The 4th INTERNATIONAL CONFERENCE ON GREEN AVIATION

Nov. 3rd, 2026. 09:30 ~ 10:00 AM @ Xi'An

ICGA2026
THE 4TH INTERNATIONAL CONFERENCE
ON GREEN AVIATION

AI-ASSISTED ARCHITECTURE-AGNOSTIC AIRCRAFT DESIGN METHODOLOGY

Prof. Kwanjung Yee

Seoul National University
Department of Aerospace Engineering
Aerospace Vehicle Design Laboratory



AVDL
Aerospace Vehicle Design Laboratory

Table of Content

MOTIVATION & OBJECTIVES

ARCHITECTURE-AGNOSTIC DESIGN FOUNDATION

METHODOLOGY FOR AI-OPERABILITY

COMPUTATIONAL REALIZATION: AMADEVS

APPLICATIONS & DEMONSTRATION

CONCLUDING REMARKS



AVDL
Aerospace Vehicle Design Laboratory

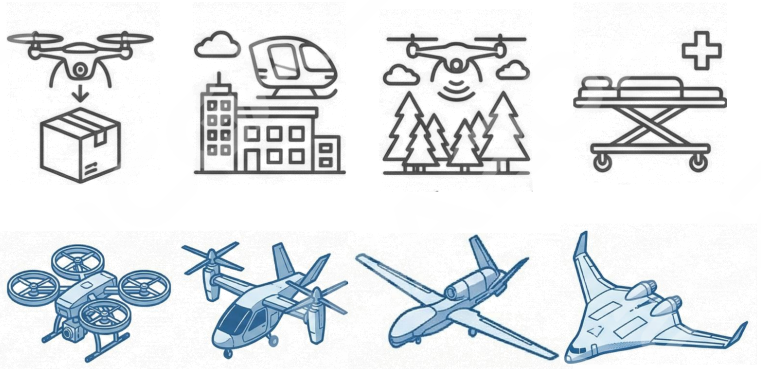


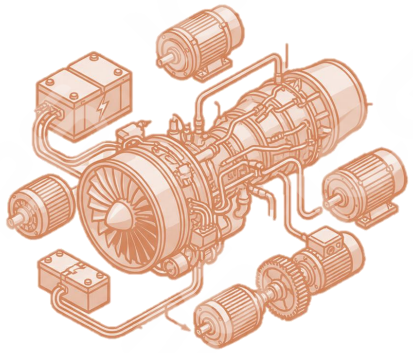
MOTIVATION & OBJECTIVES

MOTIVATION & OBJECTIVES

Aircraft Architecture Space is Expanding

- From Demand Diversification to **Aircraft Architecture Variability**
  - ✓ **Technology paradigm shifts**
    - Electrification, hybrid energy systems, autonomy, etc.
  - ✓ **Market diversification**
    - UAM/AAM, electrified aircraft, etc.
    - Changed war-paradigm, etc.
  - ✓ **Demand for constraint-driven design**
    - Climate impact, sustainability, regulation, etc.







△ Examples of modern aircraft with design complexity & diversity

⇛ **Explosion of aircraft architectures for various needs**

⇛ **Wider design space (at architecture-level) should be explored efficiently.**

MOTIVATION & OBJECTIVES

PROBLEM: Architecture-to-Workflow(MDAO) Coupling

> "A large team would spend around 60-80% of the project time to configure this first version of the executable MDAO system " [1]

- Architecture changes force **structural reconfiguration** of the MDAO workflow[2] (variables, couplings, data flow, solvers)

"Methodology-Level Bottleneck"

Bottleneck 1. Architecture-Workflow Coupling

$$\mathcal{A} \longrightarrow \mathcal{M}_{\mathcal{A}} \longrightarrow \mathcal{S}_{\mathcal{A}} \longrightarrow \mathcal{W}_{\mathcal{A}}$$

Architecture — System Model — Solver Formulation — MDAO Workflow

$$\Delta\mathcal{A} \not\Rightarrow \Delta\text{parameters},$$

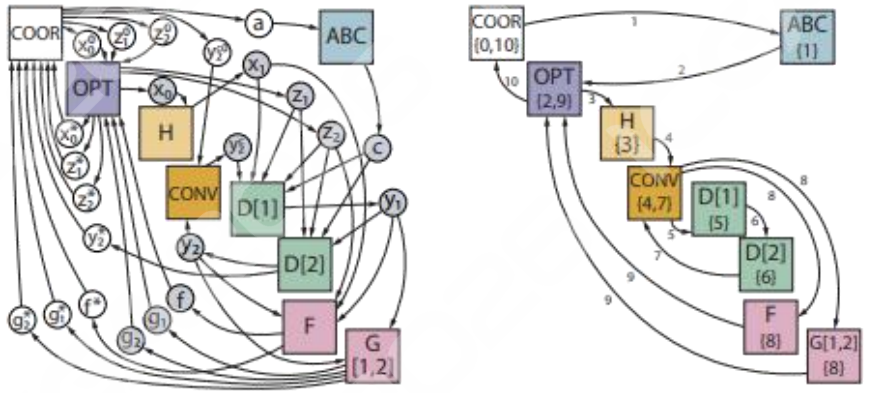
$$\mathcal{A} \to \mathcal{A}' \Rightarrow \text{Reformulation of } \{\mathcal{M}, \mathcal{S}, \mathcal{W}\}_{\mathcal{A}}$$

Changes of architecture — requires changes of model/solver/workflow

- ✓ **Architecture change changes model structure / solver formulation / workflow setup**
  - Examples of system-level solver:
    - **equations of motion (EOM)**
    - **energy balance analysis (EB)**

"Execution-Level Bottleneck"

Bottleneck 2. Intensive Human Labor



a) MDG — Image source: Ref. [1] — b) MPG

- ✓ **What induces repeated human labor:**
  - **Architecture-specific setup / Method and fidelity selection**
  - **Data connection and unit consistency checks**
  - **Convergence failure diagnosis**
  - **Manual repair and re-execution, etc.**

[1] van Gent, I. (2019). Agile MDAO systems: a graph-based methodology to enhance collaborative multidisciplinary design.
[2] Bussemaker, J., Garg, S., & Boggero, L. (2022, September). The influence of architectural design decisions on the formulation of MDAO problems. In *3rd European Workshop on MDO*.

MOTIVATION & OBJECTIVES

Research Objectives

Obj 1.

Develop an Architecture-Agnostic MDAO Foundation

- Reduce architecture-workflow coupling fundamentally.

⇒ How to verify:

- ✓ Apply the same formulation to heterogeneous aircraft architectures
- ✓ Demonstrate trim / energy-flow / sizing workflows without topology-specific solvers

Obj 2.

Develop a methodology for AI-Operability

- Reduce intensive human labor in architecture exploration & MDAO reformulation.

⇒ How to verify:

- ✓ Demonstrate AI-assisted architecture exploration workflow
- ✓ Show reduction of manual setup/diagnosis steps in representative design tasks

Enables Scalable Aircraft MDAO for Arbitrary Architectures

⇒ **Rapid system model & executable-MDAO generation are demonstrated under assistance of AI-Copilot.**

※ *Scalability* here means handling more architectural variants without proportional growth in formulation, wiring, and debugging effort.



GRAPH-BASED ARCHITECTURE-AGNOSTIC FOUNDATION

- System Modeling
- Workflow Modeling

ARCHITECTURE-AGNOSTIC FOUNDATION

Bottleneck 1. **Methodology-Level Solution: Architecture-Decoupled Formulation**

- Separate the architecture, analysis context, and method choices.
- Keep the formulation rules reusable across all of them.

$$\mathcal{U}_{\mathcal{A}}^{\sigma,m} = \Psi_{\mathcal{A}}^{\sigma,m}(\mathcal{A}, \sigma, m)$$

$$\Delta\mathcal{A}, \Delta\sigma, \Delta m \Rightarrow \Delta\Psi_{\mathcal{A}}^{\sigma,m}$$

- ✓ Analysis context change may mutate the vehicle model
- ✓ Solver/fidelity change may modify component setup
- ✓ Architecture change may require workflow reformulation

$$\mathcal{U}_{\mathcal{A}}^{\sigma,\beta} = \Psi(\mathcal{G}_{\mathcal{A}}, \sigma, \beta)$$

$$\Delta\mathcal{A} \Rightarrow \Delta\mathcal{G}_{\mathcal{A}}, \quad \Delta\sigma \Rightarrow \Delta\sigma,$$
$$\Delta m \Rightarrow \Delta\beta, \quad \not\Rightarrow \Delta\Psi$$

- ✓ Architecture is changed through graph artifacts
- ✓ Analysis context is changed through scenario artifacts
- ✓ Method/fidelity is changed through binding artifacts
- ✓ Formulation/execution rule remains reusable

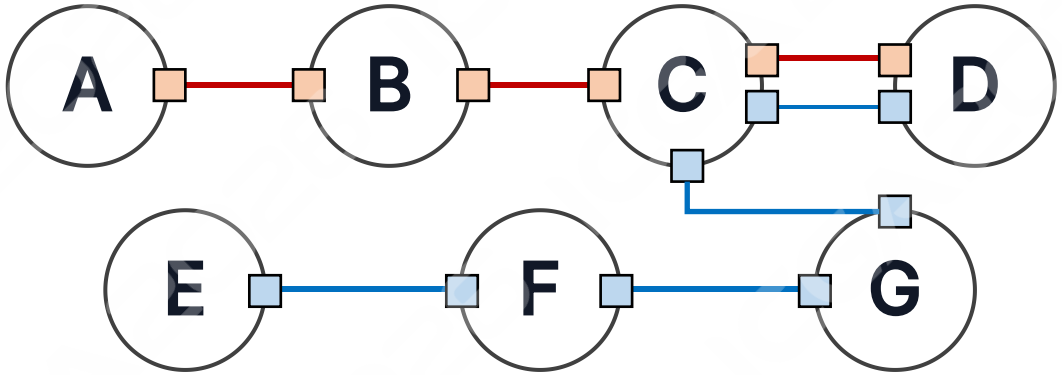
$\mathcal{A}$: aircraft architecture
$\mathcal{G}_{\mathcal{A}}$: graph−based system model for architecture $\mathcal{A}$
σ : analysis scenario or operating context
m : selected solver, method, or fidelity
β : method−binding specification
Ψ : reusable formulation and execution rule
$\mathcal{U}$: executable analysis setup

Methodology-level decoupling localizes changes in explicit formulation artifacts, not in rewritten setup logic.

ARCHITECTURE-AGNOSTIC FOUNDATION

Multi-Relational Graph Representation for Aircraft Architecture

- Not as a list of components with a configuration label on top. But as a multi-relational graph.
- Represents how components are mounted, connected, powered, and controlled as a graph.
- **Nodes :**
  - ✓ physical components
- **Edges :**
  - ✓ Components interactions
    - Assembly / energy-flow / control-signal relations, etc.
- **Ports :**
  - ✓ Interaction interfaces (attached to nodes)
    - *interface contract* for typed edges.
- A graph stores all architecture-defining relationships in one model. ⇒ Single Source of Truth(SSOT)



architecture graph

$$\mathcal{G}_{\mathcal{A}} = (\mathcal{V}_{\mathcal{A}}, \mathcal{E}_{\mathcal{A}}, \mathcal{P}_{\mathcal{A}}, \tau_{\mathcal{A}})$$

$\mathcal{V}_{\mathcal{A}}$: component and system−element nodes

$\mathcal{E}_{\mathcal{A}}$: relationship edges between nodes

$\mathcal{P}_{\mathcal{A}}$: typed ports attached to nodes

A~G $\tau_{\mathcal{A}}$: typing function for nodes, edges, and ports

⇒ **The single graph makes architecture explicit & tracible.**

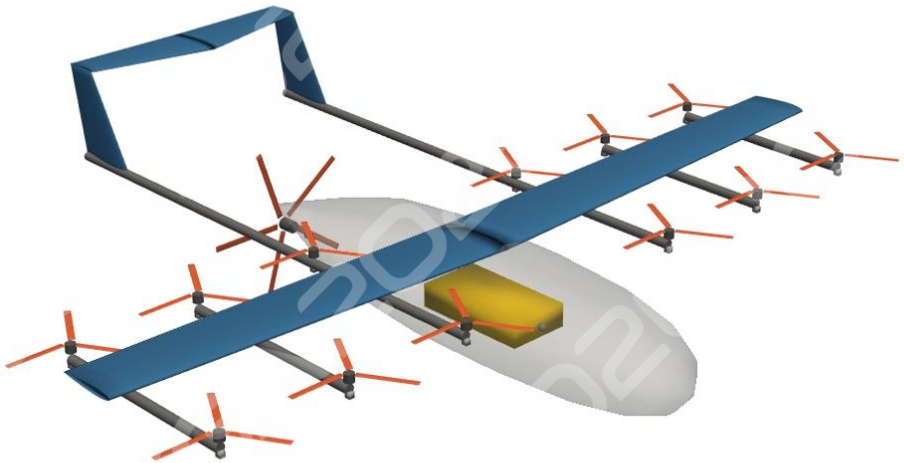
ARCHITECTURE-AGNOSTIC FOUNDATION

Vehicle Graph Modeling Example

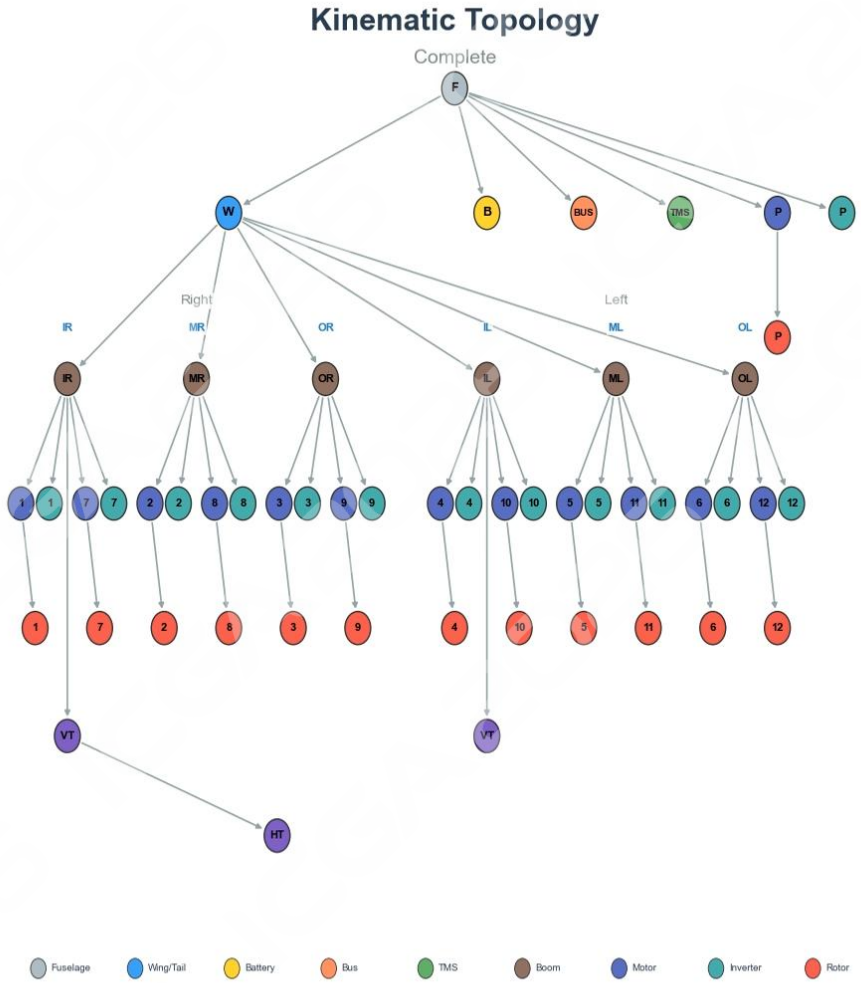
- [video] Full-Battery Lift + Cruise Vehicle Modeling

architecture graph

$$\mathcal{G}_{\mathcal{A}} = (\mathcal{V}_{\mathcal{A}}, \mathcal{E}_{\mathcal{A}}, \mathcal{P}_{\mathcal{A}}, \tau_{\mathcal{A}})$$



Wisk Cora V02





GRAPH-BASED ARCHITECTURE-AGNOSTIC FOUNDATION

- System Modeling
- Workflow Modeling

ARCHITECTURE-AGNOSTIC FOUNDATION

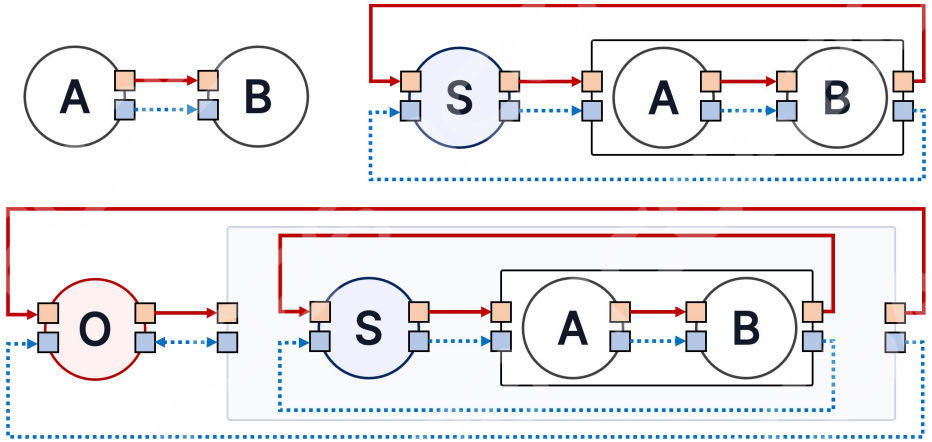
Multi-Relational Graph Representation for Workflow Composition

- Workflow graph : a declarative representation of analysis tasks and dependencies
  - ✓ How analysis tasks are organized, coupled, and iterated.

- **Nodes :**
  - ✓ analysis tasks, solvers, or optimizers

- **Edges :**
  - ✓ Task interactions
    - Data-flow / process-control relations

- **Ports :**
  - ✓ Analysis interfaces (attached to nodes)
    - Input/output *contract* ports



workflow graph

$$\mathcal{G}_W = (\mathcal{V}_W, \mathcal{E}_W, \mathcal{P}_W, \tau_W)$$

$\mathcal{V}_W$: analysis and driver nodes

$\mathcal{E}_W$: typed workflow relations

$\mathcal{P}_W$: typed interfaces attached to workflow nodes

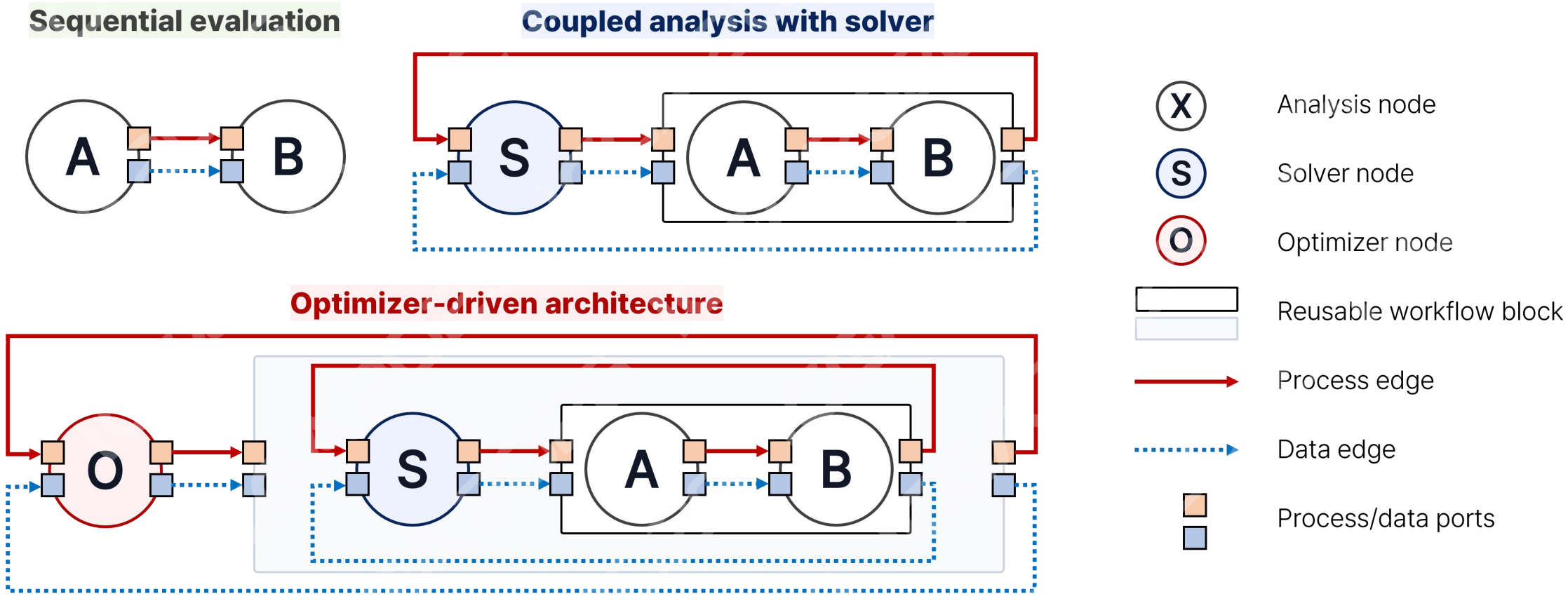
A~C / S / O τ_W : typing function for nodes, edges, and ports

⇒ **The single graph makes workflow composition explicit and traceable.**

ARCHITECTURE-AGNOSTIC FOUNDATION

From Workflow Graph to Flexible MDAO Architecture

- Arbitrary analysis workflows can be composed by declaring tasks and their dependencies.



