# Assuring Increasingly Autonomous Systems in Human-Machine Teams: An Urban Air Mobility Case Study

**S. Bhattacharyya, J. Davis, N. Narayan, A. Gupta and M. Matessa**

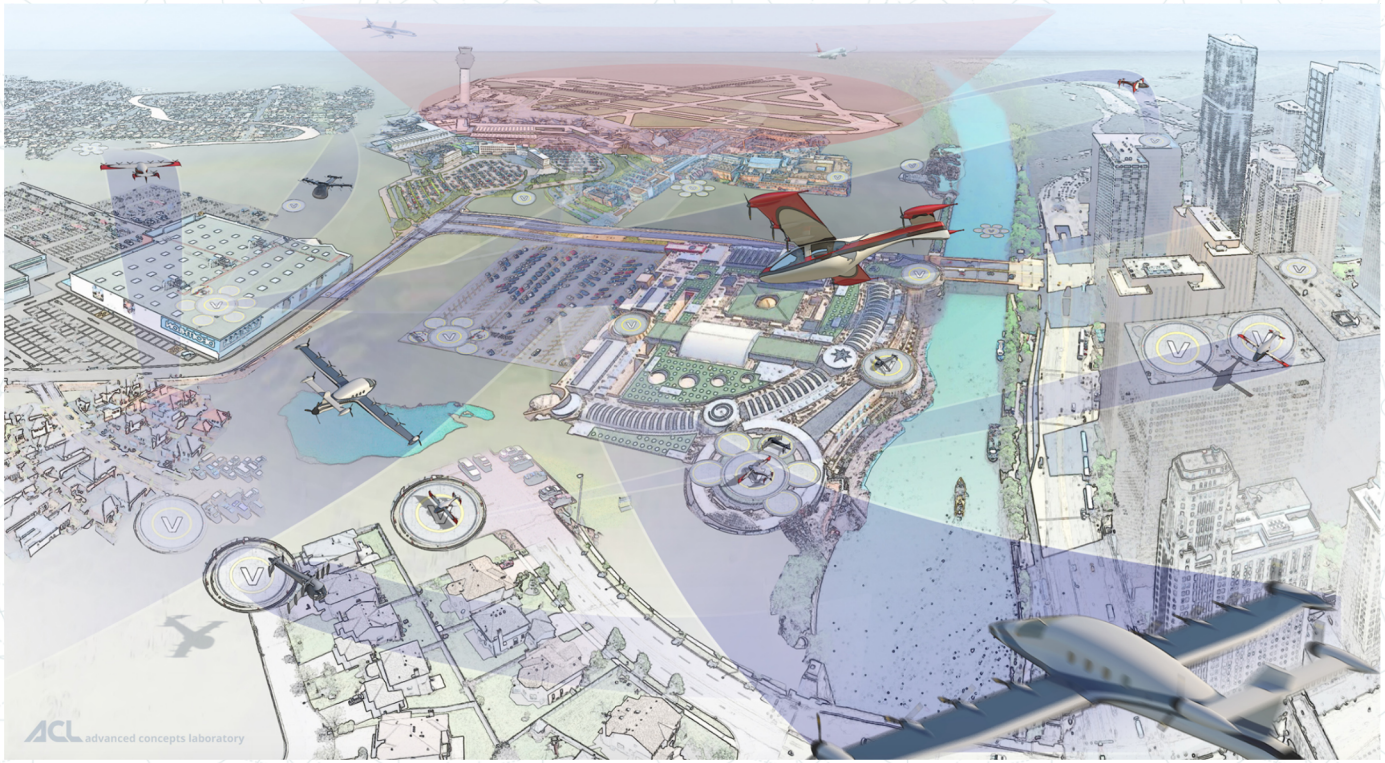
**Formal Methods for Autonomous Systems**
**October 2021**



# Motivation

- Autonomous agents are controlling or coordinating autonomous systems in collaboration with humans to execute day-to-day activities
  - Mission Critical (Autonomous delivery, Urban Air Mobility)
  - Safety Critical (Medical, Aerospace)
  - Security Critical (Cyber space)
- As the human-machine role allocation changes, we need to characterize the human contribution to safety to identify and address new potential failure modes
- Autonomous agents need to be rigorously analyzed and assured to build trust
  - Satisfaction of requirements
  - Correctness of design
  - Boundaries of performance

