Loonwerks

Code, papers, videos available at:

**Loonwerks.com/projects/aahaa.html**

Aahaa!

DARPA

ARCHITECTURE AND ANALYSIS
FOR HIGH-ASSURANCE AUTONOMY

Collins Aerospace