⇒ MDO architecture becomes configurable at the workflow level, not hard-coded in a procedural script.

Architecture-Agnostic Problem Formulation of EOM & EB

- **Goal: Architecture-dependent information is declared once and consumed by common assembly rules.**

Conventional: $\mathcal{A} \xrightarrow{\mathcal{R}_{\mathrm{EOM},\mathcal{A}},\,\mathcal{R}_{\mathrm{EB},\mathcal{A}}} (\mathcal{P}_{\mathrm{EOM},\mathcal{A}},\,\mathcal{P}_{\mathrm{EB},\mathcal{A}})$

$\Delta\mathcal{A} \Rightarrow \Delta\mathcal{R}_{\mathrm{EOM},\mathcal{A}},\,\Delta\mathcal{R}_{\mathrm{EB},\mathcal{A}}$

Proposed: $(\mathcal{G}_{\mathcal{A}},\,\sigma,\,\beta) \xrightarrow{\mathcal{R}_{\mathrm{EOM}},\,\mathcal{R}_{\mathrm{EB}}} (\mathcal{P}_{\mathrm{EOM},\mathcal{A}},\,\mathcal{P}_{\mathrm{EB},\mathcal{A}})$

$\Delta\mathcal{A} \Rightarrow \Delta\mathcal{G}_{\mathcal{A}} \Rightarrow \Delta\mathcal{P}_{\mathrm{EOM},\mathcal{A}},\,\Delta\mathcal{P}_{\mathrm{EB},\mathcal{A}} \not\Rightarrow \Delta\mathcal{R}_{\mathrm{EOM}},\,\Delta\mathcal{R}_{\mathrm{EB}}$

Equations of Motion

✓ Force and moment balance assembled from graph-declared contributors

- from active roles, controls, methods, and kinematic relations
- Inputs: system graph / analysis scenario (including method bindings)

$$m\dot{\boldsymbol{V}} = \sum_{i\in\mathcal{V}_\sigma^{\mathrm{act}}} \boldsymbol{F}_i + \boldsymbol{W}$$

$$\boldsymbol{I}\dot{\boldsymbol{\omega}} + \boldsymbol{\omega}\times(\boldsymbol{I}\boldsymbol{\omega}) = \sum_{i\in\mathcal{V}_\sigma^{\mathrm{act}}} (\boldsymbol{r}_i \times \boldsymbol{F}_i + \boldsymbol{M}_i)$$

$\mathcal{G}_{\mathcal{A}}$: structured architecture model
σ : analysis scenario
β : method binding
$\mathcal{R}_{\mathrm{EOM}}$, $\mathcal{R}_{\mathrm{EB}}$: reusable EOM and energy-balance formulation rules
$\mathcal{P}_{\mathrm{EOM}}$, $\mathcal{P}_{\mathrm{EB}}$: generated EOM and energy-balance problems

$\mathcal{V}_\sigma^{\mathrm{act}}$: scenario-active contributor set
$\boldsymbol{F}_i, \boldsymbol{M}_i$: force and moment contributed by graph element i
$\boldsymbol{r}_i$: moment arm to contributor i
$\boldsymbol{W}$: weight contribution
σ : analysis scenario selecting active contributors

Architecture-Agnostic Trim Analysis

$$(\mathcal{G},\,\sigma) \longmapsto \left\{\mathcal{V}_\sigma^{\mathrm{act}},\,\mathcal{Z}_{\mathrm{trim}}(\mathcal{G},\sigma)\right\}, \qquad \boldsymbol{z} \in \mathcal{Z}_{\mathrm{trim}}(\mathcal{G},\sigma)$$

$$\boldsymbol{z}_{\mathcal{G},\sigma}^{(k+1)} = \mathcal{S}\left(\boldsymbol{z}_{\mathcal{G},\sigma}^{(k)},\,\boldsymbol{R}_{\mathrm{trim}}\left(\boldsymbol{z}_{\mathcal{G},\sigma}^{(k)};\mathcal{G},\sigma\right)\right)$$

$$\boldsymbol{R}_{\mathrm{trim}}(\boldsymbol{z}_{\mathcal{G},\sigma};\mathcal{G},\sigma) = \begin{bmatrix} \sum_i \boldsymbol{F}_i \\ \sum_i (\boldsymbol{r}_i \times \boldsymbol{F}_i + \boldsymbol{M}_i) \end{bmatrix} = \boldsymbol{0}$$

Input: $\mathcal{G}$, σ

1. $\mathcal{V}_\sigma^{\mathrm{act}} \leftarrow$ activate contributors from σ
2. $\mathcal{Z}_{\mathrm{trim}}(\mathcal{G},\sigma) \leftarrow$ admissible trim-variable set
3. resolve views from $(\mathcal{G},\sigma,\boldsymbol{z})$, $\boldsymbol{z} \in \mathcal{Z}_{\mathrm{trim}}(\mathcal{G},\sigma)$
4. **for each** $i \in \mathcal{V}_\sigma^{\mathrm{act}}$:
   $\mu_i \leftarrow$ force/moment evaluator for contributor i
   $(\boldsymbol{F}_i, \boldsymbol{M}_i) \leftarrow \mu_i(\mathcal{G},\sigma,\boldsymbol{z})$
5. $\boldsymbol{R}_{\mathrm{trim}}(\boldsymbol{z};\mathcal{G},\sigma) \leftarrow$ assemble common-frame loads

Architecture-Agnostic Problem Formulation of EOM & EB

- **Goal: Architecture-dependent information is declared once and consumed by common assembly rules.**

Conventional: $\mathcal{A} \xrightarrow{\mathcal{R}_{\mathrm{EOM},\mathcal{A}},\,\mathcal{R}_{\mathrm{EB},\mathcal{A}}} (\mathcal{P}_{\mathrm{EOM},\mathcal{A}},\,\mathcal{P}_{\mathrm{EB},\mathcal{A}})$

$\Delta\mathcal{A} \Rightarrow \Delta\mathcal{R}_{\mathrm{EOM},\mathcal{A}},\,\Delta\mathcal{R}_{\mathrm{EB},\mathcal{A}}$

Proposed: $(\mathcal{G}_{\mathcal{A}},\,\sigma,\,\beta) \xrightarrow{\mathcal{R}_{\mathrm{EOM}},\,\mathcal{R}_{\mathrm{EB}}} (\mathcal{P}_{\mathrm{EOM},\mathcal{A}},\,\mathcal{P}_{\mathrm{EB},\mathcal{A}})$

$\Delta\mathcal{A} \Rightarrow \Delta\mathcal{G}_{\mathcal{A}} \Rightarrow \Delta\mathcal{P}_{\mathrm{EOM},\mathcal{A}},\,\Delta\mathcal{P}_{\mathrm{EB},\mathcal{A}} \not\Rightarrow \Delta\mathcal{R}_{\mathrm{EOM}},\,\Delta\mathcal{R}_{\mathrm{EB}}$

Energy-Balance

- ✓ Power-balance equations are formulated.
  - from energy roles, methods, and directed flow relations
  - Inputs: system graph / analysis scenario (including method bindings)

$$\boldsymbol{g}_{\mathcal{A}}(\boldsymbol{p}_{\mathcal{A}};\mathcal{N}_{\mathcal{A}},\boldsymbol{d}) = \left[\sum_{j\in\mathcal{P}_{\mathcal{A}}^{-}(i)} P_{j\to i} - \sum_{k\in\mathcal{P}_{\mathcal{A}}^{+}(i)} P_{i\to k} - P_{\mathrm{loss},i}\right]_{i\in\mathcal{V}_{\mathcal{A}}} = \mathbf{0}$$

$\mathcal{G}_{\mathcal{A}}$: structured architecture model
σ : analysis scenario
β : method binding
$\mathcal{R}_{\mathrm{EOM}}$, $\mathcal{R}_{\mathrm{EB}}$: reusable EOM and energy-balance formulation rules
$\mathcal{P}_{\mathrm{EOM}}$, $\mathcal{P}_{\mathrm{EB}}$: generated EOM and energy-balance problems

$\mathcal{V}_{\sigma,\mathrm{en}}^{\mathrm{act}}$: scenario-active energy-network set
$\mathcal{P}_{\mathrm{en}}(\mathcal{G},\sigma,\boldsymbol{d})$: admissible power-flow variable set
$\boldsymbol{d}$: segment demand / prescribed sink demand
g_i : node-level power-balance residual
$\boldsymbol{g}_{\mathrm{en}}$: assembled global energy-balance residual
$\boldsymbol{p}$: power-flow variable vector

Architecture-Agnostic Energy Balance Analysis

$$(\mathcal{G},\,\sigma,\,\boldsymbol{d}) \longmapsto \left\{\mathcal{V}_{\sigma,\mathrm{en}}^{\mathrm{act}},\,\mathcal{P}_{\mathrm{en}}(\mathcal{G},\sigma,\boldsymbol{d})\right\}$$

$$\boldsymbol{g}_{\mathrm{en}}(\boldsymbol{p}_{\mathcal{G},\sigma,\boldsymbol{d}};\mathcal{G},\sigma,\boldsymbol{d}) = \left[\,g_i\,\right]_{i\in\mathcal{V}_{\sigma,\mathrm{en}}^{\mathrm{act}}} = \mathbf{0}$$

Input: $\mathcal{G}$, σ, $\boldsymbol{d}$
1. $\mathcal{V}_{\sigma,\mathrm{en}}^{\mathrm{act}} \leftarrow$ activate energy nodes/links from σ
2. $\mathcal{P}_{\mathrm{en}}(\mathcal{G},\sigma,\boldsymbol{d}) \leftarrow$ admissible power-flow variable set
3. resolve energy-network and demand views from $(\mathcal{G},\sigma,\boldsymbol{d},\boldsymbol{p})$
4. **for each** $i \in \mathcal{V}_{\sigma,\mathrm{en}}^{\mathrm{act}}$:
   $\eta_i \leftarrow$ power-balance evaluator for energy node i
   $g_i \leftarrow \eta_i(\mathcal{G},\sigma,\boldsymbol{d},\boldsymbol{p})$
5. $\boldsymbol{g}_{\mathrm{en}} \leftarrow$ assemble node residuals into the global energy-balance residual
6. update $\boldsymbol{p}$ until $\boldsymbol{g}_{\mathrm{en}}$ closes

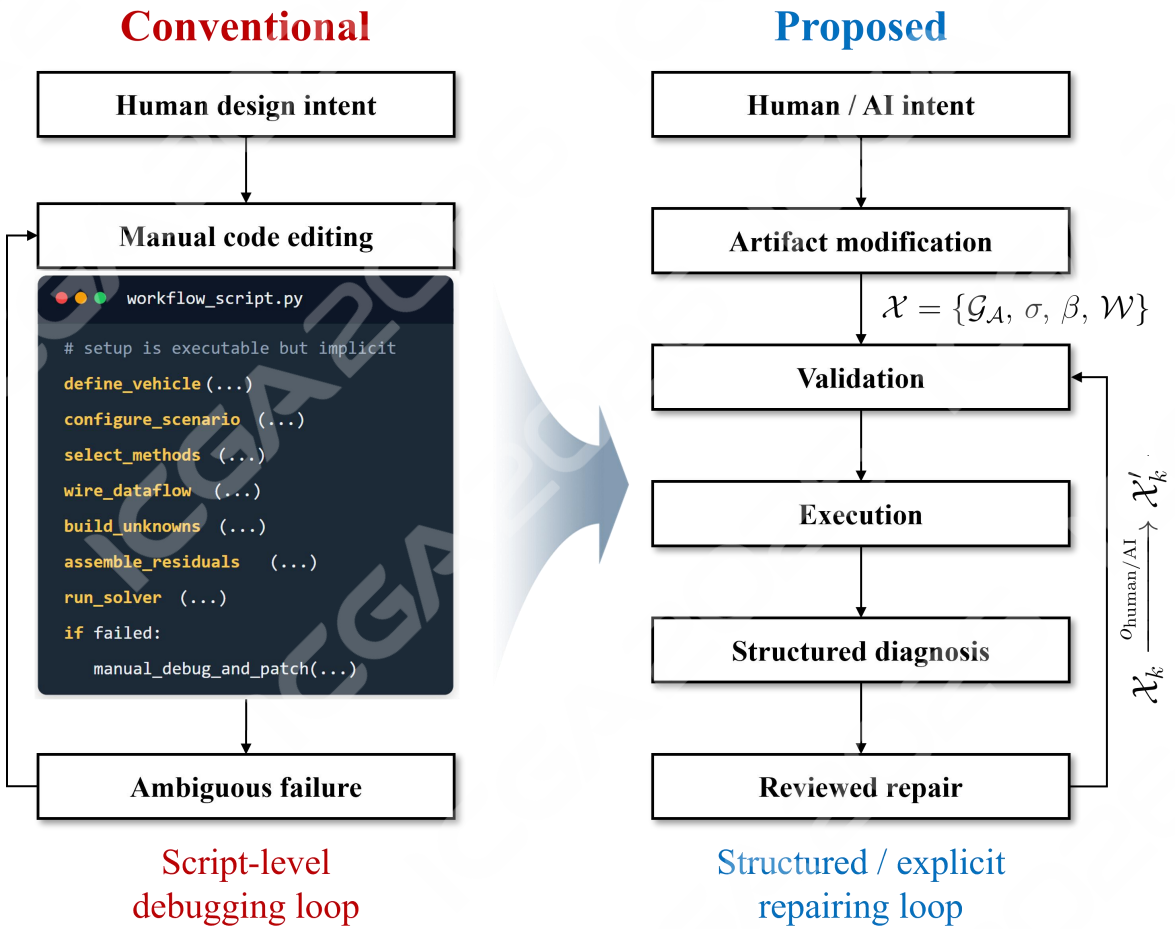


METHODOLOGY FOR AI-OPERABILITY

Bottleneck 2. Execution-Level Solution: Operability at (Explicit) Artifact※-Level

※ An artifact is a concrete, identifiable, and inspectable object produced by a defined process.

- **Goal : to make repeated system modeling & MDAO operations inspectable, modifiable, diagnosable, reviewable, and repairable by both engineers and AI agents (LLM-based).**



artifact state

$$\mathcal{X} = \{\mathcal{G}_\mathcal{A}, \sigma, \beta, \mathcal{W}\}$$

$$\mathcal{X}_k \xrightarrow{o_{\text{human/AI}}} \mathcal{X}'_k \xrightarrow{\mathcal{E}} \mathcal{O}_k \xrightarrow{\text{review/repair}} \mathcal{X}_{k+1}$$

$\mathcal{G}_\mathcal{A}$: architecture/system graph for architecture $\mathcal{A}$
σ : analysis scenario, mode, segment, or operating context
β : method binding and fidelity selection
$\mathcal{W}$: workflow declaration or execution graph
$o_{\text{human/AI}}$: human or AI-agent operation on declarative artifacts
$\mathcal{E}$: validated framework execution
k : design-iteration index
$\mathcal{X}_k$, $\mathcal{X}'_k$: artifact state before and after modification
$\mathcal{O}_k$: results and diagnostics produced at iteration k

METHODOLOGY FOR AI-OPERABILITY

Explicit & AI-Operable Artifacts

- Artifacts need to be *serializable.* (LLM agents can't handle object instance in the code.)
- Higher traceability is required.

⇒ JSON format-based artifacts definition

✓ Serializable & text-based traceability

@Workspace

Operable & Explicit Artifacts

System Graph persistent design state	**Scenario** analysis context
Workflow Graph MDAO composition	**Diagnostics** results and feedback

- Examples of artifacts

✓ Diagnostics : validation results & feedback (suggestion), etc.

```
{
  "id" : "rotor_1" ,
  "type" : "rotor" ,
  "attrs" : {
    "roles" : ["thruster" , "mass_carrier" ],
    "geometry" : { "R" : 0.61 , "tier": "L1" },
    "methods" : { "aero" : "RotorMT" },
    "controllables" : {
      "collective_deg" : { "range" : [-5,25 ] },
      "rpm" : { "range" : [50,5000 ] }
    },
    "ports" : { "mount", "energy_in", "rpm_cmd" }
  }
}
```

LLM-based assistance becomes practical based on explicit, serializable & traceable design artifacts.

METHODOLOGY FOR AI-OPERABILITY

Bounded Agent Operation Contract on Design Artifacts

- AI agents do not directly edit scripts(codes), so that lightweight LLMs can operate reliably.
- They invoke bounded operations with typed inputs, validation checks, and reviewable outputs.

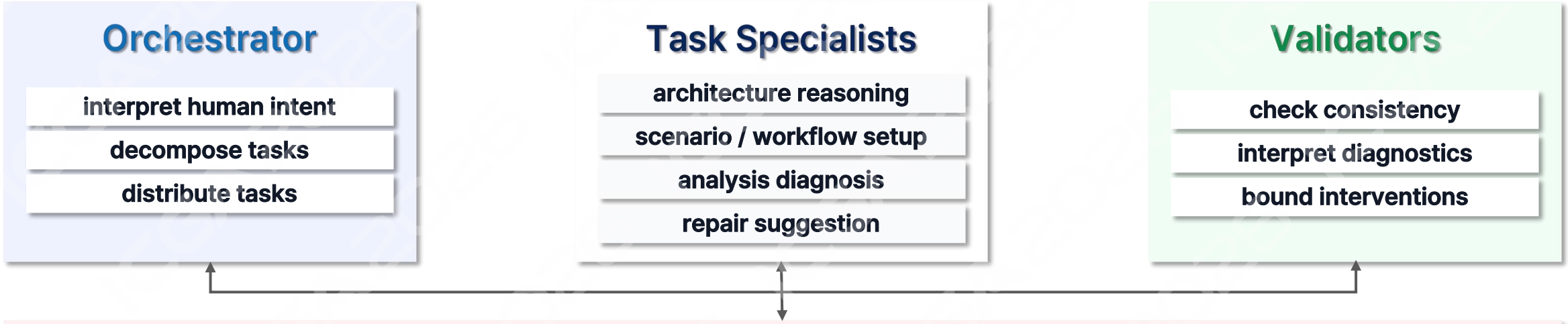
Operation	Target Artifact	Input	Output	Guardrail
Inspect	system & workflow graph / scenario	artifact id	current state	read-only
Query	system & workflow graph / method library	dependency, port, role	candidates	schema-filtered
Propose	system & workflow graph / scenario	modification request	candidate diff	no state change yet
Validate	modified artifacts	candidate state	pass/fail diagnostics	mandatory
Run Analysis	system & workflow graph / scenario	executable setup	result artifact	validated only
Repair	all artifacts with diagnostics	failure object	repair proposal	after human review

Structured operations define the agent's inspection, query, proposal, and evaluation space on design artifacts.

METHODOLOGY FOR AI-OPERABILITY

Multi-Agents System & Role Decomposition

- Coordinator, specialists, and validation interact over the same explicit design artifacts
- AI assistance is organized by role, not by a single monolithic agent.
- Reasoning, specialization, and checking are separated around shared design artifacts.
- Agent reasoning is grounded in workspace context: current artifact state, recent modifications, and active user focus.



Multi-agents system & role separation makes the AI-assistance loop more efficient & economic.