# NASA, Uber to Explore Safety, Efficiency of Future Urban Airspace




ASSISTLab
Assured Safety, Security and Intent with Sytematic Tactics

Collins Aerospace

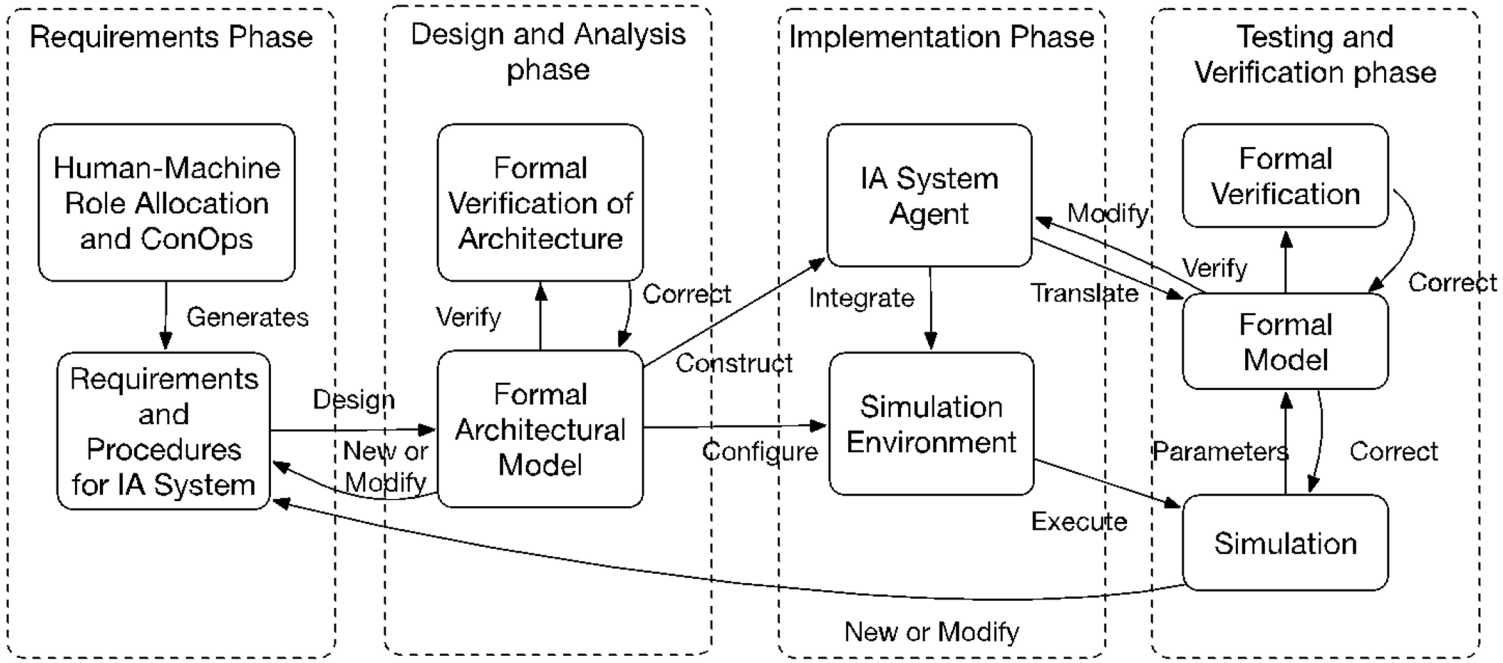
FLORIDA TECH

https://www.nasa.gov/press-release/nasa-uber-to-explore-safety-efficiency-of-future-urban-airspace

## Research Challenges

- What are the requirements and responsibility for increasingly autonomous systems (IAS)?
  - Objectives (contingency management)
  - Human machine interaction
- How to check the correctness of the requirements for Human-IAS interface early?
  - Architecture analysis and design
- What modeling paradigm to implement the IAS behaviors in?
  - Cognitive architecture (Soar, ACT-R)
- Where to simulate the IAS behavior?
  - Xplane
- How to verify the IAS Agent behavior?
  - Translation into formal environment

# Methodology



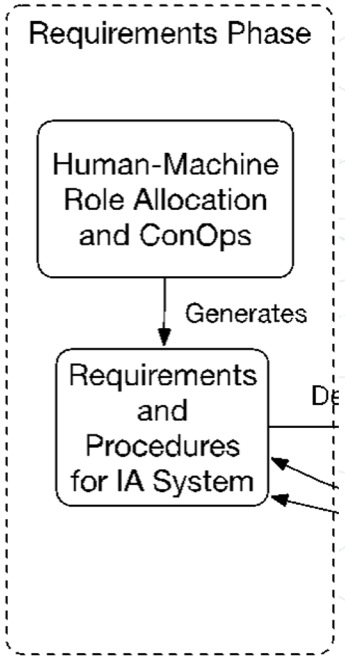

ASSISTLab
Assured Safety, Security and Intent with Sytematic Tactics

Collins Aerospace

FLORIDA TECH

## Requirements from Scenarios

- Scenarios provide roles & responsibilities for the pilot and IA system (IAS)
- CRM provides interaction rules between pilot and IAS
- These responsibilities and interaction rules generate requirements and procedures



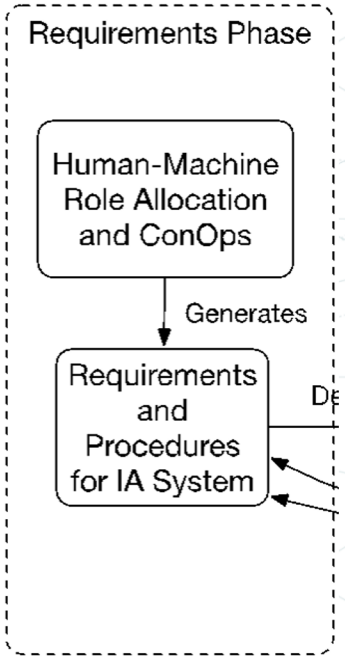

ASSISTLab
Assured Safety, Security and Intent with Sytematic Tactics

Collins Aerospace

FLORIDA TECH

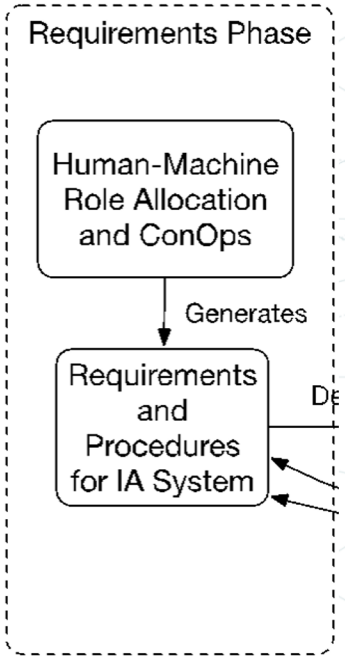
## Scenarios

- Unreliable sensor
  - IAS detects unreliable sensor
  - Pilot decides whether or not to ignore sensor
- Aborted landing
  - Pilot detects unsuitable landing area
  - IAS provides reroute options
  - Pilot decides on reroute plan