5. COMPUTATION REALIZATION: AMADEVS

COMPUTATION REALIZATION: AMADEVS

AMADEVS: Advanced Modeling, Analysis & Design Environment for Vehicle Systems[1]

⇒ An **in-house Python library** for arbitrary aircraft modeling & MDAO workflow building (BACKEND)

- ✓ Architecture-Agnostic Design Foundation
- ✓ Explicit and AI-Operable Interface

AMADEVS

python

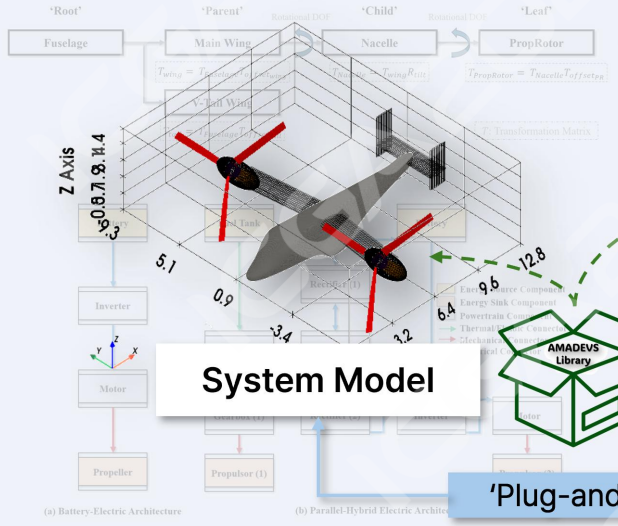
AMADEVS
as a foundation of modeling & execution for aircraft MDAO

System Modeling → **Workflow Modeling** → **MDAO Execution**

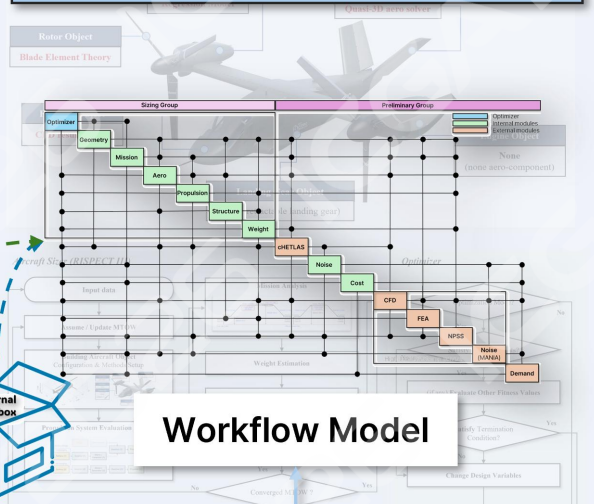
① Build System Model

② Build Workflow Model

③ Execute & Get Results



System Model

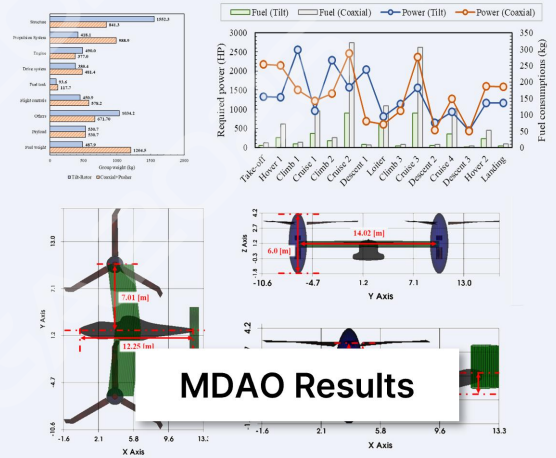


Workflow Model

AMADEVS Library

External Toolbox

'Plug-and-Play' Modeling



MDAO Results

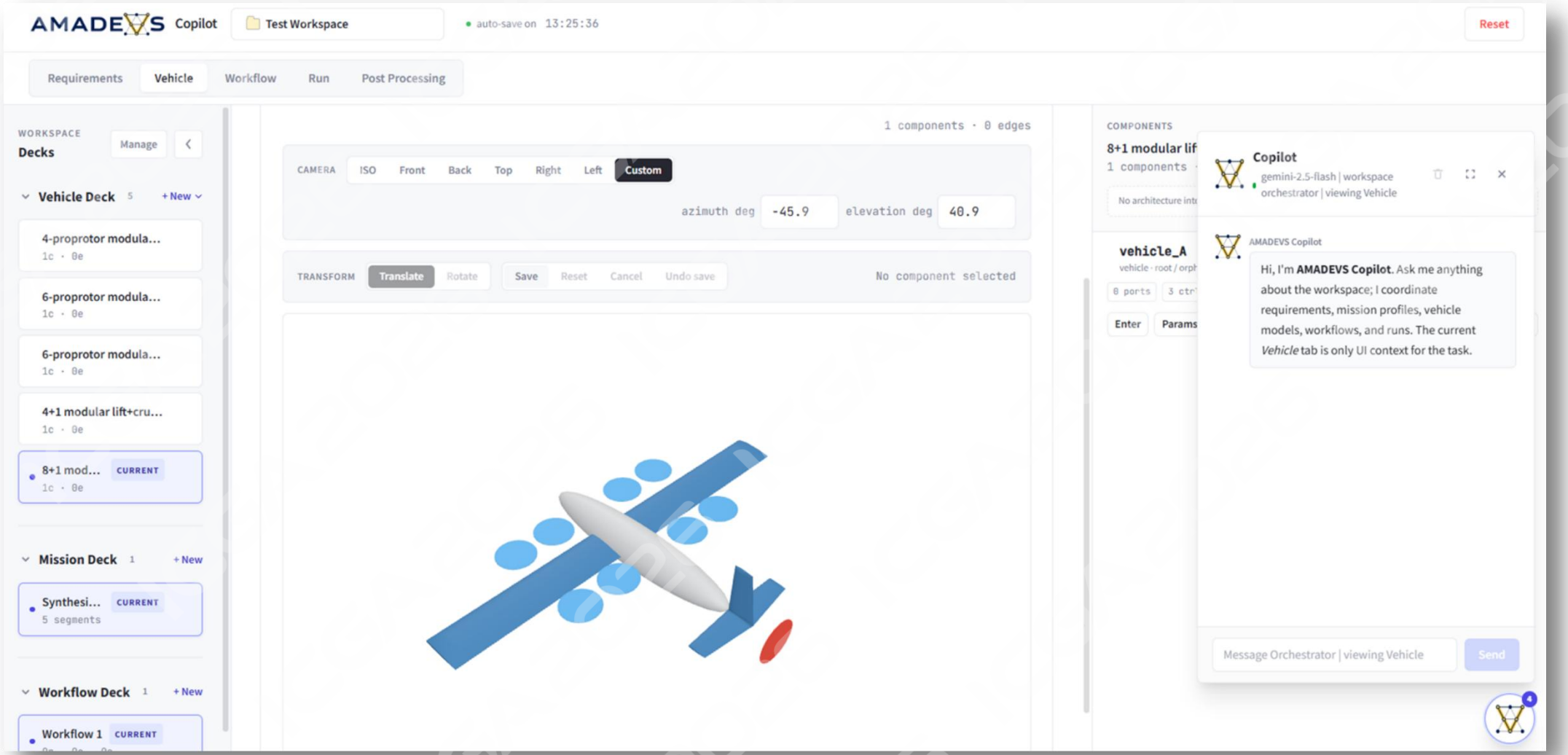
Rapid Prototyping of Aircraft MDAO

[1] Kim, J., Park, S., Chatelain, B., Jeong, S., Ryu, J., Joo, S., & Yee, K. (2026). AMADEVS: A Generalized Aircraft Modeling & Analysis Environment for Multidisciplinary Design. In AIAA SCITECH 2026 Forum (p. 1687).

COMPUTATION REALIZATION: AMADEVS

AMADEVS Web-Based Platform with AI-Copilot (FRONTEND)

- Requirements & Mission ⇒ Vehicle Modeling ⇒ Workflow Modeling ⇒ Run ⇒ Post Processing





6. APPLICATIONS & DEMONSTRATION

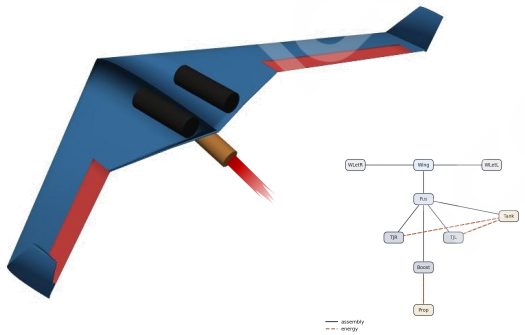
APPLICATIONS & DEMONSTRATION

Demonstration Strategy

- All applications were presented in the dissertation:
  - ✓ **"Kim, J. (2026). Architecture-agnostic and AI-operable aircraft design using a graph-based methodology [Doctoral dissertation, Seoul National University]."**

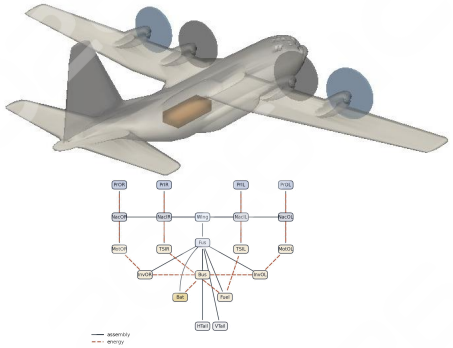
Application I

- **RATO BWB UAS Trim Analysis**
  - ✓ Introduces a NEW force contributor
  - ⇒ Solid-rocket booster
    - That is absent from conventional aircraft taxonomy
  - ⇒ Evidence of arch-agnostic EOM assembly



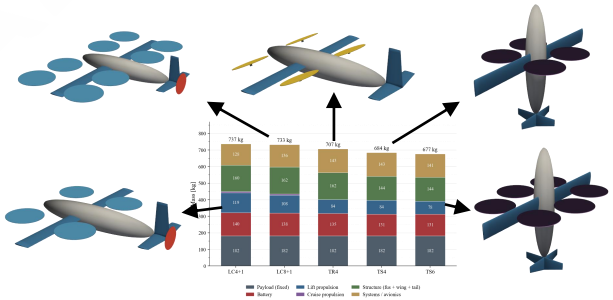
Application II

- **True-Parallel Hybrid Cargo Aircraft Mission Analysis**
  - ✓ Introduces a NEW propulsion architecture
  - ⇒ True-parallel hybrid system
    - That is absent from conventional propulsion system taxonomy
  - ⇒ Evidence of arch-agnostic EB assembly



Application III

- **eVTOL UAS Design Exploration**
  - ✓ Demonstrates the end-to-end aircraft design.
  - ⇒ Using AI-Copilot System
    - Resolving the execution-level bottleneck of MDAO
  - ⇒ Effectiveness of AI-operable aircraft design



Application III

AI-Assisted Aircraft Design – Winged-eVTOL Aircraft Design

- Mission & Requirements → Architecture Candidates Generation →Workflow Modeling → Run → Results

*I want to design a **full-battery winged eVTOL aircraft that delivers a 182 kg payload** over the Uber Elevate UAM reference mission — 100 km cruise at 55 m/s, cruise altitude 500 m, with vertical takeoff and landing at sea level.*

*Please set up the **formal requirements** (hard payload, hard MTOW cap of 1500 kg, soft range target, full-battery assumption) and **a five-segment representative mission profile** (takeoff hover, climb to cruise, 100 km cruise, descent, landing hover).*

Brainstorm a set of candidate winged-VTOL architectures suitable for this mission
— span lift-plus-cruise, tilt-rotor, tailsitter, and partial-tilt classes, with a couple of rotor-count variants where it makes sense.

*Set up a **generic initial-sizing workflow** at the workspace level — architecture-agnostic*
*in shape: the same chain of **propulsion system sizing → trim → mission-energy-flow → mission-KPI → weight-estimation, wrapped by an MTOW fix**-point loop*

APPLICATIONS & DEMONSTRATION

Application III — AI-Assisted Aircraft Design – Winged-eVTOL Aircraft Design

- Mission & Requirements → Architecture Candidates Generation →Workflow Modeling → Run → Results

*I want to design a **full-battery winged eVTOL aircraft that delivers a 182 kg payload** over the Uber Elevate UAM reference mission — 100 km cruise at 55 m/s, cruise altitude 500 m, with vertical takeoff and landing at sea level.*

*Please set up the **formal requirements** (hard payload, hard MTOW cap of 1500 kg, soft range target, full-battery assumption) and **a five-segment representative mission profile** (takeoff hover, climb to cruise, 100 km cruise, descent, landing hover).*

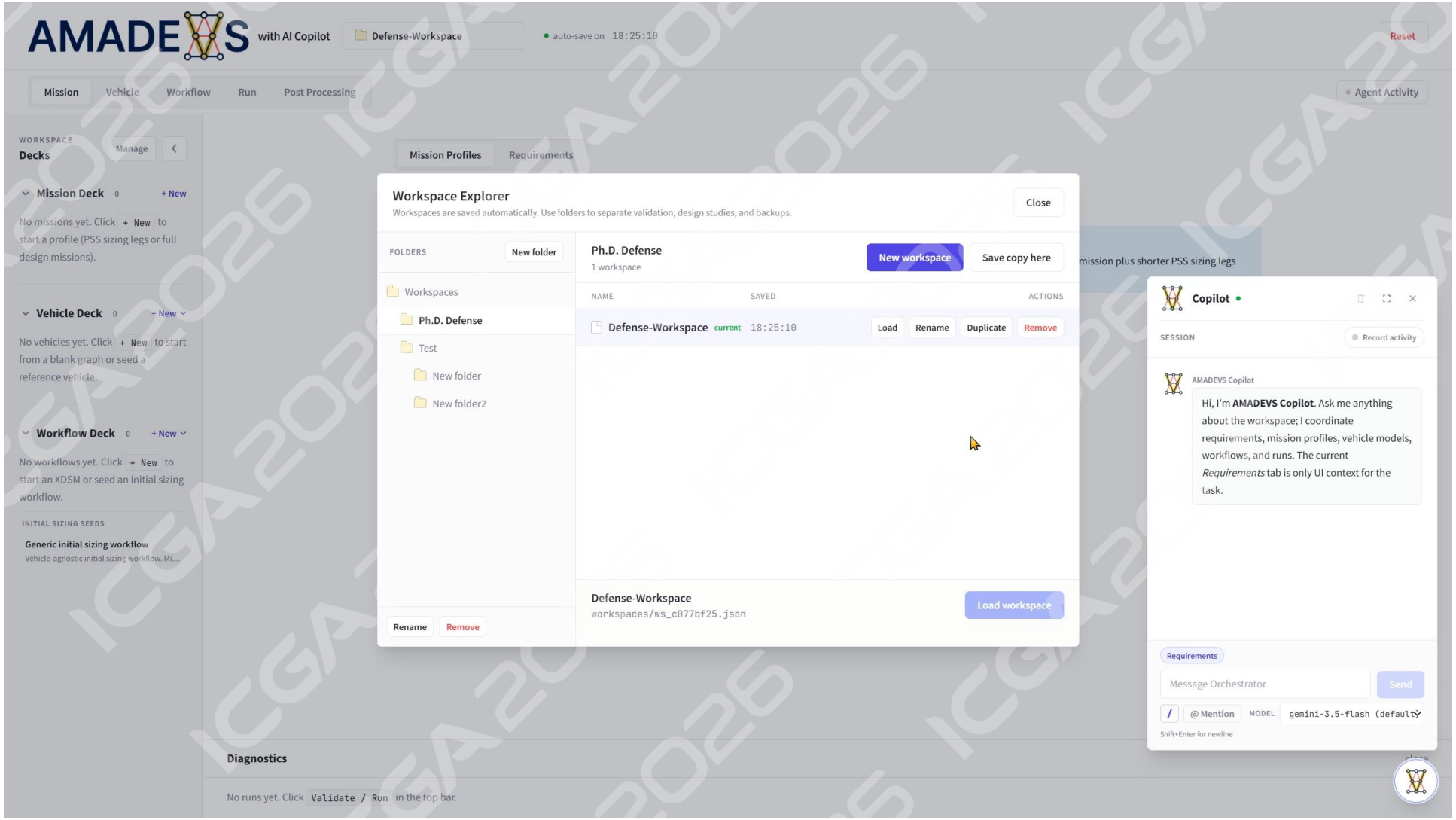
Brainstorm a set of candidate winged-VTOL architectures suitable for this mission
— span lift-plus-cruise, tilt-rotor, tailsitter, and partial-tilt classes, with a couple of rotor-count variants where it makes sense.

*Set up a **generic initial-sizing workflow** at the workspace level — architecture-agnostic*
*in shape: the same chain of **propulsion system sizing** → **trim** → **mission-energy-flow** → **mission-KPI** → **weight-estimation**, wrapped by an MTOW fix*

APPLICATIONS & DEMONSTRATION

Application III AI-Assisted Aircraft Design – Winged-eVTOL Aircraft Design [video]

- Mission & Requirements → Architecture Candidates Generation →Workflow Modeling → Run → Results



APPLICATIONS & DEMONSTRATION

Application III AI-Assisted Aircraft Design – Winged-eVTOL Aircraft Design

- Mission & Requirements → Architecture Candidates Generation →Workflow Modeling → Run → Results

I want to design a full-battery winged eVTOL aircraft that delivers a 182 kg payload over the Uber Elevate UAM reference mission — 100 km cruise at 55 m/s, cruise altitude 500 m, with vertical takeoff and landing at sea level.

Please set up the formal requirements (hard payload, hard MTOW cap of 1500 kg, soft range target, full-battery assumption) and a five-segment representative mission profile (takeoff hover, climb to cruise, 100 km cruise, descent, landing hover).

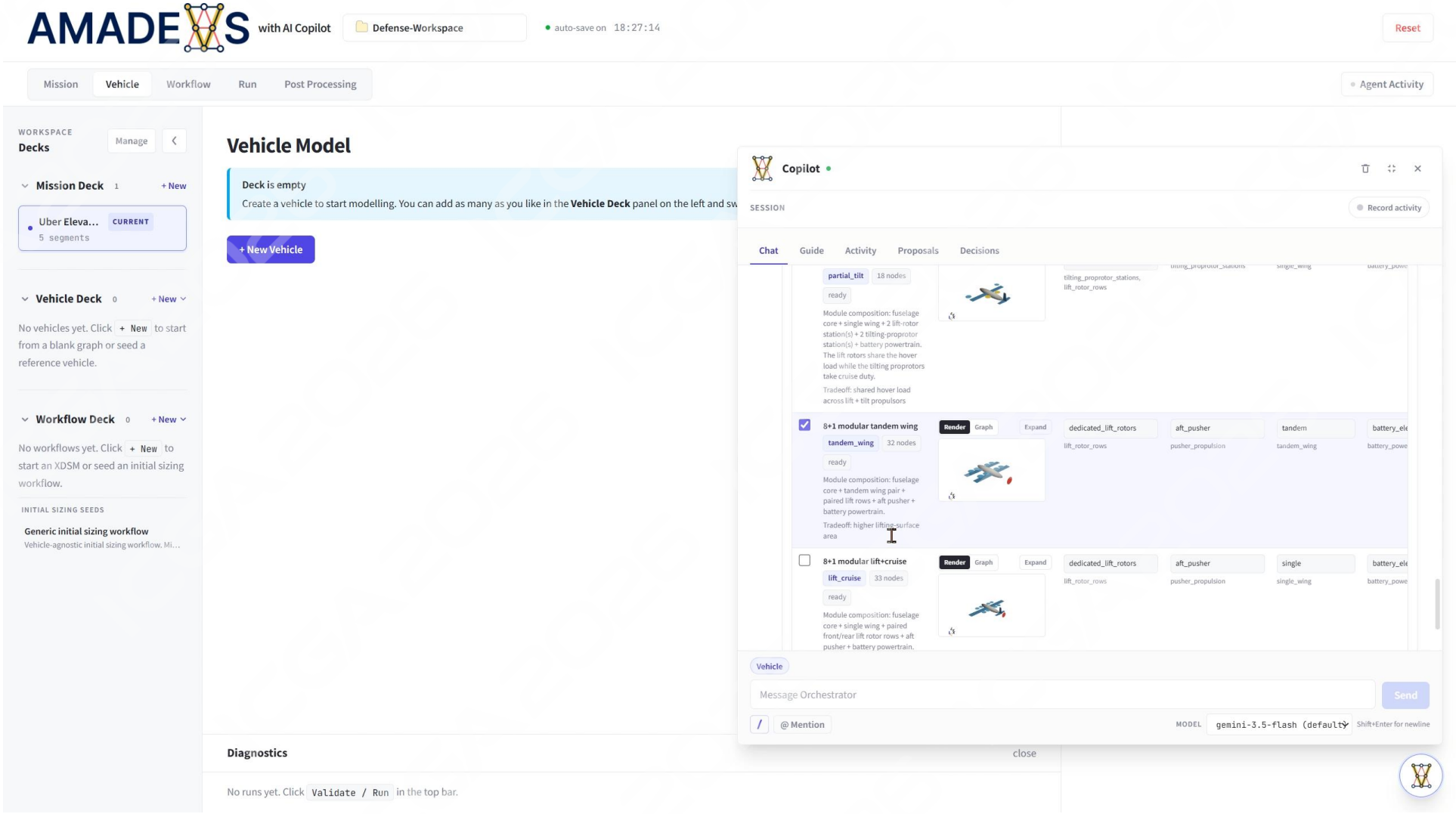
Brainstorm a set of candidate winged-VTOL architectures suitable for this mission
— span lift-plus-cruise, tilt-rotor, tailsitter, and partial-tilt classes, with a couple of rotor-count variants where it makes sense.

Set up a generic initial-sizing workflow at the workspace level — architecture-agnostic
in shape: the same chain of propulsion system sizing → trim → mission-energy-flow → mission-KPI → weight-estimation, wrapped by an MTOW fixed-point loop

APPLICATIONS & DEMONSTRATION

Application III AI-Assisted Aircraft Design – Winged-eVTOL Aircraft Design [video]

- Mission & Requirements → Architecture Candidates Generation →Workflow Modeling → Run → Results



APPLICATIONS & DEMONSTRATION

Application III AI-Assisted Aircraft Design – Winged-eVTOL Aircraft Design [video]

- Mission & Requirements → Architecture Candidates Generation → Workflow Modeling → Run → Results

I want to design a full-battery winged eVTOL aircraft that delivers a 182 kg payload over the Uber Elevate UAM reference mission — 100 km cruise at 55 m/s, cruise altitude 500 m, with vertical takeoff and landing at sea level.

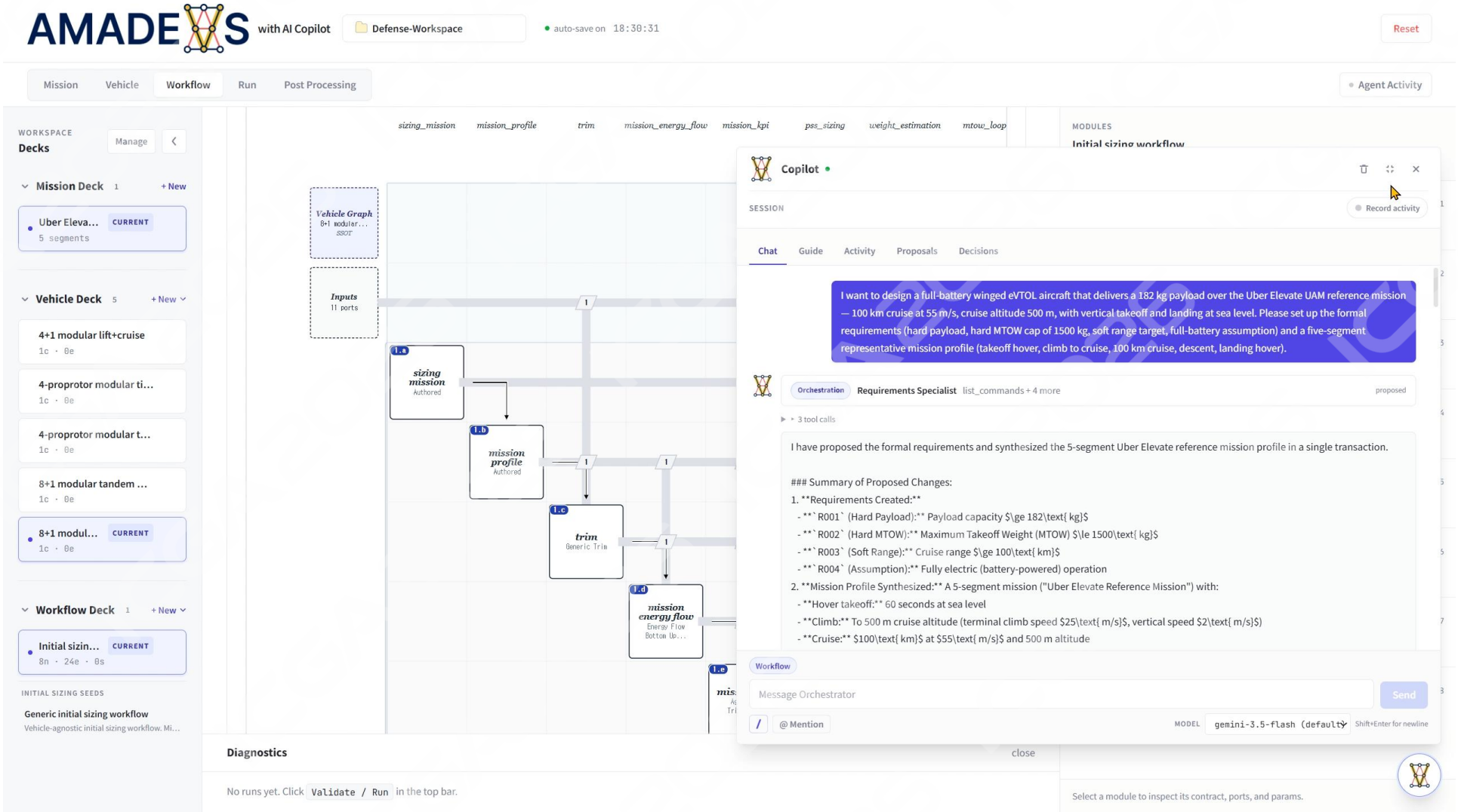
Please set up the formal requirements (hard payload, hard MTOW cap of 1500 kg, soft range target, full-battery assumption) and a five-segment representative mission profile (takeoff hover, climb to cruise, 100 km cruise, descent, landing hover).

Brainstorm a set of candidate winged-VTOL architectures suitable for this mission
— span lift-plus-cruise, tilt-rotor, tailsitter, and partial-tilt classes, with a couple of rotor-count variants where it makes sense.

Set up a generic initial-sizing workflow at the workspace level — architecture-agnostic
in shape: the same chain of propulsion system sizing → trim → mission-energy-flow → mission-KPI → weight-estimation, wrapped by an MTOW fixed-point loop

Application III AI-Assisted Aircraft Design – Winged-eVTOL Aircraft Design [video]

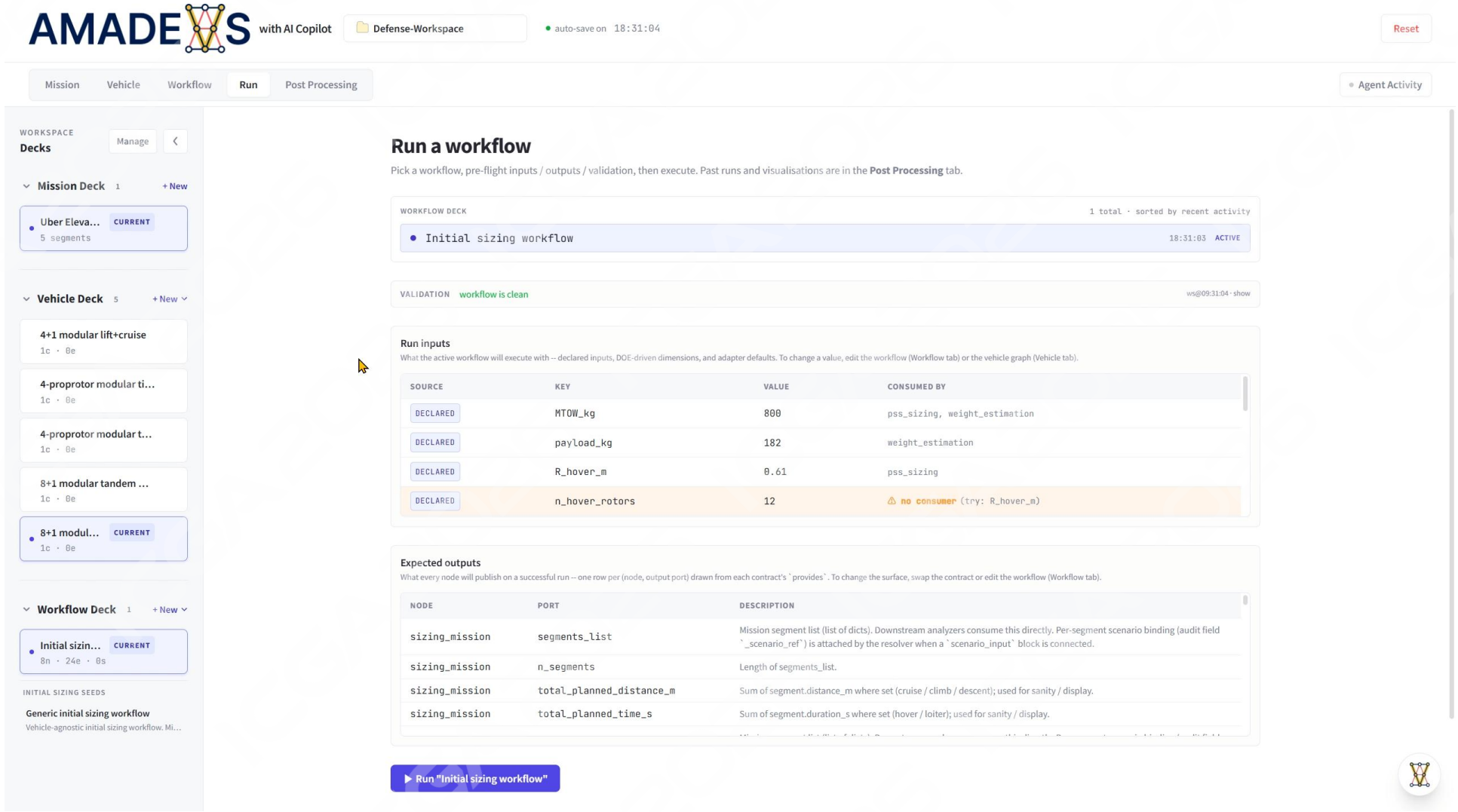
- Mission & Requirements → Architecture Candidates Generation → Workflow Modeling → Run → Results



APPLICATIONS & DEMONSTRATION

Application III AI-Assisted Aircraft Design – Winged-eVTOL Aircraft Design [video]

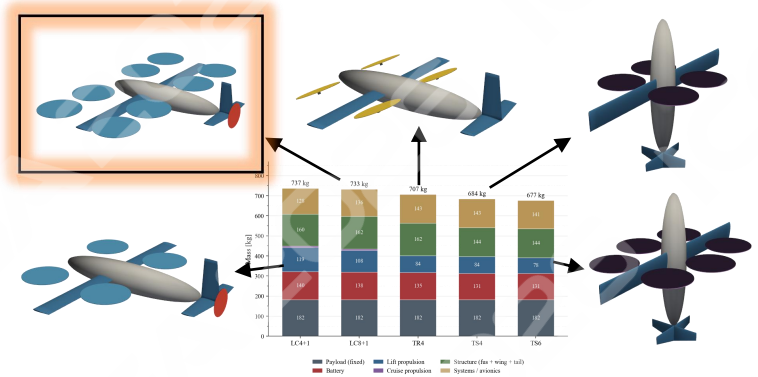
- Mission → Architecture Candidates Generation →Workflow Modeling → Run → Results



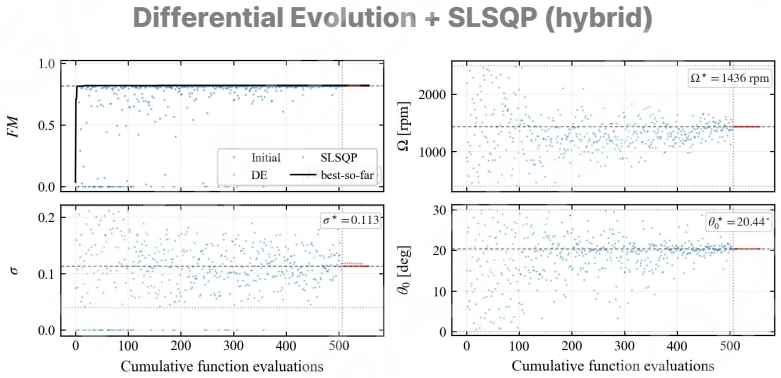
APPLICATIONS & DEMONSTRATION

Application III AI-Assisted Aircraft Design – Winged-eVTOL Aircraft Design

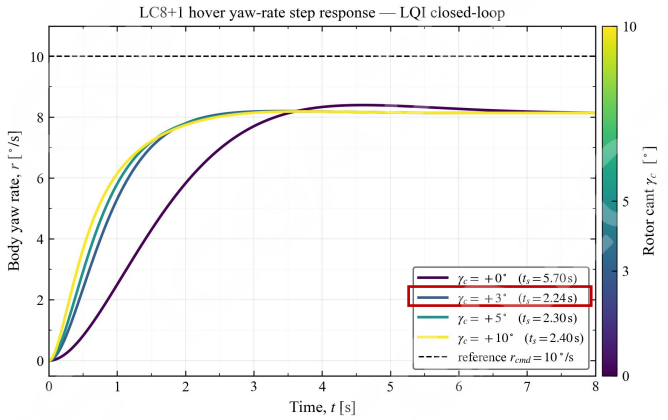
- Preliminary Design Examples (summarized)



SLSQP MTOW minimization results



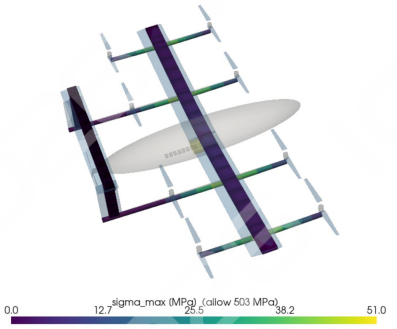
Optimization of aerodynamic components (rotor/propeller/wing/tail)



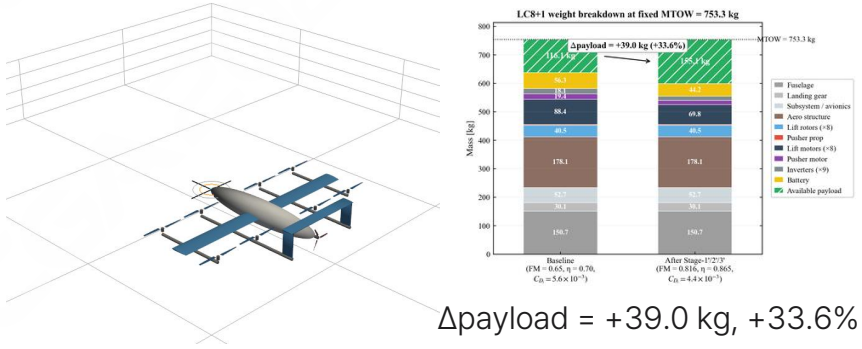
Yaw authority improvement (rotor cant angle design)



Baseline model enrichment using AI Copilot [video]



Wing/boom structural sizing (1-D beam network model)



Δpayload = +39.0 kg, +33.6%

Pirouette flight simulation [video] & final results



CONCLUDING REMARKS

CONCLUDING REMARKS

Conclusion & Contributions

Problem Formulation
at Methodology-/Execution-Level of Aircraft MDAO

Developed an Architecture-Agnostic MDAO Foundation

- Graph-Based System & Workflow Representation
- Reusable / Topology-Invariant Analysis Formulation

Developed a methodology for AI-Operability

- AI-Operable Artifact Exposure
- Multi-Agent Collaboration Structure

Computational Realization (AMADEVS)

- Arbitrary Modeling of System & Workflow Architecture

Human-AI Collaboration Interface

- Collaboration between Human Engineer & AI-Agents

Demonstration of Three Applications (RATO/TPH/eVTOL aircraft)

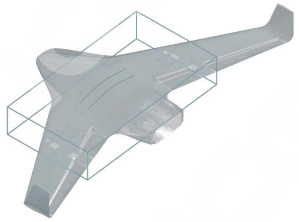
Validated Rapid Aircraft Design via Architecture-Agnostic & AI-Operable Methodology

Enables Scalable Aircraft MDAO for Arbitrary Architectures

CONCLUDING REMARKS – FUTURE WORK

Industrial Extension (by a co-founded startup, *NvilZ Co., Ltd.* www.nvilz.ai)

- The graph-based, AI-operable design methodology was re-implemented as a commercial engineering stack (AIDEN) and applied to an agentic UAV design.
- A single design graph carries the aircraft from concept to manufacturing review.
- Beyond the research scope:
  - ✓ Knowledge-base for LLM agent (specialized to aerospace engineering)
  - ✓ Added an off-the-shelf component database
  - ✓ CAD-level detailed design, and assembly simulation. (Driven by LLM agents)



△ Industrial extension example: CAD-level detailed design (by NvilZ Co., Ltd.)

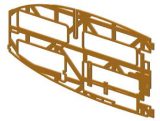
Future Work

- **Architecture-Level Optimization**
  - ✓ Topology optimization over vehicle graphs

    ⇒ Toward architecture-variable optimization
- **Advanced Agent Engineering**
  - ✓ Optimizing multi-agent collaboration structure for better performance
  - ✓ Automated agent-driven aircraft design

    ⇒ Toward autonomously evolving aircraft design
- **Integrated MBSE-MDAO**

  ⇒ Toward autonomous end-to-end aircraft design



△ Industrial extension example: assembly simulation (by NvilZ Co., Ltd.)

Thank you for your attention

Contact info.

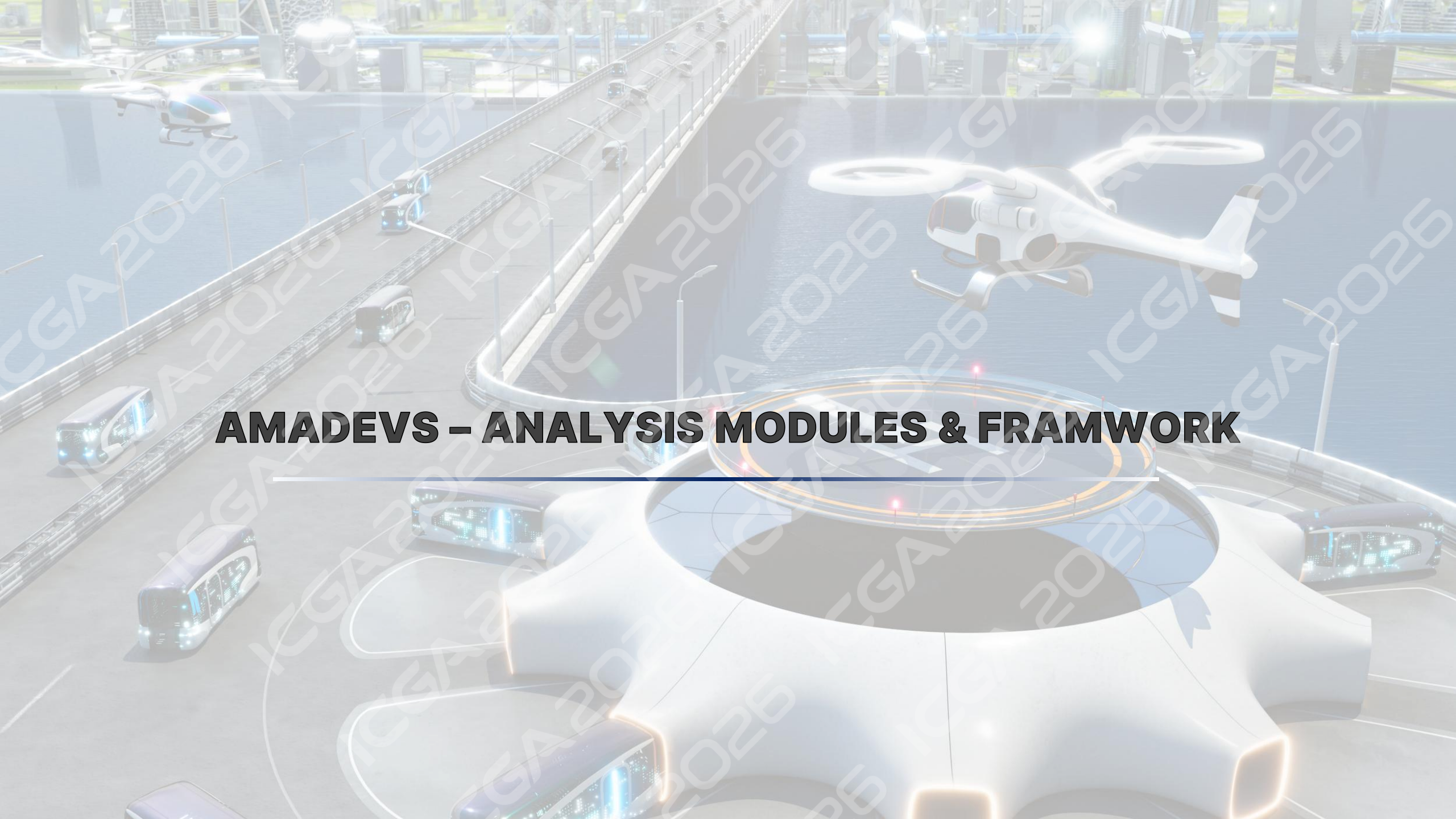
Kwanjung Yee (kjyee@snu.ac.kr)

Jeonghwan Kim (jace.kim@nvilz.ai)

Seoul National University

Department of Aerospace Engineering

Aerospace Vehicle Design Laboratory



AMADEVS – ANALYSIS MODULES & FRAMWORK

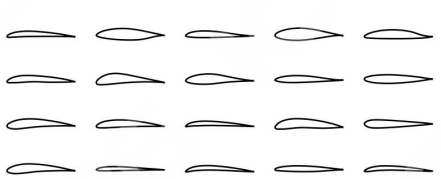
[1] Kim, Hojin, et al. "Improved Conceptual Design of eVTOL Aircraft: Considering Rotor–Rotor Interactional Effects." International Journal of Aeronautical and Space Sciences (2025): 1-20.
[2] Clarke, Matthew A., et al. "Aerodynamic optimization of wing-mounted propeller configurations for distributed electric propulsion architectures." AIAA Aviation 2021 Forum. 2021.
[3] Lukaczyk, Trent W., et al. "SUAVE: an open-source environment for multi-fidelity conceptual vehicle design." 16th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference. 2015.
[4] Jeong, S., Kim, J., & Yee, K. (2026). Development of an Aero-Structural Aircraft Fuselage Optimization Methodology Considering Internal Layout. In AIAA SCITECH 2026 Forum (p. 1899).
[5] Kang, Y.-E., Lee, D., & Yee, K. (2025). Intuitive and feasible geometric representation of airfoil using variational autoencoder. Journal of Computational Design and Engineering, 12(2), 27–48.
[6] Kang, Y.-E. (2024). Efficient Machine Learning Framework for Extensive Design Space Exploration in Aerodynamic Optimization of Lifting Surfaces. Ph.D. Dissertation, Seoul National University.