## CRM

- Communication
  - Rule: Wait for acknowledgement
    - Ensures other crew member has information
- Management
  - Rule: Human pilot decides on action
    - Pilot in Command listens to crew but has final decision authority



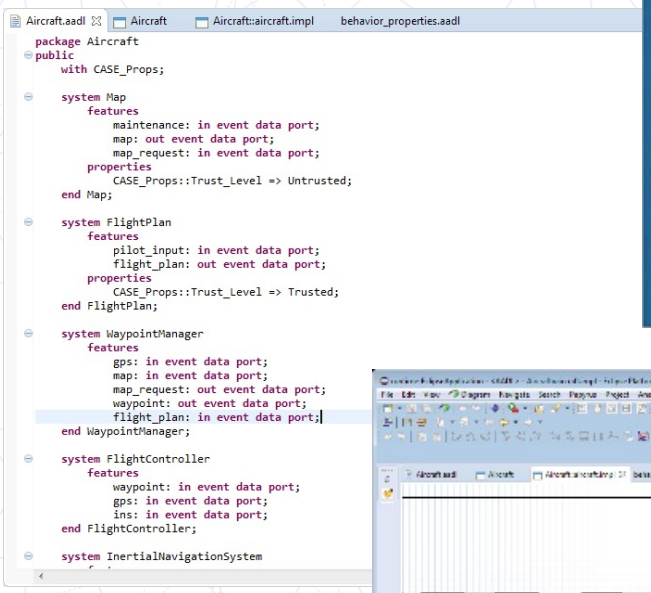

ASSISTLab
Assured Safety, Security and Intent with Sytematic Tactics

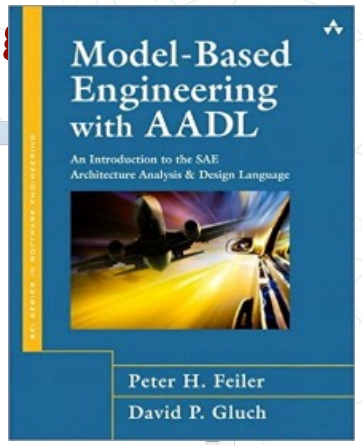
Collins Aerospace

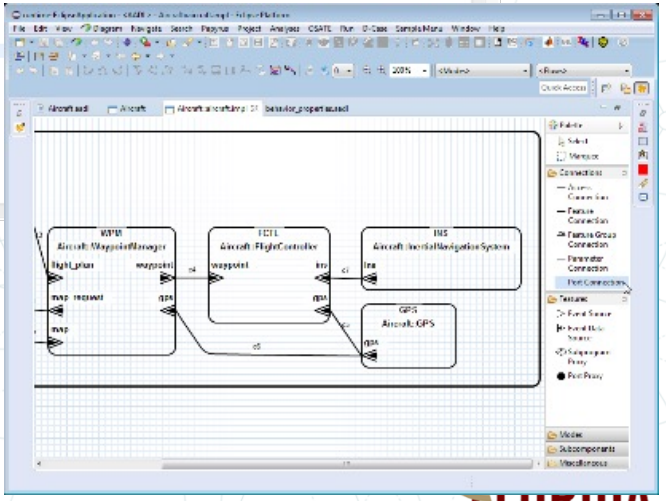
FLORIDA TECH

# Motivation Architecture Analysis and Design Lan (AADL)

- SAE AS5506 standard
- Embedded, real-time, distributed systems
- Physical hardware
  - processors, buses, memory, devices
- Application software
  - software functions, data, threads, processes
- Extendable syntax (annex)
- Open source tools, supported by SEI
  - Open Source AADL Tool Environment (OSATE)



- Sufficiently rigorous semantics to support analysis
- Correct level of abstraction (supports construction)
- Syntax allows addition of new capabilities

FLORIDA TECH SOARTECH

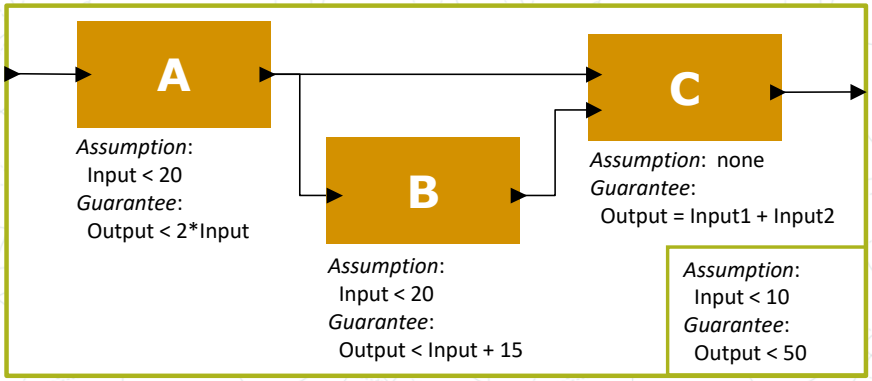


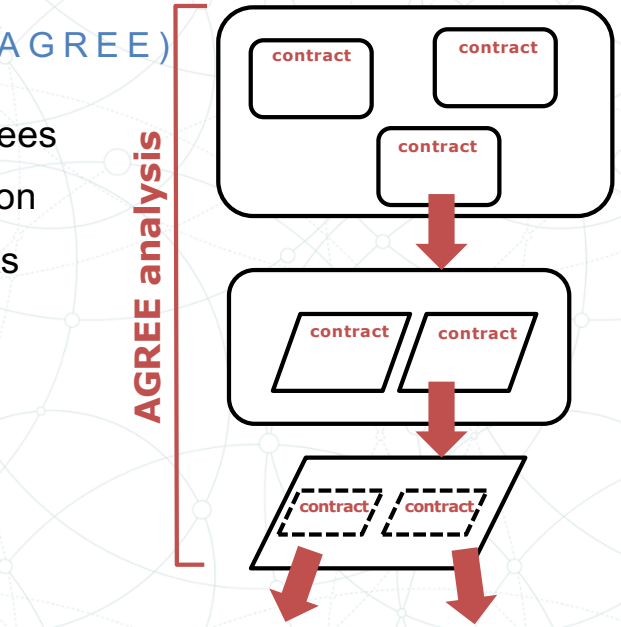
FLORIDA TECH

# Compositional Reasoning

## ASSUME GUARANTEE REASONING ENVIRONMENT (AGREE)

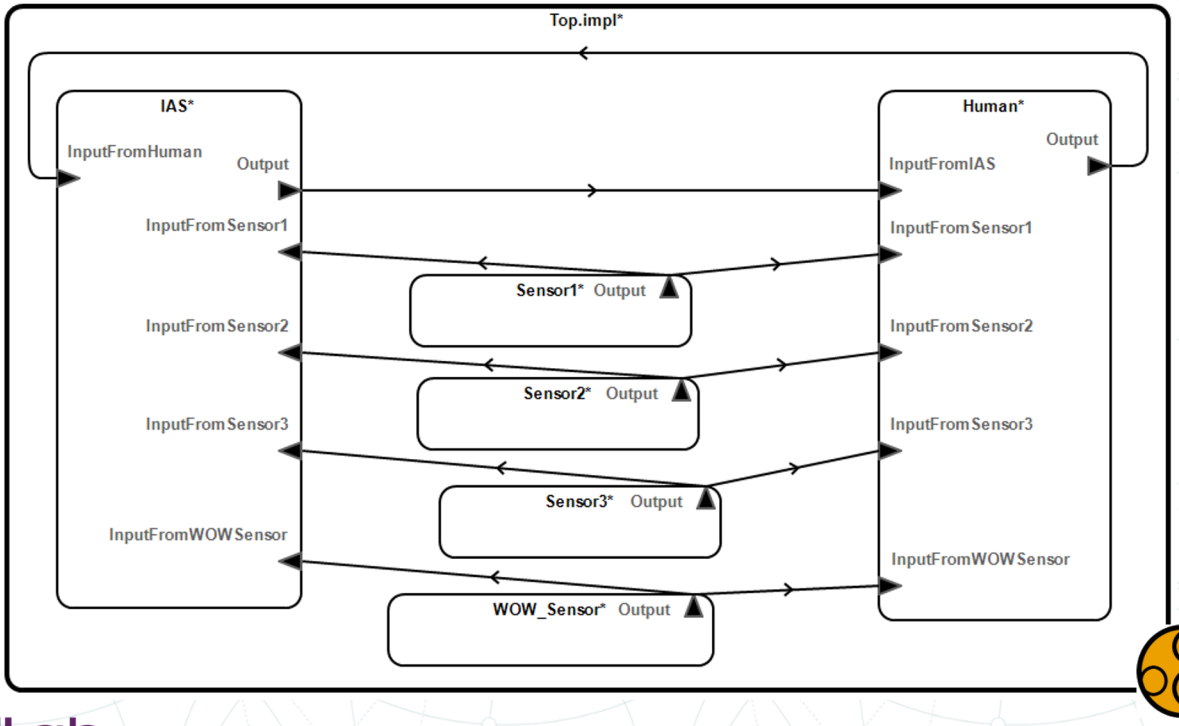
- Each subsystem has a contract consisting of assumptions and guarantees
- The contract of a component abstracts the behavior of its implementation
- Contracts at each layer must be satisfied by contracts of its components
- Leaf component contracts must be satisfied by implementation
- Compositional analysis provides **scalability**

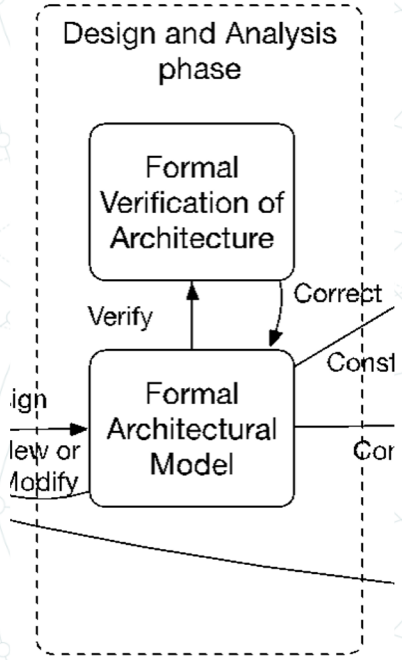


# Formal Architectural Model for the IAS-Human Team

We modeled the architecture in the Architecture Analysis and Design Language (AADL) in the Open Source AADL Tool Environment (OSATE).