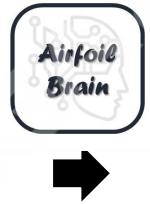
Available Analysis Modules in AMADEVS

- **Aerodynamics**
  - ✓ [wing/rotor/prop] Steady/unsteady VLM
  - ✓ [wing] Non-linear lifting line theory
  - ✓ [wing] Strip-theory-based profile drag estimation
  - ✓ [rotor/prop] BET with various inflow-models
  - ✓ [rotor/prop] Extended momentum theory
  - ✓ [prop-wing] Propeller-wing coupled solver (Conway's)
  - ✓ [components] DATCOM-/regression-based aerodynamic models
  - ✓ [components] Semi-empirical & regression models
  - ✓ [rotor/prop] CAMRAD II & regressions
  - ✓ [airfoil] Airfoil Brain [ref]
  - ✓ [fuselage] COMPACT – aerodynamic module [4]
  - ✓ [components] (external) CFD solvers
- **Structure**
  - ✓ [1D structural components] Euler-Bernoulli / Timoshenko beam model
  - ✓ [structure network] 1D beam network analysis module
  - ✓ [fuselage] COMPACT – structure module [4]
  - ✓ [components] (external) FEAT solvers

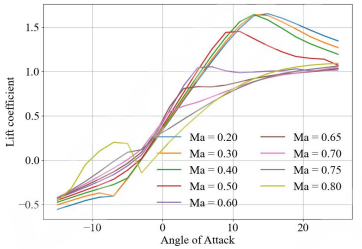
⌗ Modules developed but not embedded at the moment

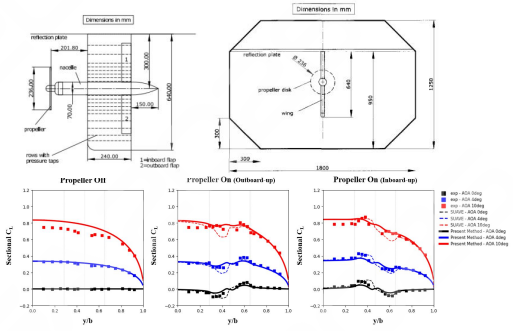
Design candidates

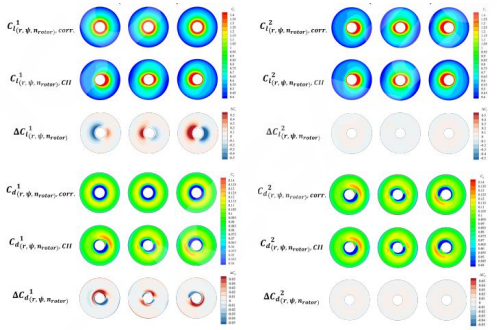


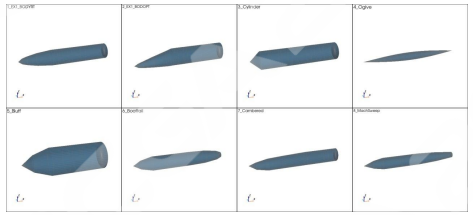


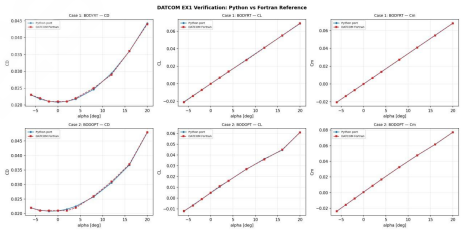
Performance metrics (C81 table)

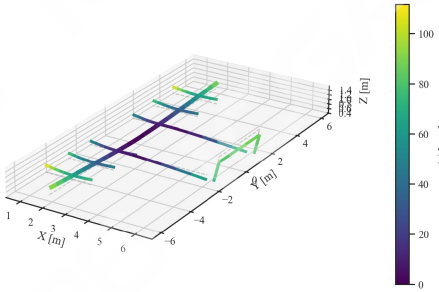


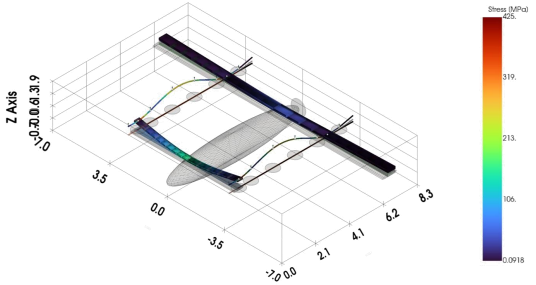










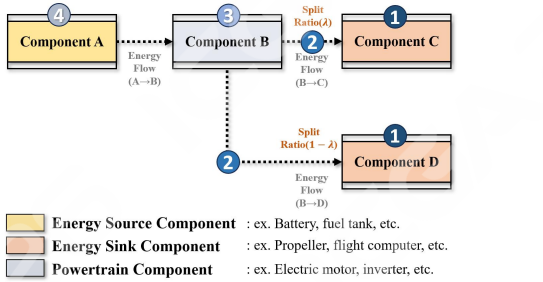


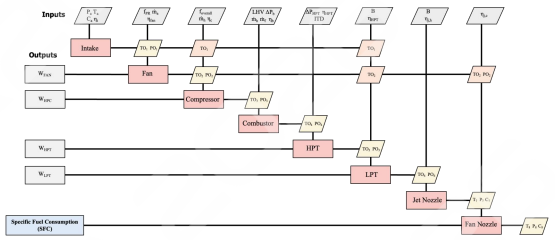
Available Analysis Modules in AMADEVS

Propulsion System

- ✓ Generalized energy-flow solver
- ✓ [motor] AF-PMSM design module
- ✓ [motor] Equivalent circuit and MTPA control based analysis & regression models
- ✓ [generator] Equivalent circuit and MPPA control based analysis & regression models
- ✓ [inverter/converter/rectifier] Average loss model & regression models
- ✓ [battery] Simple near-linear discharge model (SAMSUNG INR18650-30Q)
- ✓ [battery] Battery voltage drop model using Thevenin equivalent circuit model
- ✓ [GT] Simple cycle models & (external) PyCycle [6]
- ✓ [solid rocket] Solid rocket boost analysis & design module
- ✓ [turbojet] turbojet engine optimal sizing module
- ✓ [wiring] Single Void Discharge method based wiring model
- ✓ [battery TMS] Air-/liquid-cooled battery TMS sizing module
- ✓ [HFC] Integrated HFC system analysis module

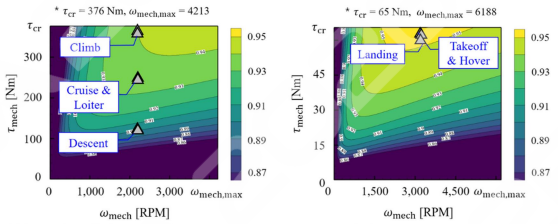
※ Modules developed but not embedded at the moment



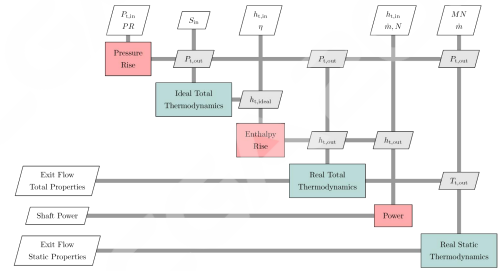


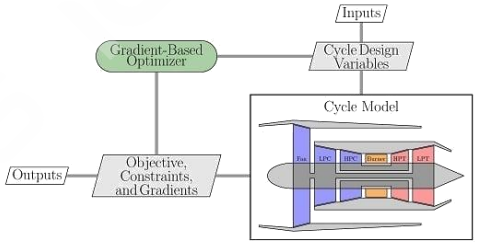
1. Step 1: Start at Energy Sink Components (Requires Power P)
2. Step 2: Propagate Demand Upwards via Connectors
3. Step 3: Calculate Required Input for Predecessors ($P_{in} = P_{out}/\eta$)
4. Step 4: Aggregate Total Energy/Fuel Demand at Energy Sources

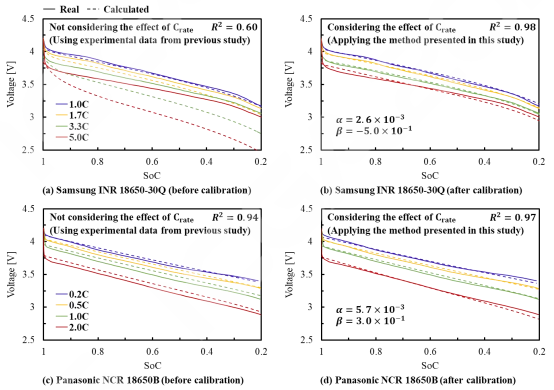
Thermal/Fluidic Connector : ex. Fuel flow, etc.
Mechanical Connector : ex. Shaft torque, RPM, etc.
Electrical Connector : ex. Voltage, current, etc.



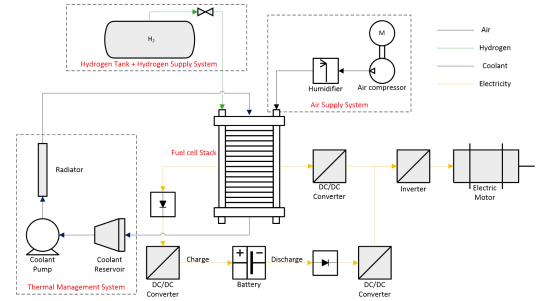
a) Motor 1 (propeller) b) Motor 2 (12 lift-DPs)







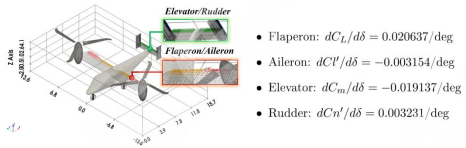
(a) Samsung INR 18650-30Q (before calibration) (b) Samsung INR 18650-30Q (after calibration)
(c) Panasonic NCR 18650B (before calibration) (d) Panasonic NCR 18650B (after calibration)



[1] Lee, D., & Yee, K. (2024). Novel Electric Propulsion System Analysis Method for Electric Vertical Takeoff and Landing Aircraft Conceptual Design. Journal of Aircraft.
[2] Park, Junhwi, Taemin Jeong, and Kwanjung Yee. "Development of Air-Cooled Battery Sizing Method for eVTOL Aircraft." *AIAA AVIATION FORUM AND ASCEND 2024*. 2024.
[3] Lim W, Samsung SDI, INR18650-30Q battery datasheet.
[4] Park, J., Lee, D., & Yee, K. (2022). Development of Fuel Cell/Battery Hybrid System Sizing Methodology for eVTOL Aircraft. In AIAA AVIATION 2022 Forum.
[5] Park, J., Lee, D., Lim, D., & Yee, K. (2022). A refined sizing method of fuel cell-battery hybrid system for eVTOL aircraft. Applied Energy.
[6] Hendricks, E. S., & Gray, J. S. (2019). pyCycle: A tool for efficient optimization of gas turbine engine cycles. Aerospace, 6(8), 87.

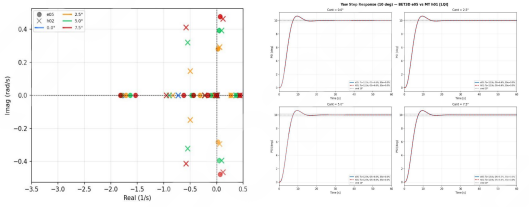
Available Analysis Modules in AMADEVS

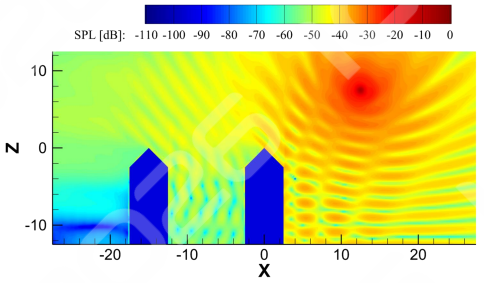
- **Flight Dynamics & Control**
  - ✓ Generalized equation of motion builder & solver
  - ✓ Vehicle moment of inertial estimation module
  - ✓ Static-/dynamic stability analysis modules
  - ✓ Linear mixer design module
  - ✓ LQR/LQI-based optimal controller design modules
  - ✓ Rotorcraft/fixed-wing flying qualities analysis modules
- **Noise**
  - ✓ [source] Gutin and Deming formulation
  - ✓ [source] Loading/thickness: Ffowcs Williams-Hawkings(FW-H) Farrasat 1A formula
  - ✓ [source] Broadband: Brooks-Pope-Marcolini (BPM) airfoil self-noise
  - ✓ [propagation] Gaussian Beam Tracing (GBT)
  - ✓ [propagation] NPD data (noise-power-distance) based
- **Others**
  - ✓ ATR estimation model (climate impact analysis)
  - ✓ Aircraft cost analysis module (MADAM) [4]
  - ✓ Component layout optimization module (in COMPACT [5])
    - ⁜ Modules developed but not embedded at the moment

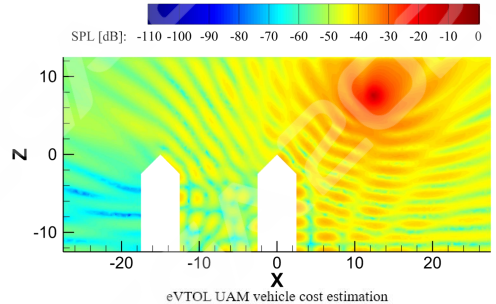


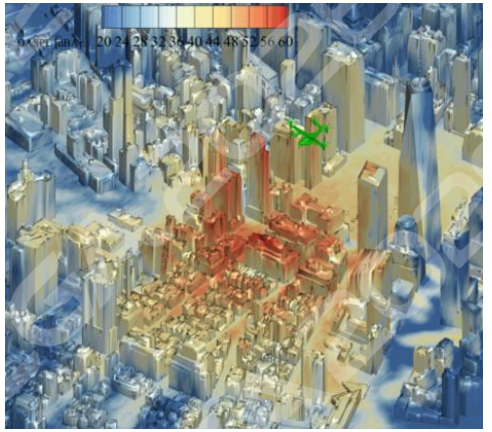
- Flaperon: $dC_L/d\delta = 0.020637/\text{deg}$
- Aileron: $dCl'/d\delta = -0.003154/\text{deg}$
- Elevator: $dC_m/d\delta = -0.019137/\text{deg}$
- Rudder: $dCn'/d\delta = 0.003231/\text{deg}$

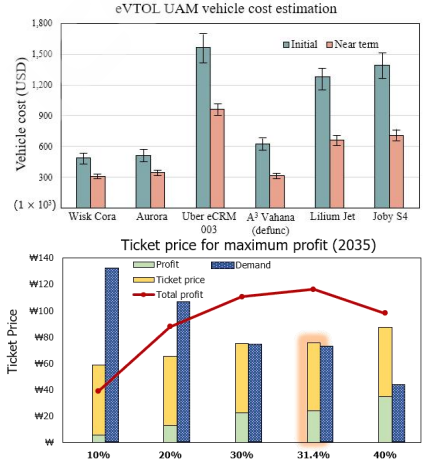
Control	$dC_L/d\delta$	$dC_Y/d\delta$	$dCl'/d\delta$	$dC_m/d\delta$	$dCn'/d\delta$
Flaperon	0.020637	0.000000	-0.000000	0.000091	0.000000
Aileron	0.000000	0.000219	-0.003154	-0.000000	0.000153
Elevator	0.005837	0.000000	-0.000000	-0.019137	-0.000000
Rudder	-0.000000	-0.004720	-0.000202	0.000000	0.003231

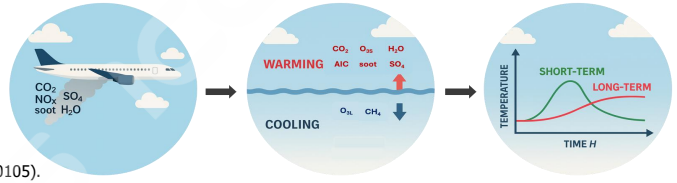


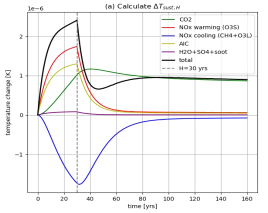










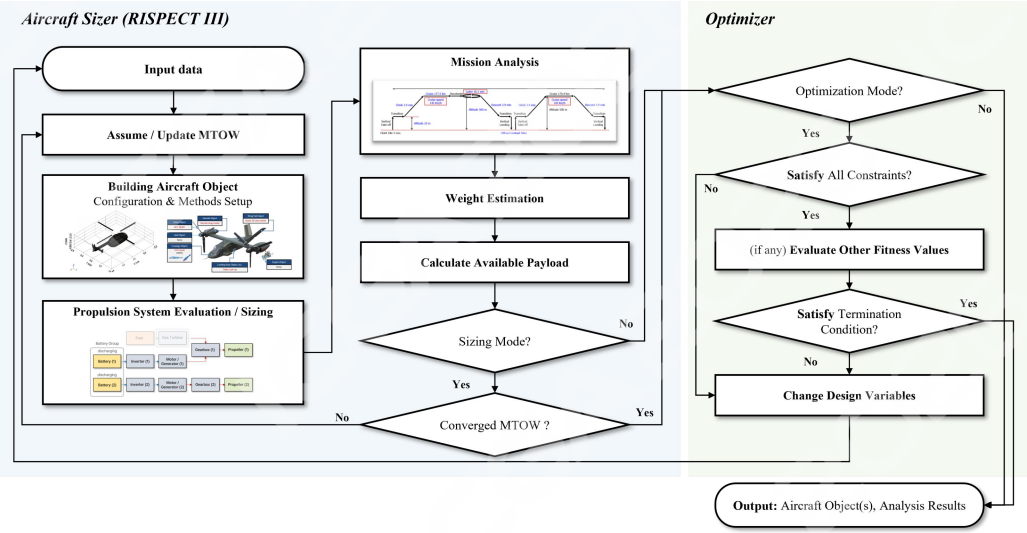


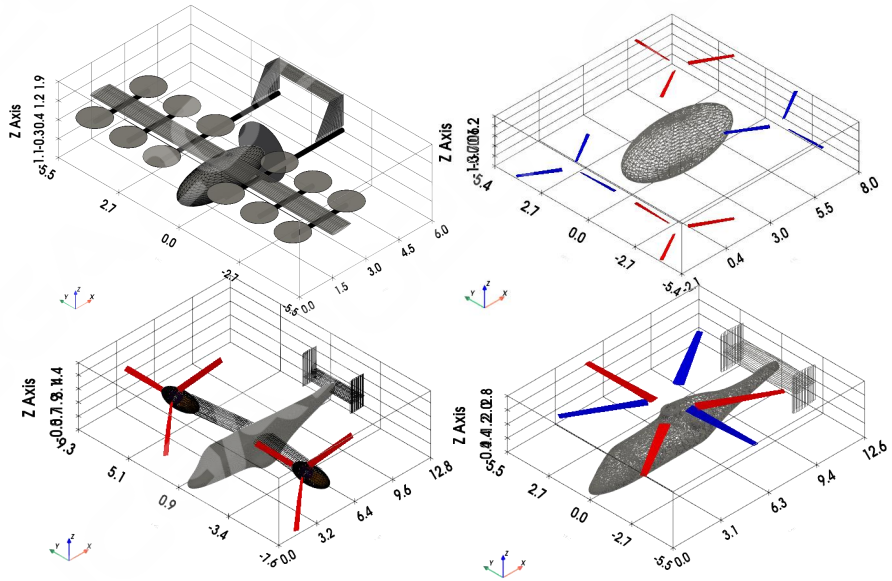
[1] Furkat Yunus et al. 2023. "Efficient prediction of airborne noise propagation in a non-turbulent urban environment using Gaussian beam tracing method." J. Acoust. Soc. Am. 153, 2362
[2] V. Cerveny et al. 1982. "Computation of wave fields in inhomogeneous media—Gaussian beam approach." Geophysical Journal International 70, 109.
[3] Brown, A., & Harris, W. (2018). A vehicle design and optimization model for on-demand aviation. In 2018 AIAA/ASCE/AHS/ASC structures, structural dynamics, and materials conference (p. 0105).
[4] Kim, H., Lee, H., & Yee, K. (2025). A Framework for Deriving Top-Level Requirements in UAM Ecosystem with System Performance Considerations. In AIAA SCITECH 2025 Forum (p. 2697).
[5] Jeong, S., Kim, J., & Yee, K. (2026). Development of an Aero-Structural Aircraft Fuselage Optimization Methodology Considering Internal Layout. In AIAA SCITECH 2026 Forum (p. 1899).

AMADEVS as a Framework Builder

- AMADEVS can also serve as a foundation **for building higher-level design and analysis frameworks** rather than operating only as a stand-alone tool

1. RISPECT III: Rotorcraft System Design & Performance Analysis Framework
   - ✓ RISPECT+[1] has been reimplemented in an object-oriented manner on top of the AMADEVS architecture.
2. ASPECT: Fixed-Wing Aircraft System Design & Performance Analysis Framework[2]
   - ✓ A framework developed under AMADEVS for design, performance analysis, and environmental impact assessment of fixed-wing aircraft



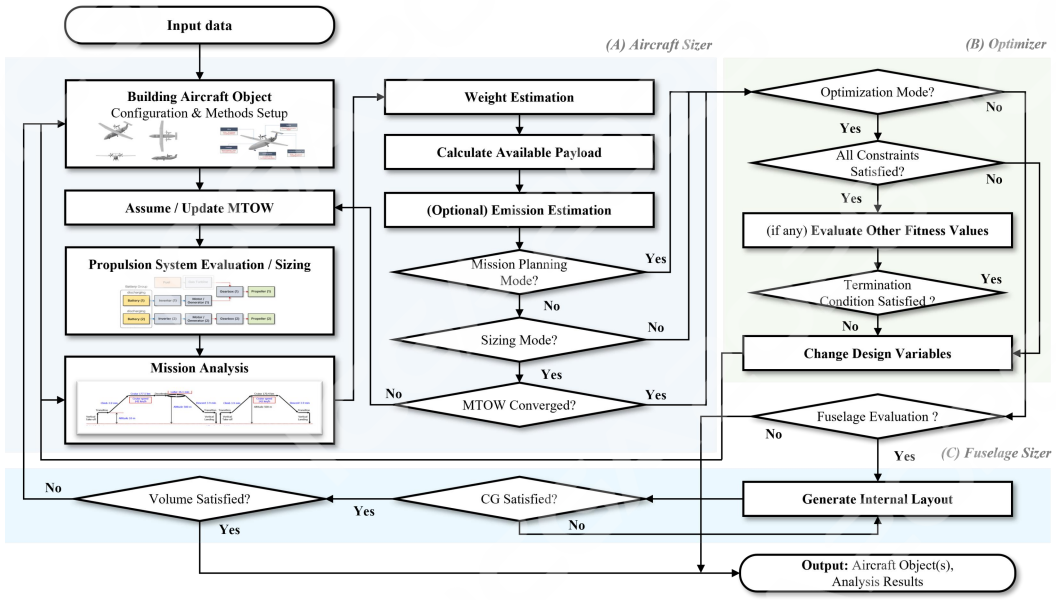


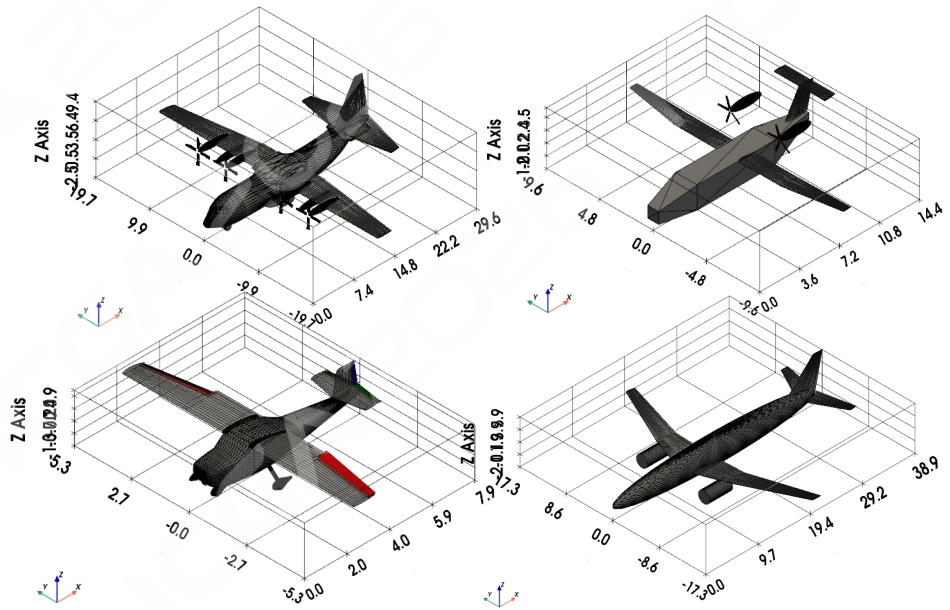
[1] Lee, D., Lim, D., & Yee, K. (2022). Generic design methodology for vertical takeoff and landing aircraft with hybrid-electric propulsion. Journal of Aircraft, 59(2), 278-292.
[2] Chatelain, B., Kim, J., & Yee, K. (2026). An Object-Oriented Framework for Flexible Design of Fixed-Wing Aircraft. In AIAA SCITECH 2026 Forum (p. 2649).
[3] Kim, J., Park, S., Chatelain, B., Jeong, S., Ryu, J., Joo, S., & Yee, K. (2026). AMADEVS: A Generalized Aircraft Modeling & Analysis Environment for Multidisciplinary Design. In AIAA SCITECH 2026 Forum (p. 1687).