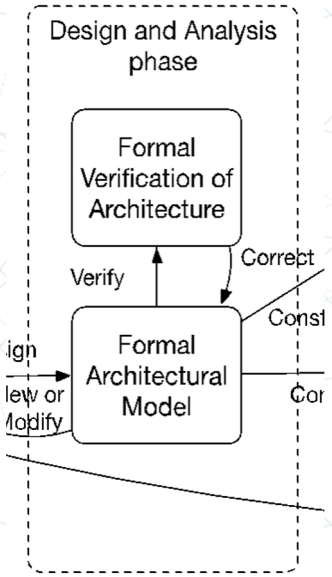

ASSISTLab
Assured Safety, Security and Intent with Sytematic Tactics

Collins Aerospace

FLORIDA TECH

# Formal Architectural Model: Assume-Guarantee Contracts

Assumptions and guarantees for each component are captured in the Assume Guarantee REasoning Environment (AGREE), an open-source plug-in to OSATE for compositional reasoning.




**Selected IAS Requirement: Active Sensor Determination**

```
guarantee "The active sensor should stay the same unless the operator agrees that it is
    unreliable and there is another reliable sensor available.":
            Output.Active_Sensor =
                    if (previous_recommended_sensor = NIL)
                        then previous_active_sensor
                    else if ((previous_active_sensor = 1 and
                            InputFromHuman.Sensor1_Unreliable_Response = enum(Response, Agree))
                                or (previous_active_sensor = 2 and
                                 InputFromHuman.Sensor2_Unreliable_Response = enum(Response, Agree))
                                or (previous_active_sensor = 3 and
                                 InputFromHuman.Sensor3_Unreliable_Response = enum(Response, Agree)))
                        then previous_recommended_sensor
                    else previous_active_sensor;
```

This IAS requirement is mapped to nuXmv for verification over the IAS agent implementation.

ASSISTLab
Assured Safety, Security and Intent with Sytematic Tactics

Collins Aerospace

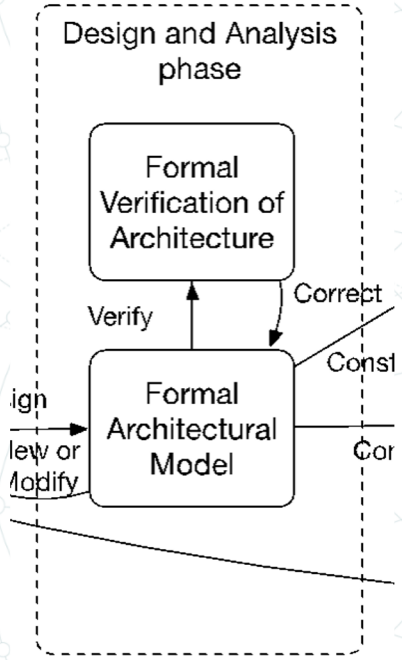
FLORIDA TECH

# Formal Verification of the Architecture

Team properties are specified and formally verified with AGREE *assuming* that the human operator and IAS satisfy their assume-guarantee contracts.

**Selected IAS-Human Team Property:**

```
lemma "If Sensor 1 is both the active sensor and unreliable, it must be the case that
        either the pilot disagreed with the IAS assessment
            or the sensor just became unreliable on this timestep
            or there was no reliable sensor available on the previous timestep":
                ((IAS.Output.Active_Sensor = 1) and not
                    IAS.Output.Sensor1_Reliable)
                    =>  ((Human.Output.Sensor1_Unreliable_Response
                        = enum(Response, Disagree))
                        or prev(IAS.Output.Sensor1_Reliable, true)
                        or not prev(reliable_sensor_available, true));
```



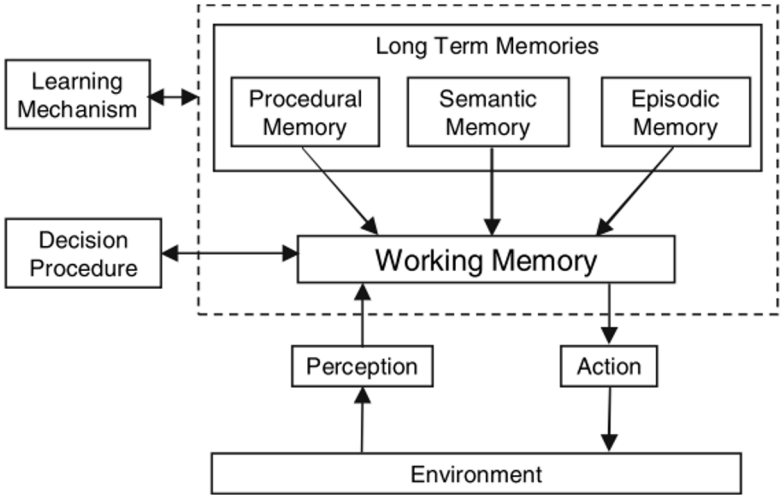

ASSISTLab
Assured Safety, Security and Intent with Sytematic Tactics

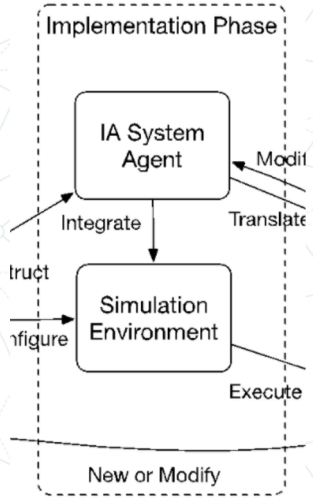
Collins Aerospace

FLORIDA TECH

# IAS Behavior Implementation: Soar a Cognitive Architecture

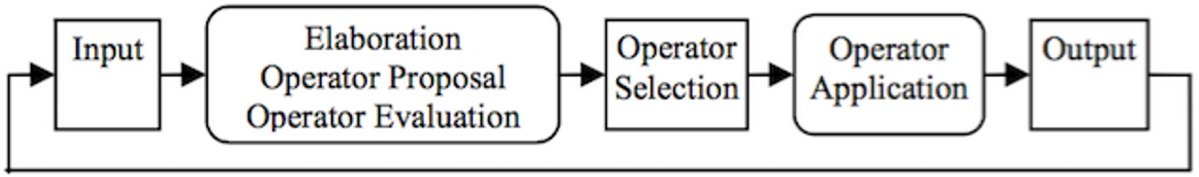
- Main components
  - Perception
  - Production systems (decision procedures)
  - Memory
  - Learning Mechanisms

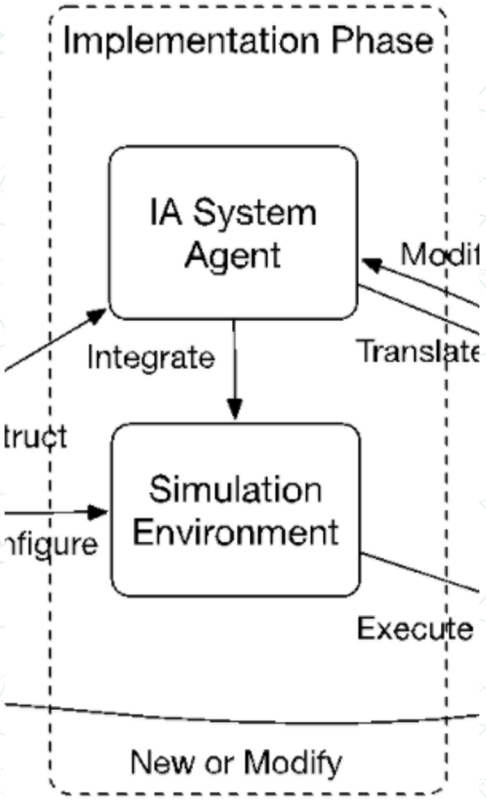



Ref: https://www.researchgate.net/figure/The-Soar-cognitive-architecture-adapted-from-Lehman-et-al-2006_fig7_225257926