AMADEVS as a Framework Builder

- AMADEVS can also serve as a foundation **for building higher-level design and analysis frameworks** rather than operating only as a stand-alone tool

1. RISPECT III: Rotorcraft System Design & Performance Analysis Framework
   - ✓ RISPECT+[1] has been reimplemented in an object-oriented manner on top of the AMADEVS architecture.
2. ASPECT: Fixed-Wing Aircraft System Design & Performance Analysis Framework[2]
   - ✓ A framework developed under AMADEVS for design, performance analysis, and environmental impact assessment of fixed-wing aircraft





[1] Lee, D., Lim, D., & Yee, K. (2022). Generic design methodology for vertical takeoff and landing aircraft with hybrid-electric propulsion. Journal of Aircraft, 59(2), 278-292.
[2] Chatelain, B., Kim, J., & Yee, K. (2026). An Object-Oriented Framework for Flexible Design of Fixed-Wing Aircraft. In AIAA SCITECH 2026 Forum (p. 2649).
[3] Kim, J., Park, S., Chatelain, B., Jeong, S., Ryu, J., Joo, S., & Yee, K. (2026). AMADEVS: A Generalized Aircraft Modeling & Analysis Environment for Multidisciplinary Design. In AIAA SCITECH 2026 Forum (p. 1687).



LITERATURE REVIEW & RESEARCH QUESTIONS

MOTIVATION & OBJECTIVES

Why Existing Approaches [※1] Still Leave the Bottlenecks

"Methodology-Level Bottleneck"

Bottleneck 1. Architecture-Workflow Coupling

"Execution-Level Bottleneck"

Bottleneck 2. Intensive Human Labor

Classic → Modern

Category A

Monolithic/Hardcoded Frameworks

- Vehicle topology and sequential analysis steps are rigidly hardcoded together.
- Entirely incapable of handling novel configurations without code rewrites.

e.g. FLOPS, GASP, NDARC, RISPECT+, etc.

Category B

Object-Oriented Approaches

- Introduced OOP(object-oriented programming) for improved modularity and model representation
- But data connections remain implicit in predefined scripts, bottlenecking when topology changes.

e.g. SUAVE, RCAIDE, OpenAD, pyACDT, MICADO, Gascon, RISPECT III/ASPECT, etc.

Category C

Orchestration-Centric MDAO Tools

- Strong at orchestrating complex MDAO. (providing analytic derivative computation, and distributed process integration)
- Yet, they only solve or connect equations—leaving the mathematical reformulation of residual to human engineers.

e.g. OpenMDAO, Aviary, RCE, Condor, etc.

Category D

Graph-Based Workflow Formulators

- Succeed in dynamically routing and assembling workflows based on architectural graphs, mitigating manual re-wiring.
- But the governing residuals, and architecture-specific reformulation remains manual.

e.g. KADMOS, ADORE (AGILE 4.0)

- **Existing AI-agent-based studies[※2] mainly automate tool-specific operation, but do not address architecture-level MDAO workflow formulation.**

※ References were presented in Appendix-Z01

[1] M. Alder, E. Moerland, J. Jepsen and B. Nagel. Recent Advances in Establishing a Common Language for Aircraft Design with CPACS. Aerospace Europe Conference 2020, Bordeaux, France, 2020.

MOTIVATION & OBJECTIVES

Research Questions

How can architecture-variable aircraft design be supported while reducing both configuration-specific tool dependence and human-intensive exploration effort?

Problems	Architecture-Workflow Coupling	Intensive Human Labor
Questions	RQ1. How can architecture-workflow coupling be fundamentally reduced in MDAO of various aircraft architectures?	RQ2. How can the intensive human labor required for system modeling & MDAO execution be reduced while preserving human supervision?
Needs	Architecture-Agnostic Design Foundation	AI-Agent-Operable Methodology



PROBLEM FORMULATION

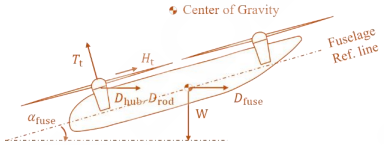
Bottleneck 1. Methodology-Level Problem: Architecture-MDAO Coupling - Example 1: Equations of Motion (EOM)

- **Architecture-dependent formulation of EOM**

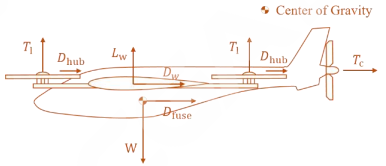
$$\dot{\boldsymbol{x}}_{\mathcal{A}} = \boldsymbol{F}_{\mathcal{A}}(\boldsymbol{x}_{\mathcal{A}}, \boldsymbol{u}_{\mathcal{A}}; \mathcal{M}_{\mathcal{A}}, \boldsymbol{c})$$

$\mathcal{A}$: aircraft architecture
$\mathcal{M}_{\mathcal{A}}$: architecture-specific system model
$\boldsymbol{x}_{\mathcal{A}}$: architecture-dependent state vector
$\boldsymbol{u}_{\mathcal{A}}$: architecture-dependent input/control vectc
$\boldsymbol{c}$: operating condition
$\boldsymbol{F}_{\mathcal{A}}$: architecture-dependent dynamics function

$$\mathcal{A}_{\mathrm{MC}}: \quad \dot{\boldsymbol{x}}_{\mathrm{MC}} = \boldsymbol{F}_{\mathrm{MC}}(\boldsymbol{x}_{\mathrm{MC}}, \boldsymbol{u}_{\mathrm{MC}}; \mathcal{M}_{\mathrm{MC}}, \boldsymbol{c}),$$
$$\boldsymbol{u}_{\mathrm{MC}} = \begin{bmatrix} T_1 & \cdots & T_{N_r} \end{bmatrix}^{\mathsf{T}},$$
$$\begin{bmatrix} \boldsymbol{F} \\ \boldsymbol{M} \end{bmatrix}_{\mathrm{MC}} = \sum_{i=1}^{N_r} \begin{bmatrix} \boldsymbol{R}_i \boldsymbol{f}_{r,i}(T_i) \\ \boldsymbol{r}_i \times \boldsymbol{R}_i \boldsymbol{f}_{r,i}(T_i) + \boldsymbol{m}_{r,i}(T_i) \end{bmatrix} + \begin{bmatrix} \boldsymbol{F}_g \\ \boldsymbol{0} \end{bmatrix}$$

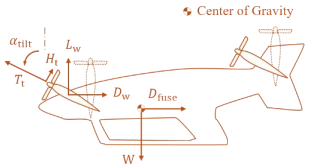


$$\mathcal{A}_{\mathrm{LC}}: \quad \dot{\boldsymbol{x}}_{\mathrm{LC}} = \boldsymbol{F}_{\mathrm{LC}}(\boldsymbol{x}_{\mathrm{LC}}, \boldsymbol{u}_{\mathrm{LC}}; \mathcal{M}_{\mathrm{LC}}, \boldsymbol{c}),$$
$$\boldsymbol{u}_{\mathrm{LC}} = \begin{bmatrix} T_{L,1} & \cdots & T_{L,N_L} & T_p & \delta_e & \delta_a & \delta_r \end{bmatrix}^{\mathsf{T}},$$
$$\begin{bmatrix} \boldsymbol{F} \\ \boldsymbol{M} \end{bmatrix}_{\mathrm{LC}} = \sum_{i=1}^{N_L} \Phi_{r,i}(T_{L,i}) + \Phi_p(T_p) + \Phi_w(\alpha, \delta_a) + \Phi_t(\alpha, \delta_e, \delta_r) + \Phi_b(V) + \begin{bmatrix} \boldsymbol{F}_g \\ \boldsymbol{0} \end{bmatrix}$$



- **Architecture-EOM coupling affects to reconfiguration of:**
  - ✓ Trim analysis
  - ✓ Stability analysis
  - ✓ Control design
  - ✓ Flying/handling qualities assessment, etc.

$$\mathcal{A}_{\mathrm{TR}}: \quad \dot{\boldsymbol{x}}_{\mathrm{TR}} = \boldsymbol{F}_{\mathrm{TR}}(\boldsymbol{x}_{\mathrm{TR}}, \boldsymbol{u}_{\mathrm{TR}}; \mathcal{M}_{\mathrm{TR}}, \boldsymbol{c}),$$
$$\boldsymbol{u}_{\mathrm{TR}} = \begin{bmatrix} T_1 & \cdots & T_{N_t} & \beta_1 & \cdots & \beta_{N_t} & \boldsymbol{\delta}^{\mathsf{T}} \end{bmatrix}^{\mathsf{T}},$$
$$\begin{bmatrix} \boldsymbol{F} \\ \boldsymbol{M} \end{bmatrix}_{\mathrm{TR}} = \sum_{i=1}^{N_t} \begin{bmatrix} \boldsymbol{R}_i(\beta_i) \boldsymbol{f}_{r,i}(T_i) \\ \boldsymbol{r}_i \times \boldsymbol{R}_i(\beta_i) \boldsymbol{f}_{r,i}(T_i) + \boldsymbol{m}_{r,i}(T_i, \beta_i) \end{bmatrix} + \Phi_{\mathrm{aero}}(\alpha, \boldsymbol{\delta}) + \begin{bmatrix} \boldsymbol{F}_g \\ \boldsymbol{0} \end{bmatrix}$$



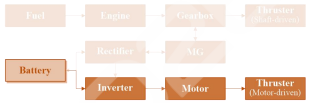
Changing aircraft architecture(configuration) ➔ EOM reformulation ➔ Rewriting of MDAO workflow

Bottleneck 1. Methodology-Level Problem: Architecture-MDAO Coupling - Example 2: Energy Balance Analysis

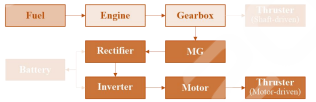
- **Architecture-dependent formulation of EB**

$$\boldsymbol{g}_{\mathcal{A}}(\boldsymbol{p}_{\mathcal{A}};\mathcal{N}_{\mathcal{A}},\boldsymbol{d}) = \left[\sum_{j\in\mathcal{P}^{-}_{\mathcal{A}}(i)} P_{j\to i} - \sum_{k\in\mathcal{P}^{+}_{\mathcal{A}}(i)} P_{i\to k} - P_{\mathrm{loss},i}\right]_{i\in\mathcal{V}_{\mathcal{A}}} = \mathbf{0}$$

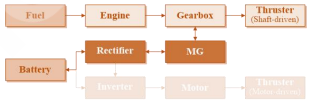
$\mathcal{V}_{\mathcal{A}}$: active energy-node set induced by architecture $\mathcal{A}$
g_i : local power-balance residual at node i
$\mathcal{P}^{-}_{\mathcal{A}}(i)$: predecessor set of node i
$\mathcal{P}^{+}_{\mathcal{A}}(i)$: successor set of node i
$P_{j\to i},\ P_{i\to k}$: directed power flows
$P_{\mathrm{loss},i}$: node-level conversion or transmission loss



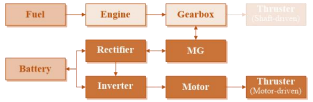
$$\mathcal{A}_{\mathrm{FB}}: \quad \boldsymbol{g}_{\mathrm{FB}}(\boldsymbol{p}_{\mathrm{FB}};\mathcal{N}_{\mathrm{FB}},\boldsymbol{d}) = \mathbf{0},$$
$$\boldsymbol{p}_{\mathrm{FB}} = \begin{bmatrix} P_{\mathrm{bat}} & P_{\mathrm{inv},1} & \cdots & P_{\mathrm{mot},N} & P_{\mathrm{shaft},1} & \cdots & P_{\mathrm{shaft},N} \end{bmatrix}^{\mathsf{T}},$$
$$P_{\mathrm{bat}} = \sum_{i=1}^{N} \frac{P_{\mathrm{shaft},i}}{\eta_{\mathrm{bat}}\eta_{\mathrm{inv},i}\eta_{\mathrm{mot},i}}, \qquad P_{\mathrm{shaft},i} = d_i$$



$$\mathcal{A}_{\mathrm{SH}}: \quad \boldsymbol{g}_{\mathrm{SH}}(\boldsymbol{p}_{\mathrm{SH}};\mathcal{N}_{\mathrm{SH}},\boldsymbol{d}) = \mathbf{0},$$
$$\boldsymbol{p}_{\mathrm{SH}} = \begin{bmatrix} P_{\mathrm{fuel}} & P_{\mathrm{eng}} & P_{\mathrm{gen}} & P_{\mathrm{bus}} & P_{\mathrm{bat}} & P_{\mathrm{shaft},1:N} \end{bmatrix}^{\mathsf{T}},$$
$$P_{\mathrm{bus}} = \eta_{\mathrm{gen}}\eta_{\mathrm{eng}}P_{\mathrm{fuel}} + \eta_{\mathrm{bat}}P_{\mathrm{bat}},$$
$$P_{\mathrm{bus}} = \sum_{i=1}^{N} \frac{P_{\mathrm{shaft},i}}{\eta_{\mathrm{inv},i}\eta_{\mathrm{mot},i}}, \qquad P_{\mathrm{shaft},i} = d_i$$



$$\mathcal{A}_{\mathrm{PH}}: \quad \boldsymbol{g}_{\mathrm{PH}}(\boldsymbol{p}_{\mathrm{PH}};\mathcal{N}_{\mathrm{PH}},\boldsymbol{d}) = \mathbf{0},$$
$$\boldsymbol{p}_{\mathrm{PH}} = \begin{bmatrix} P_{\mathrm{fuel}} & P_{\mathrm{shaft,eng}} & P_{\mathrm{bat}} & P_{\mathrm{shaft,em}} & P_{\mathrm{shaft}} \end{bmatrix}^{\mathsf{T}},$$
$$P_{\mathrm{shaft}} = P_{\mathrm{shaft,eng}} + P_{\mathrm{shaft,em}}, \qquad P_{\mathrm{shaft}} = d,$$
$$P_{\mathrm{shaft,eng}} = \eta_{\mathrm{gb}}\eta_{\mathrm{eng}}P_{\mathrm{fuel}}, \qquad P_{\mathrm{shaft,em}} = \eta_{\mathrm{mot}}\eta_{\mathrm{inv}}\eta_{\mathrm{bat}}P_{\mathrm{bat}}$$



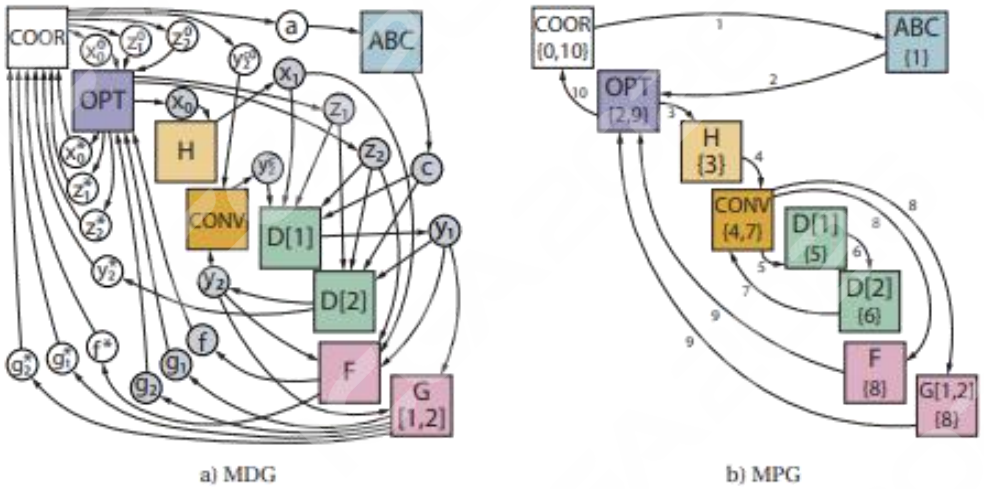
$$\mathcal{A}_{\mathrm{SPH}}: \quad \boldsymbol{g}_{\mathrm{SPH}}(\boldsymbol{p}_{\mathrm{SPH}};\mathcal{N}_{\mathrm{SPH}},\boldsymbol{d}) = \mathbf{0},$$
$$\boldsymbol{p}_{\mathrm{SPH}} = \begin{bmatrix} P_{\mathrm{fuel}} & P_{\mathrm{shaft,eng}} & P_{\mathrm{gen}} & P_{\mathrm{bat}} & P_{\mathrm{bus}} & P_{\mathrm{shaft,em}} & P_{\mathrm{shaft}} \end{bmatrix}^{\mathsf{T}},$$
$$P_{\mathrm{bus}} = \eta_{\mathrm{gen}}\eta_{\mathrm{eng}}P_{\mathrm{fuel}} + \eta_{\mathrm{bat}}P_{\mathrm{bat}},$$
$$P_{\mathrm{shaft}} = \eta_{\mathrm{gb}}\eta_{\mathrm{eng}}P_{\mathrm{fuel}} + \eta_{\mathrm{mot}}\eta_{\mathrm{inv}}P_{\mathrm{bus}}, \qquad P_{\mathrm{shaft}} = d$$

- **Architecture-EB coupling affects to reconfiguration of:**
  - ✓ Propulsion system sizing
  - ✓ Performance analysis / mission analysis
  - ✓ Failure mode analysis, etc.

Changing aircraft propulsion architecture ⇒ EB reformulation ⇒ Rewriting of MDAO workflow

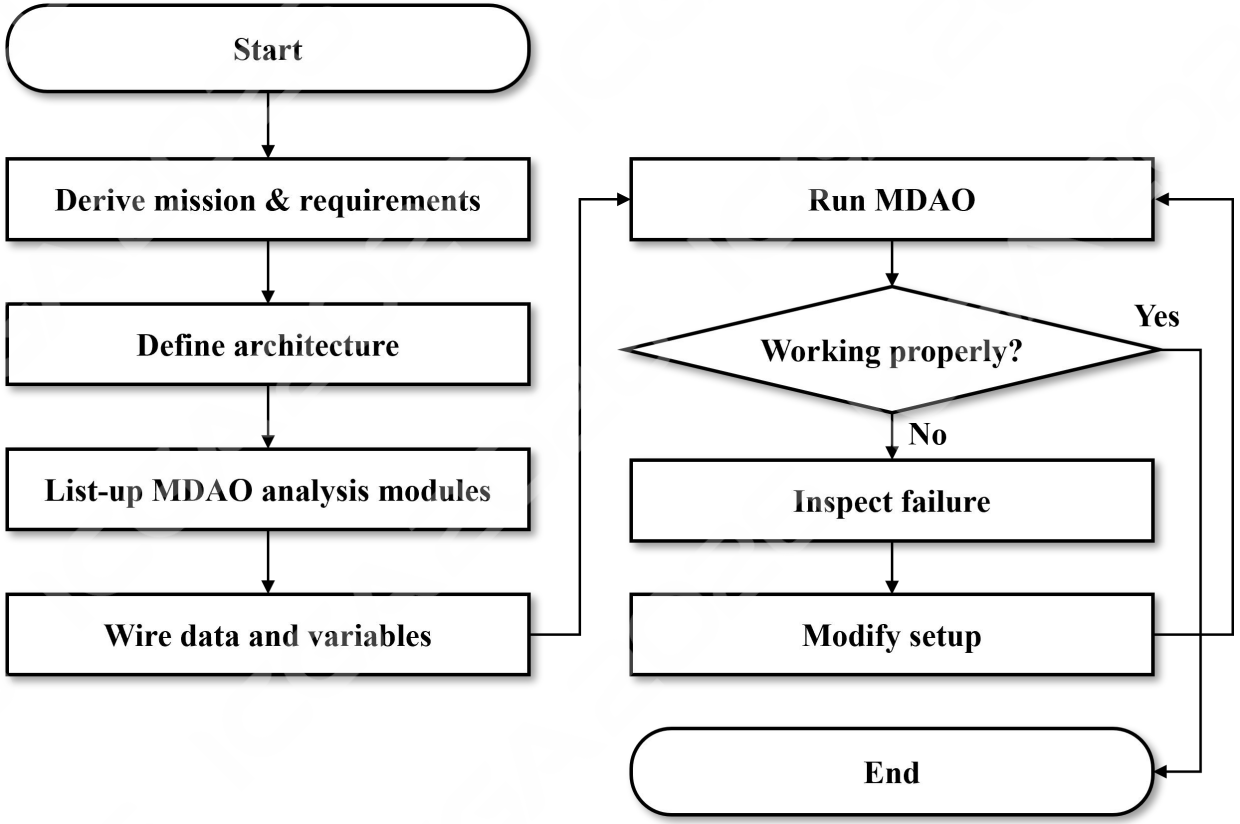
Bottleneck 2. Execution-Level Problem: Intensive Human Labor - Repeated Human Operations

- Aircraft MDAO requires repeated human operations.
  - ✓ Setup
  - → Execution
  - → Diagnosis
  - → Repair
  - → Iteration



a) MDG b) MPG

Image source: Ref. [1]



The bottleneck is not only "having analysis tools" but also "configuring them."