# Soar Processing Cycle



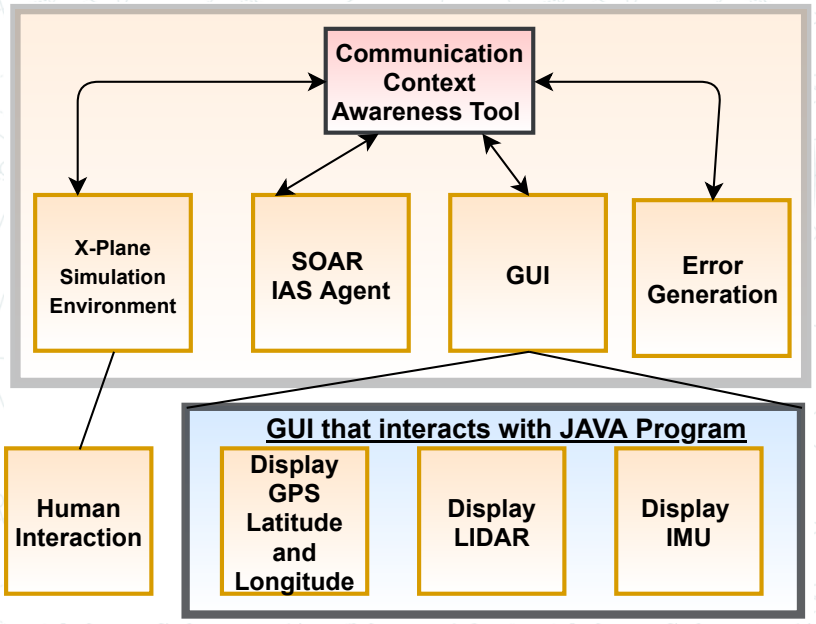



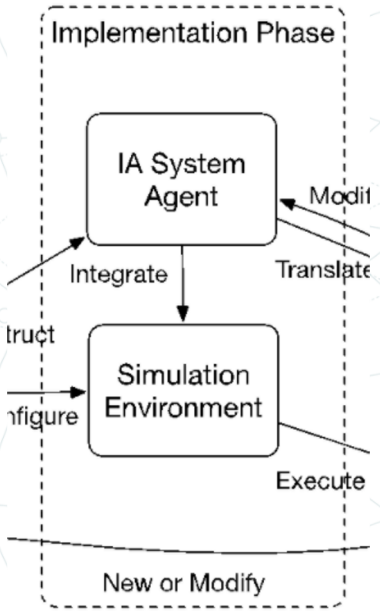

ASSISTLab
Assured Safety, Security and Intent with Sytematic Tactics

Collins Aerospace

FLORIDA TECH

# Simulation Architecture



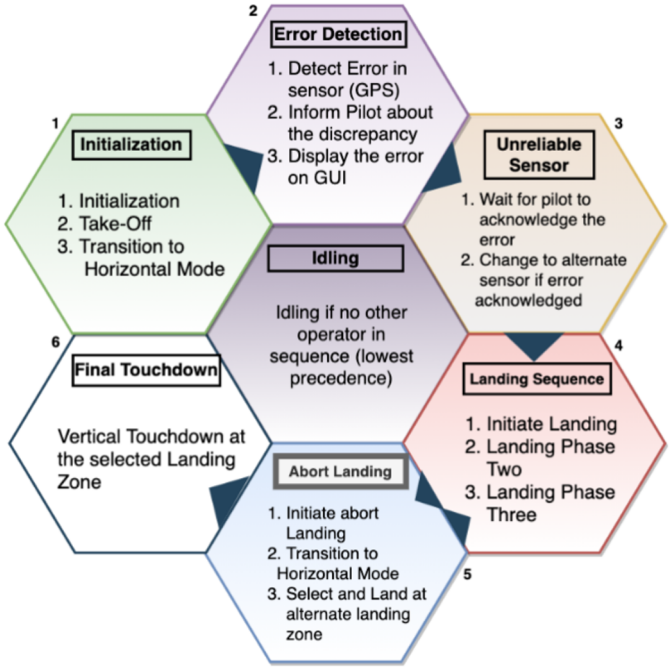



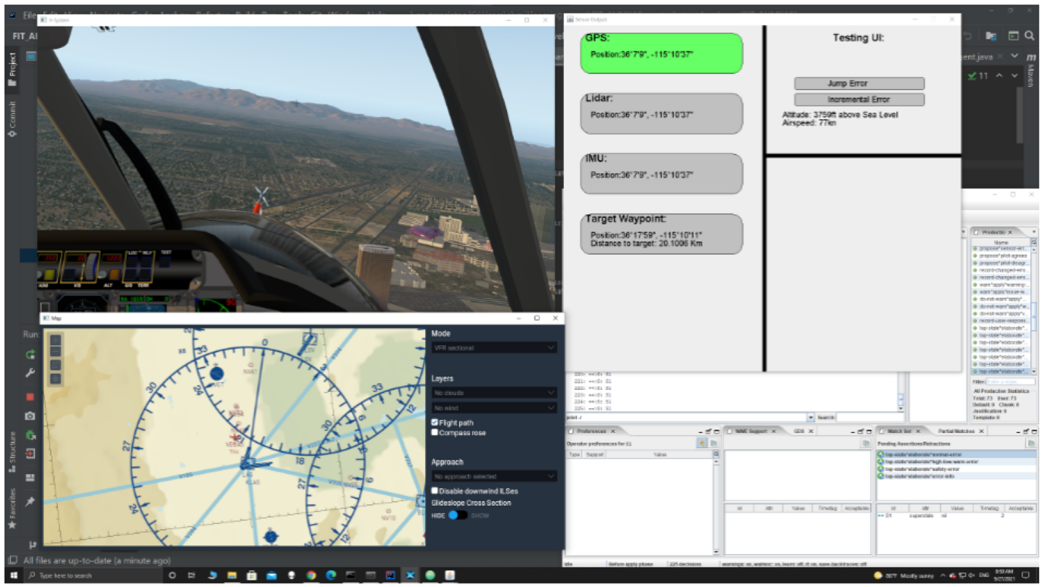

ASSISTLab
Assured Safety, Security and Intent with Sytematic Tactics

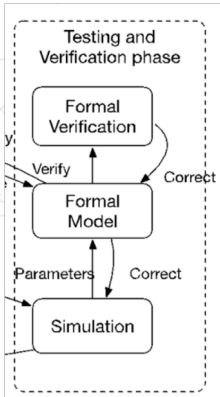
Collins Aerospace

FLORIDA TECH

# Simulation Overview




ASSISTLab
Assured Safety, Security and Intent with Sytematic Tactics

Collins Aerospace

FLORIDA TECH

# Translation for Formal Verification (Soar to nuXmv)

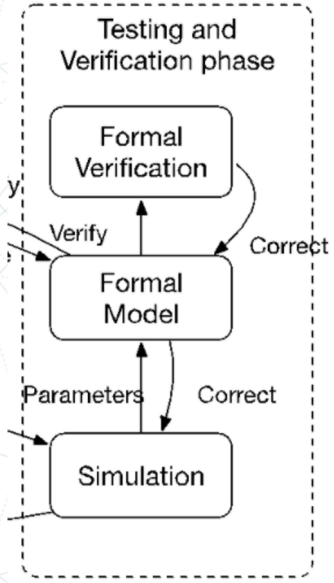
**IAS Agent Soar Takeoff Rule**

sp {propose*takeoff
(state < s > ^name takeoff)
(< s > ^flight-mode vertical)
(< s > ^io.input-link.flightdata < fd >)
(< fd > ^throttle < th > < 0.9)
−− >
(write (crlf) —Throttle — < th >)
(< s > ^ operator < o > +)
(< o > ^ name takeoff) }

**nuXmv Model for the Takeoff Rule**

VAR soarAgent : soar-Rules(state_superstate, state_operator_name, state_name, state_flight-mode, state_io_throttle, state_io_altitude, state_io_airspeed, state_sensor-unreliable, state_io_sensor-error, state_io_initiate-landing, state_landing, state_io_distance-to-target, state_io_abort-landing, state_io_target-altitude, state_io_autoflaps, state_io_air-brake, state_io_target-speed);
TRANS
next (state_operator_name) =
case
(state = run & state_name=takeoff & state_flight-mode=vertical & state_io_throttle<0.9): takeoff;
TRUE : state_operator_name;
esac;

nuXmv
- A symbolic model checker
- Verification algorithms
- SMT based model checking



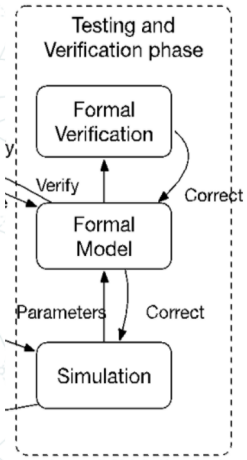

ASSISTLab
Assured Safety, Security and Intent with Sytematic Tactics

Collins Aerospace

FLORIDA TECH

# Simulation Architecture Category of Properties for Formal Verification

- *Reachability*: **LTLSPEC F q, in future q holds**, where q can be (state_io _altitude $>$ 10000) or (state_operator_name = transition)
- *Invariants*: **INVARSPEC q, invariant q** is satisfied, where q can be (state_io_throttle $<=$ 1.0)
- *Handling off nominal situations*: **LTLSPEC F(X p$-$ $>$q (next p leads to q)) or (p U q) (p until q) or (p S q) (p since q)**, where p$-$ $>$q can be of the form, counter_detect_transition_to_lidar $<=$ 5 $-$ $>$ state_io_sensor-to-use = lidar)
- *Handling normal operations*: **LTLSPEC F(X p$-$ $>$q) or (p U q) or (p S q)** , where p$-$ $>$q can be of the form, state_operator_name = state_io_throttle $<$ 1.00 $-$ $>$ state_flight-mode = horizontal)
- *Response to occurrence of event* **LTLSPEC G(p $-$ $>$F q), Globally p leads to q in future**, where p is (state_operator_name = gps-sensor-error-over-limit) and q is (state_sensor-unreliable = yes)




ASSISTLab
Assured Safety, Security and Intent with Sytematic Tactics

Collins Aerospace

FLORIDA TECH

# Simulation Architecture Results and Lessons Learned

- AGREE-to-nuXmv mapping of IAS requirements
- Interaction delay
- Variable type declaration
- Separation of responsibilities
- Mapping from architecture to implementation

| No. | Error Type | Findings |
|---|---|---|
| 1. | IAS Design Error | Soar agent missed response to human agree - disagree rule, as it was being handled at the CCAT interface. |
| 2. | IAS Design Error | state_operator_name throttle case condition missed the equal logical operator(th $<$ 0.9 & th $>$ 0.9). As a result, the throttle value was exceeding 1.0, which is an error. |
| 3. | IAS Design Error | SOAR agent missed the human selection response after abort landing. |
| 4. | Translation Error | Superstate, a state in Soar before Soar graph is generated only indicates that the Soar graph exits or not, it does not need to be translated, but was translated |
| 5. | IAS Design Error | Soar agent missed a case statement to set abort_landing to "NO" after it has been addressed. |
| 6. | Translation Error | Type of some of the variables were generated as integer, but were used as real, it was detected through properties that proved immediately |
| 7. | IAS Requirements Error | Parentheses/order of operations error with regard to selecting the recommended sensor |
| 8. | Operator "Requirements" Error | Selection of landing option was occurring too late, one time step after the options were ready. |
| 9. | IAS Design Error | IAS does not check that the sensor it is switching to is reliable before recommending a switch to the pilot. This is a divergence from the IAS requirements as specified in AGREE. |

## Conclusion and Continuing Work

- AHMIIAS assurance framework **integrates human-machine interactions in a formal model**
- Approach helped **identify and validate the responsibilities of the IAS and the human**.
- Our approach illustrated integration of simulation and formal verification of an IAS agent
- We identified several errors by using this approach and we captured several lessons learned.
- Our continuing work is exploring how our IAS agent can be extended to include **learning mechanisms** in collaboration with Dr. Randy Jones at SoarTech.
- We are  identifying the **human machine interactions that need to be implemented for a learning system**
- Extend our t**ranslation algorithm and verification approach to accommodate an IAS agent that learns.**

We would like to thank Natasha Neogi and Paul Miner of NASA LaRC for their input on Urban Air Mobility scenarios of interest and for their feedback on safety assessment and verification approaches. This research was funded by NASA.

FLORIDA TECH

FLORIDA'S STEM UNIVERSITY™

# Thank you.