[1] van Gent, I. (2019). Agile MDAO systems: a graph-based methodology to enhance collaborative multidisciplinary design.

Bottleneck 2. Execution-Level Problem: Intensive Human Labor – Difficulties of Automation

- **Hidden Procedural Operation**
  - ✓ These operations are difficult for AI agents to operate & assist because they are often implicitly embedded in procedural workflow code.

Hidden procedural script	Consequence for AI agent
State is implicit.	Hard to inspect current design state.
Operation target is code.	Agent must edit scripts directly.
Context is distributed.	Higher risk of hallucination.
Side effects are implicit.	Modification impact is hard to predict.
Failures are unstructured.	Repair target is ambiguous.

Design operation = code editing

workflow_script.py

```
# setup is executable but implicit
define_vehicle(...)
configure_scenario (...)
select_methods (...)
wire_dataflow (...)
build_unknowns (...)
assemble_residuals (...)
run_solver (...)
if failed:
    manual_debug_and_patch(...)
```

Hard to manage task context

⇒ **Expensive** (due to high token consumption)

⇒ **Risk of Hallucination**※ ⇒ **Low reliability** of work

※ Hallucination refers to an AI model generating plausible-sounding but incorrect or ungrounded information.

Hidden procedural operation limits AI-agent operated aircraft MDAO.

PROBLEM FORMULATION

Methodological Requirements: "What Must the Methodology Provide?"

- These decoupling targets define the methodological requirements.

For architecture-workflow decoupling:

✓ **R1. Unified Architecture Representation**

- Represent heterogeneous aircraft configurations under a single formal model.

→ Avoids per-configuration data structures

✓ **R2. Model–Viewpoint Separation**

- Separate what the aircraft is from how it is analyzed.

→ Avoids modifying the vehicle model for each analysis.

✓ **R3. Topology-Invariant Workflow Assembly**

- Derive variables, residuals, couplings, and active components using common rules.

→ Avoids topology-specific trim/energy workflow code

✓ **R4. Explicit Model-Method Decomposition**

- Select discipline methods independently of the architecture representation.

→ Avoids hardwired component-method coupling and workflow rewrites during fidelity changes.

For AI-agent operability:

✓ **R5. Machine-Readable Design State**

- AI agents inspect architecture, analysis intent, methods, constraints, and results

→ Avoids hidden assumptions

✓ **R6. Declarative Modification Interface**

- Agents modify explicit design and analysis declarations rather than procedural workflow code.

→ Avoids code-level agent intervention

✓ **R7. Structured Diagnostics and Repair Guidance**

- Return validation and analysis failures with location, cause, and suggested repair.

→ Avoids unstructured failure debugging

✓ **R8. Human-Oversight-Compatible Iteration**

- Keep AI actions traceable, reviewable, and correctable by engineers.

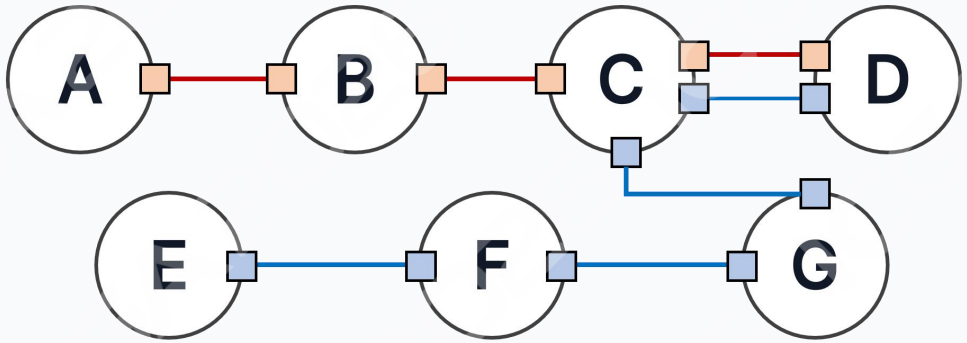
→ Avoids black-box automation and loss of engineering accountability



GRAPH-BASED METHODOLOGY OVERVIEW

GRAPH-BASED ARCHITECTURE-AGNOSTIC FOUNDATION

Foundation Overview : Graph-Based Representation of Architecture & Workflow



System graph

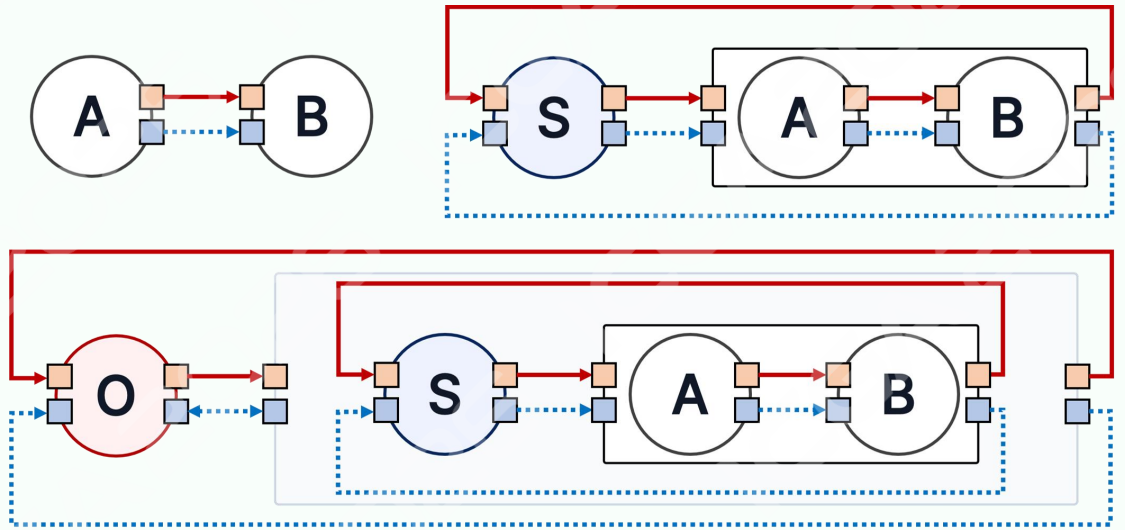
$$\mathcal{G}_{\mathcal{A}} = (\mathcal{V}_{\mathcal{A}}, \mathcal{E}_{\mathcal{A}}, \mathcal{P}_{\mathcal{A}}, \tau_{\mathcal{A}})$$

$\mathcal{V}_{\mathcal{A}}$: component and system−element nodes

$\mathcal{E}_{\mathcal{A}}$: relationship edges between nodes

$\mathcal{P}_{\mathcal{A}}$: typed ports attached to nodes

A~G $\tau_{\mathcal{A}}$: typing function for nodes, edges, and ports



Workflow graph

$$\mathcal{G}_{\mathcal{W}} = (\mathcal{V}_{\mathcal{W}}, \mathcal{E}_{\mathcal{W}}, \mathcal{P}_{\mathcal{W}}, \tau_{\mathcal{W}})$$

$\mathcal{V}_{\mathcal{W}}$: analysis and driver nodes

$\mathcal{E}_{\mathcal{W}}$: typed workflow relations

$\mathcal{P}_{\mathcal{W}}$: typed interfaces attached to workflow nodes

A~C / S / O $\tau_{\mathcal{W}}$: typing function for nodes, edges, and ports

GRAPH-BASED ARCHITECTURE-AGNOSTIC FOUNDATION

Why a Graph-Based 'System' Representation? From component lists to explicit relationships

- An aircraft architecture is more than a list of components.
- What matters:
  - ✓ where it is mounted / how it transfers energy / what it controls / what loads it carries, etc.
- To model these *interactions* between components:

⇒ Architecture is defined by how components are mounted, connected, powered, and controlled as a graph.

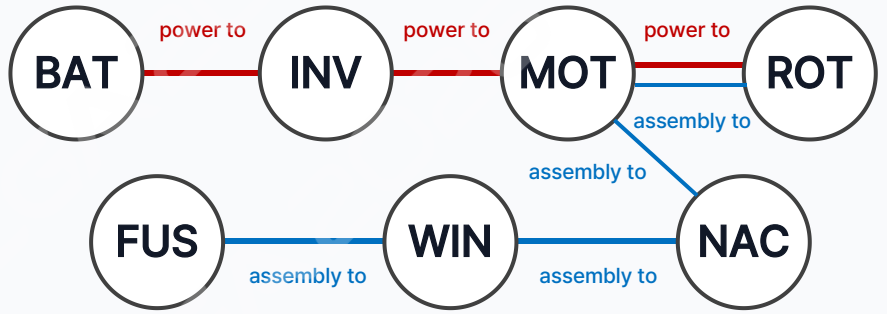
⇒ Components are nodes, and interactions are edges.

graph = nodes + edges

Architecture as a list

fuselage	wing
rotor	motor
battery	controller

Architecture as a graph



System model is modeled as an 'executable graph' to simulate its system-level behavior (such as energy-flow, kinematics)

Component Node Semantics

- Component nodes declare what exist.
  - ✓ Not how it must always be analyzed
- A node contains: data + roles + ports + analysis methods + variables

architecture graph

$$\mathcal{G}_{\mathcal{A}} = (\mathcal{V}_{\mathcal{A}}, \mathcal{E}_{\mathcal{A}}, \mathcal{P}_{\mathcal{A}}, \tau_{\mathcal{A}})$$

$\mathcal{V}_{\mathcal{A}}$: component and system−element nodes

$\mathcal{E}_{\mathcal{A}}$: relationship edges between nodes

$\mathcal{P}_{\mathcal{A}}$: typed ports attached to nodes

$\tau_{\mathcal{A}}$: typing function for nodes, edges, and ports

A~G

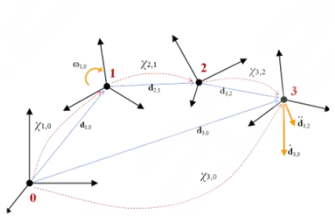
Arbitrary system model

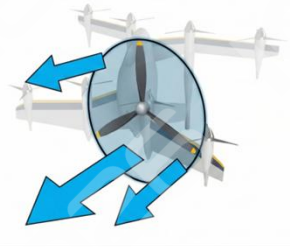


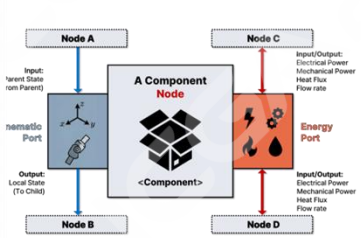
An arbitrary 'component' (e.g. proprotor)

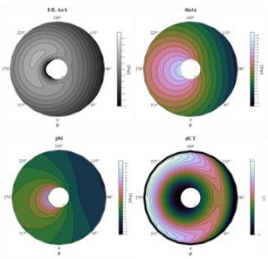
A Component Node (e.g. proprotor)

Physical Properties	Roles/Sub-roles	Ports	Analysis Methods	Controllables
• Position & Rotation • Weight / Inertia • Component CG location • Material properties • Aerodynamic properties	• Thruster • Forces & Moment Contributor • Energy Sink, etc.	• Kinematic port • Energy port • Control port	• Aerodynamic methods (MT, BET, etc.) • Structure methods (beam model, etc.) • Weight estimation methods, etc.	• Collective pitch • RPM • Thrust (MT)











GRAPH-BASED ARCHITECTURE-AGNOSTIC FOUNDATION

Edge & Port Semantics

- Edges: relationship between nodes (components)

  Relation type, source / target ports, mount transform, flow direction, command variable, etc.

  - ✓ Assembly: modeling kinematic relationship
  - ✓ Energy: modeling energy-flow route
  - ✓ Control: modeling control signals
- Ports: Standardized Interfaces
  - ✓ Each component has interfaces:
    - Kinematic port
    - Energy port
- Supports state propagation and multi-domain energy flow analysis
- Enables consistent system integration modeling across arbitrary architectures
  - ✓ Supports structured diagnosis

architecture graph

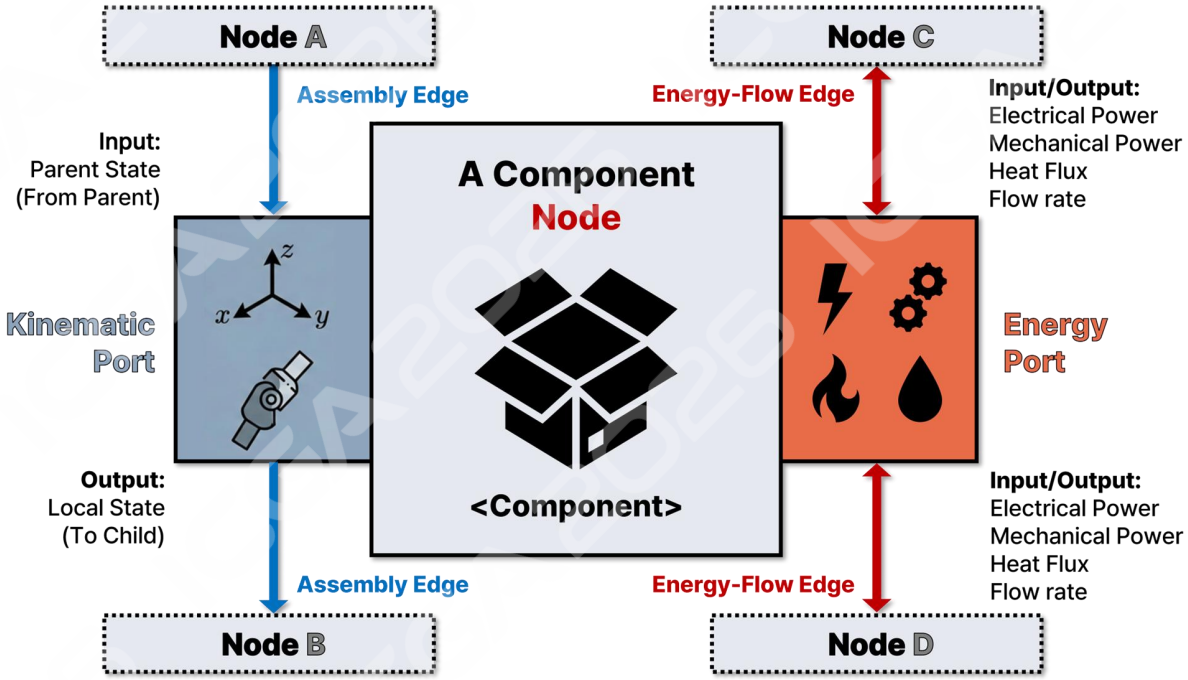
$$\mathcal{G}_{\mathcal{A}} = (\mathcal{V}_{\mathcal{A}}, \mathcal{E}_{\mathcal{A}}, \mathcal{P}_{\mathcal{A}}, \tau_{\mathcal{A}})$$

$\mathcal{V}_{\mathcal{A}}$: component and system−element nodes

$\mathcal{E}_{\mathcal{A}}$: relationship edges between nodes

$\mathcal{P}_{\mathcal{A}}$: typed ports attached to nodes

$\tau_{\mathcal{A}}$: typing function for nodes, edges, and ports



⇛ **Architecture is represented with structured interface semantics that can be inspected and validated.**

ARCHITECTURE-AGNOSTIC FOUNDATION

Topology as Relations: Kinematic Networks

- Aircraft geometry is constructed by **assembling components**, not filling templates.
- **Assembly (kinematic) edges define:**
  - ✓ relative position, mount orientation, degree of freedom of joint
  - ✓ force/moment transform paths, etc.

⇒ Automatic/recursive coordinate transformation is supported.

- [Examples]
  - ✓ Tiltrotor aircraft assembly, etc.

architecture graph

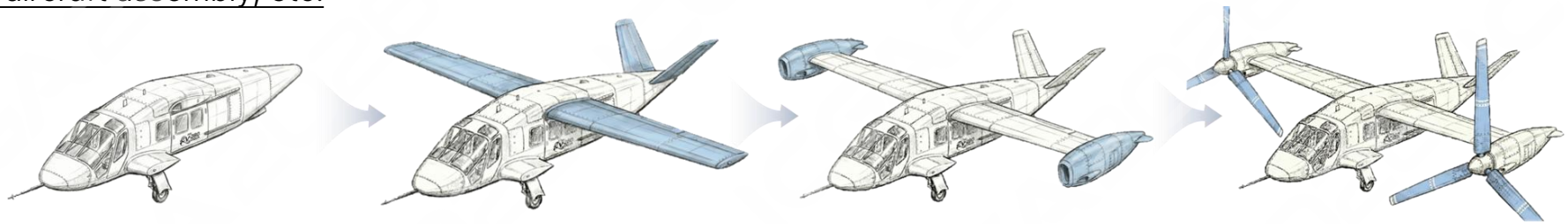
$$\mathcal{G}_{\mathcal{A}} = (\mathcal{V}_{\mathcal{A}}, \mathcal{E}_{\mathcal{A}}, \mathcal{P}_{\mathcal{A}}, \tau_{\mathcal{A}})$$

$\mathcal{V}_{\mathcal{A}}$: component and system−element nodes

$\mathcal{E}_{\mathcal{A}}$: relationship edges between nodes

$\mathcal{P}_{\mathcal{A}}$: typed ports attached to nodes

$\tau_{\mathcal{A}}$: typing function for nodes, edges, and ports



'Root' 'Parent' 'Child' 'Leaf'

Rotational DOF Rotational DOF

Fuselage → Main Wing → Nacelle → PropRotor

Fuselage → V-Tail Wing

$T_{wing} = T_{Fuselage} T_{offset_{wing}}$

$T_{Nacelle} = T_{wing} R_{tilt}$

$T_{PropRotor} = T_{Nacelle} T_{offset_{PR}}$

$T_{tail} = T_{Fuselage} T_{offset_{tail}}$

T: Transformation Matrix

ARCHITECTURE-AGNOSTIC FOUNDATION

Topology as Relations: Energy Networks

- **Energy and physical flows are modeled explicitly as a network.**
  - ✓ e.g. propulsion system, etc.
- **Energy edges define:**
  - ✓ Physical quantities
    - Mechanical connector
      - torque and shaft power transfer
    - Electrical connector
      - current and electrical power flow
    - Thermal / fluidic connector
      - heat transfer and mass/flow transport
  - ✓ Direction of energy-flow
  - ✓ Losses
  - ✓ Split ratios
    - To model arbitrary hybrid architectures
- [Examples]
  - ✓ (a) Battery-electric architecture
  - ✓ (b) Parallel-hybrid electric architecture

architecture graph

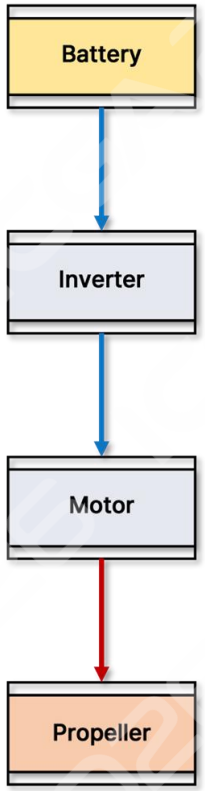
$$\mathcal{G}_{\mathcal{A}} = (\mathcal{V}_{\mathcal{A}}, \mathcal{E}_{\mathcal{A}}, \mathcal{P}_{\mathcal{A}}, \tau_{\mathcal{A}})$$

$\mathcal{V}_{\mathcal{A}}$: component and system−element nodes

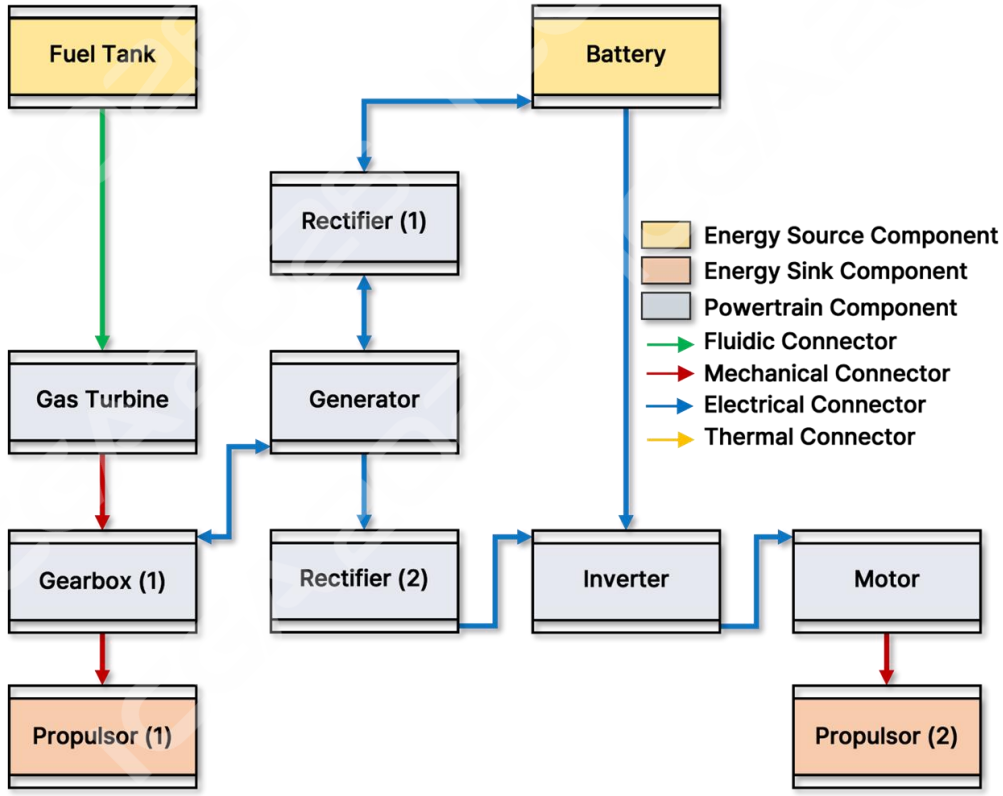
$\mathcal{E}_{\mathcal{A}}$: relationship edges between nodes

$\mathcal{P}_{\mathcal{A}}$: typed ports attached to nodes

A~G $\tau_{\mathcal{A}}$: typing function for nodes, edges, and ports



(a) Battery-Electric Architecture

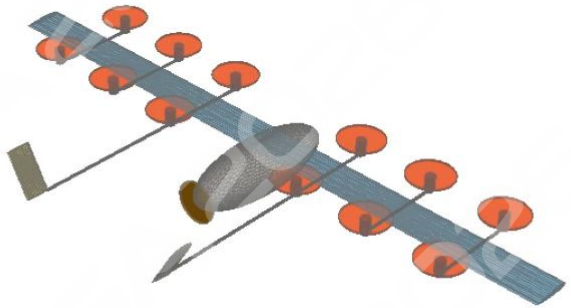


(b) Parallel-Hybrid Electric Architecture

Application III AI-Assisted Aircraft Design – Winged-eVTOL Aircraft Design [video]

- Other configurations explored by AI-Copilot (using Morphological Analysis concept)
  - ✓ Component 'modules' are prepared as 'sub-graph' by architecture-specialist (agent)
  - ⇒ Assembled by morphological matrix element selection

Dimension	Opt 1	Opt 2	Opt 3	Opt 4	Opt 5
Wing Config	None	**Single**	Tandem		
Fuselage	Conv.	**Twin Boom**			
Tail Type	H-tail	**V-tail**	Twin-V	Λ-tail	None
Lift System	**Distributed**	Tilting	Proprotor		
Rotor Count	4	6	8	**12**	
Cruise Prop	**Pusher**	None			



GRAPH-BASED ARCHITECTURE-AGNOSTIC FOUNDATION

Model-Method Decoupling and Dispatch

- **Model**: what the aircraft is
- **Method**: how the response is evaluated

- Analysis methods are selected declaratively (without changing the system model).

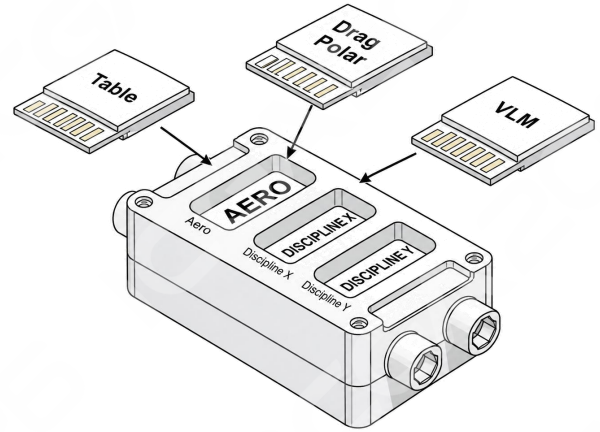
⇒ A system model does not hardwire a single method, fidelity level, or discipline implementation.

⇒ Each component can independently bind analysis methods.

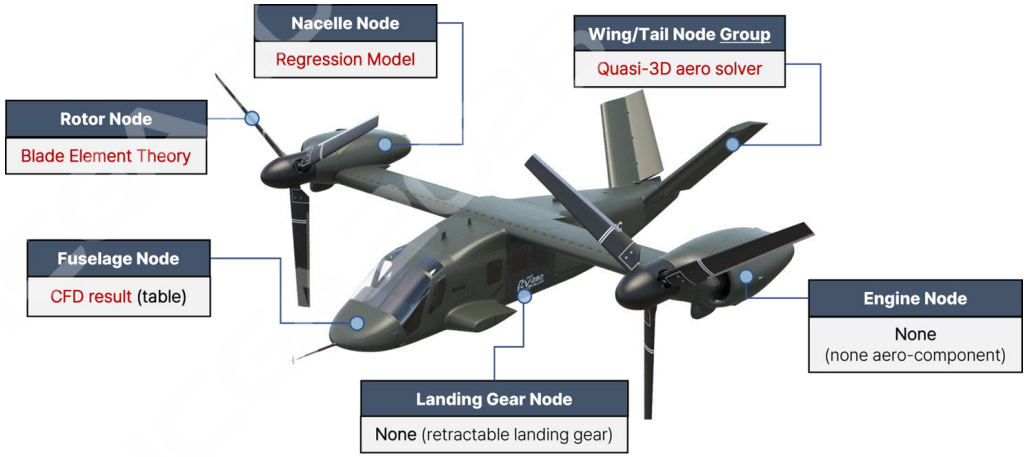
- ✓ Low- to high-fidelity models per component (selective-fidelity)
- ✓ Also supports easy coupling with external solvers

⇒ Supports consistent modeling, analysis & design for various design phase

- ✓ From conceptual- to preliminary-level analysis & design



△ Concept of analysis method dispatch for a 'component'



△ Example of aerodynamic methods dispatch for tiltrotor

ARCHITECTURE-AGNOSTIC FOUNDATION

Model-Method-Scenario Decoupling

- Model: what the aircraft is
- Method: how the response is evaluated
- **Scenario: under what context it is examined**

A "*scenario*" is a declarative specification of analysis context.

- ✓ One aircraft, multiple analytical context
  - e.g. hover, cruise, OEI, and transition change context.
- ✓ Embedding context in the model forces repeated reformulation
- ✓ Embedding method choice in the model couples fidelity to architecture

- ✓ **The vehicle model carries only persistent architectural information.**
- ✓ **Method selection is decoupled from vehicle model.**
- ✓ **Flight condition and mission purpose belong to the an alysis context.**

Model
what the aircraft is
components
relations and ports

⟷

Scenario
under what context
active configuration
condition and objective

⟷

Method
how the physics is evaluated
analysis operator
numerical method / fidelity

Architecture-agnostic analysis requires independent control of model, method, and scenario.

Architecture-Agnostic Problem Formulation of EOM & EB

- **Goal: Architecture-dependent information is declared once and consumed by common assembly rules.**

Conventional: $\mathcal{A} \xrightarrow{\mathcal{R}_{\mathrm{EOM},\mathcal{A}},\,\mathcal{R}_{\mathrm{EB},\mathcal{A}}} (\mathcal{P}_{\mathrm{EOM},\mathcal{A}},\ \mathcal{P}_{\mathrm{EB},\mathcal{A}})$

$\Delta\mathcal{A} \Rightarrow \Delta\mathcal{R}_{\mathrm{EOM},\mathcal{A}},\ \Delta\mathcal{R}_{\mathrm{EB},\mathcal{A}}$

Proposed: $(\mathcal{G}_{\mathcal{A}},\ \sigma,\ \beta) \xrightarrow{\mathcal{R}_{\mathrm{EOM}},\,\mathcal{R}_{\mathrm{EB}}} (\mathcal{P}_{\mathrm{EOM},\mathcal{A}},\ \mathcal{P}_{\mathrm{EB},\mathcal{A}})$

$\Delta\mathcal{A} \Rightarrow \Delta\mathcal{G}_{\mathcal{A}} \Rightarrow \Delta\mathcal{P}_{\mathrm{EOM},\mathcal{A}},\ \Delta\mathcal{P}_{\mathrm{EB},\mathcal{A}} \not\Rightarrow \Delta\mathcal{R}_{\mathrm{EOM}},\ \Delta\mathcal{R}_{\mathrm{EB}}$

Equations of Motion

- ✓ Force and moment balance assembled from graph-declared contributors
  - from active roles, controls, methods, and kinematic relations
  - Inputs: system graph / analysis scenario (including method bindings)

$$m\dot{\boldsymbol{V}} = \sum_{i\in\mathcal{V}_\sigma^{\mathrm{act}}} \boldsymbol{F}_i + \boldsymbol{W}$$

$$\boldsymbol{I}\dot{\boldsymbol{\omega}} + \boldsymbol{\omega}\times(\boldsymbol{I}\boldsymbol{\omega}) = \sum_{i\in\mathcal{V}_\sigma^{\mathrm{act}}} (\boldsymbol{r}_i \times \boldsymbol{F}_i + \boldsymbol{M}_i)$$

$\mathcal{G}_{\mathcal{A}}$: structured architecture model
σ : analysis scenario
β : method binding
$\mathcal{R}_{\mathrm{EOM}}$, $\mathcal{R}_{\mathrm{EB}}$: reusable EOM and energy-balance formulation rules
$\mathcal{P}_{\mathrm{EOM}}$, $\mathcal{P}_{\mathrm{EB}}$: generated EOM and energy-balance problems

$\mathcal{V}_\sigma^{\mathrm{act}}$: scenario-active force/moment contributor set
$\mathcal{X}_{\mathrm{dyn}}(\mathcal{G},\sigma)$: admissible dynamic-state set
$\mathcal{U}_{\mathrm{dyn}}(\mathcal{G},\sigma)$: admissible control/input set
$\boldsymbol{f}_{\mathrm{dyn}}$: assembled nonlinear flight-dynamics right-hand side
$\boldsymbol{x}^*$, $\boldsymbol{u}^*$: analysis point for linearization
$\boldsymbol{A}$, $\boldsymbol{B}$: linearized state and control Jacobians

Architecture-Agnostic Flight Dynamics Analysis

$$(\mathcal{G},\ \sigma) \longmapsto \left\{\mathcal{V}_\sigma^{\mathrm{act}},\ \mathcal{X}_{\mathrm{dyn}}(\mathcal{G},\sigma),\ \mathcal{U}_{\mathrm{dyn}}(\mathcal{G},\sigma)\right\}$$

$$\dot{\boldsymbol{x}} = \boldsymbol{f}_{\mathrm{dyn}}(\boldsymbol{x},\boldsymbol{u};\mathcal{G},\sigma)\,, \qquad [\boldsymbol{A}\ \boldsymbol{B}] = \left[\frac{\partial \boldsymbol{f}_{\mathrm{dyn}}}{\partial \boldsymbol{x}}\ \frac{\partial \boldsymbol{f}_{\mathrm{dyn}}}{\partial \boldsymbol{u}}\right]\Bigg|_{(\boldsymbol{x}^*,\boldsymbol{u}^*)}$$

Input: $\mathcal{G}$, σ, $(\boldsymbol{x}^*,\boldsymbol{u}^*)$
1. $\mathcal{V}_\sigma^{\mathrm{act}} \leftarrow$ activate contributors from σ
2. $\mathcal{X}_{\mathrm{dyn}}(\mathcal{G},\sigma)$, $\mathcal{U}_{\mathrm{dyn}}(\mathcal{G},\sigma) \leftarrow$ admissible state/control sets
3. resolve operating-point, control, and mass/state views from $(\mathcal{G},\sigma,\boldsymbol{x}^*,\boldsymbol{u}^*)$
4. **for each** $i \in \mathcal{V}_\sigma^{\mathrm{act}}$:
   $\mu_i \leftarrow$ force/moment evaluator for contributor i
   $(\boldsymbol{F}_i,\boldsymbol{M}_i) \leftarrow \mu_i(\mathcal{G},\sigma,\boldsymbol{x}^*,\boldsymbol{u}^*)$
5. $\boldsymbol{f}_{\mathrm{dyn}} \leftarrow$ assemble rigid-body dynamics from common-frame loads
6. $(\boldsymbol{A},\boldsymbol{B}) \leftarrow$ linearize $\boldsymbol{f}_{\mathrm{dyn}}$ at $(\boldsymbol{x}^*,\boldsymbol{u}^*)$

METHODOLOGY FOR AI-OPERABILITY

Structured Diagnostics as the Feedback Channel

- Machine-readable failure information enables revision, retry, and explanation
- Without structured diagnostics, the agent loop cannot become repairable.

Structured diagnostics as the feedback channel

What failed	constraint, analysis, or operation
Where	graph element, scenario, or workflow step
Why	machine-readable cause
Suggested repair	next action for the agent or engineer

Diagnostic Loop




```
{
  "type" : "error",
  "code" : "missing_required" ,
  "node_id" : "trim",
  "port" : "W" ,
  "message" :
    "Node 'trim' requires port 'W'
    but no upstream node provides it."
  ,
  "suggestion"
    "Add a module that emits 'W',
    or wire this port to a producer."
}
```

The AI loop becomes repairable only when diagnostics are explicit, localized, and actionable.

METHODOLOGY FOR AI-OPERABILITY

Human-Supervised AI-Assistat Workflow

- AI assistance is bounded by explicit human approval for state-changing actions.
- The target is AI-assisted design under human supervision, not AI-autonomous design.
- Agents reason and evaluate candidate changes, but critical approval authority remains with the engineers.



Agents propose and evaluate; humans retain approval authority for state-changing actions.

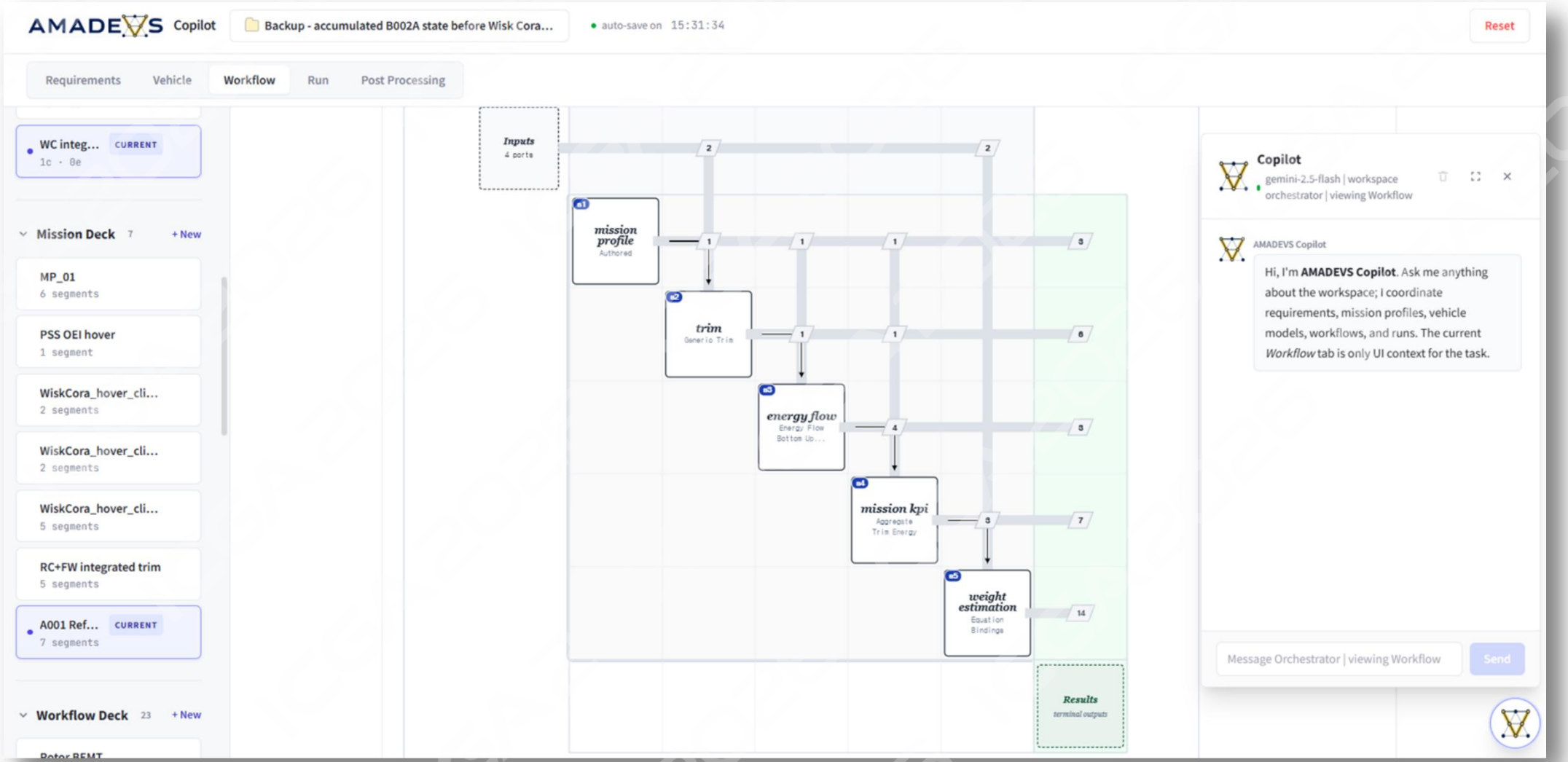
Only after validation, diff inspection, and review the candidate become accepted state.

Human supervision is the boundary that keeps AI assistance from becoming unbounded autonomy.

COMPUTATION REALIZATION: AMADEVS

AMADEVS Web-Based Platform with AI-Copilot (FRONTEND)

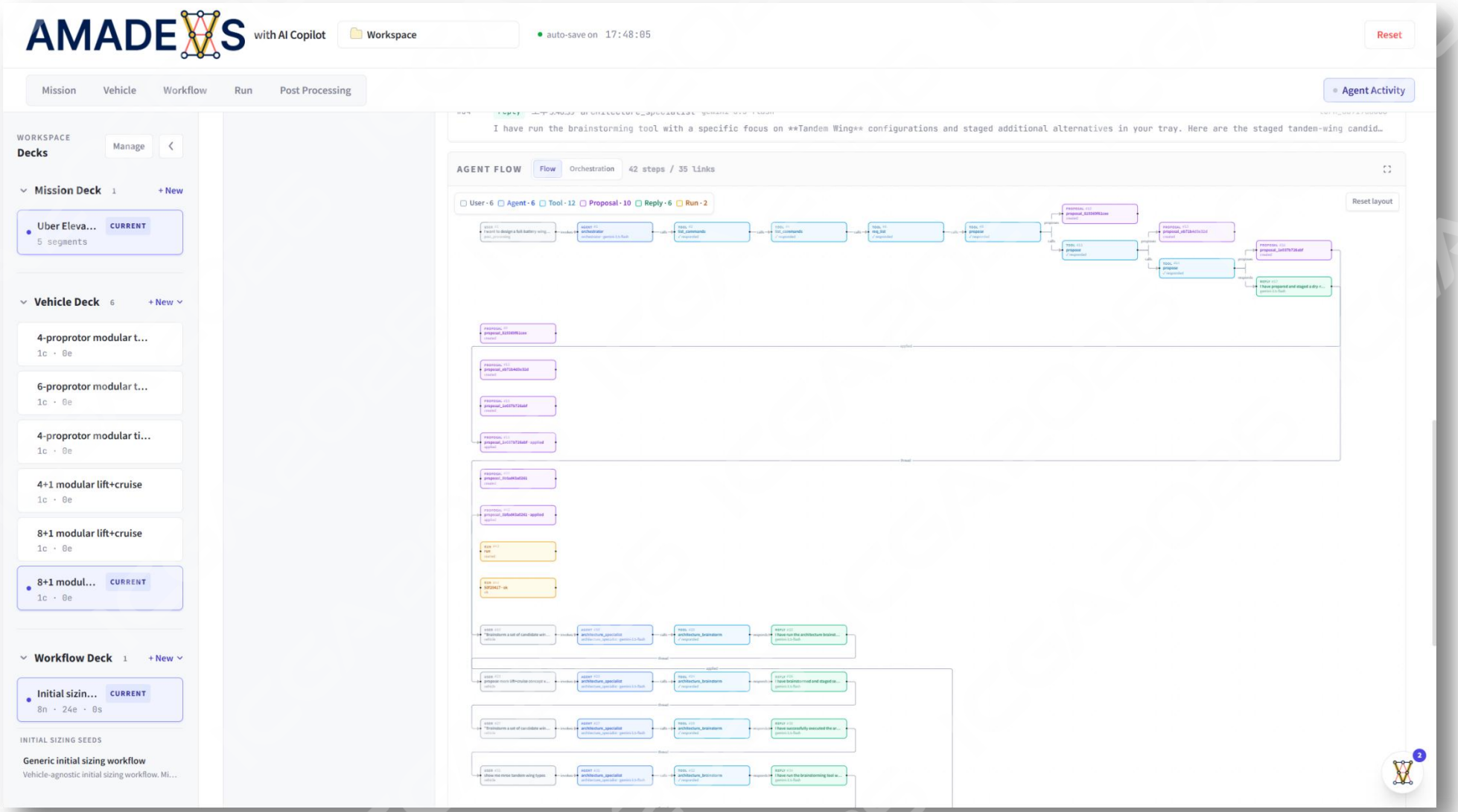
- Requirements & Mission ⇒ Vehicle Modeling ⇒ Workflow Modeling ⇒ Run ⇒ Post Processing



COMPUTATION REALIZATION: AMADEVS

AMADEVS Web-Based Platform with AI-Copilot (FRONTEND)

- Agent Workflow Monitoring (observability)




OTHER APPLICATIONS

Application I — Architecture-Agnostic Trim Analysis of RATO BWB UAS

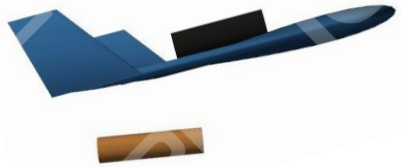
- Rocket-Assisted Take-Off UAS Concept



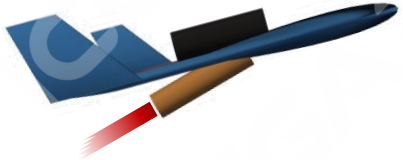
Cruise flight configuration



Transition & climb flight



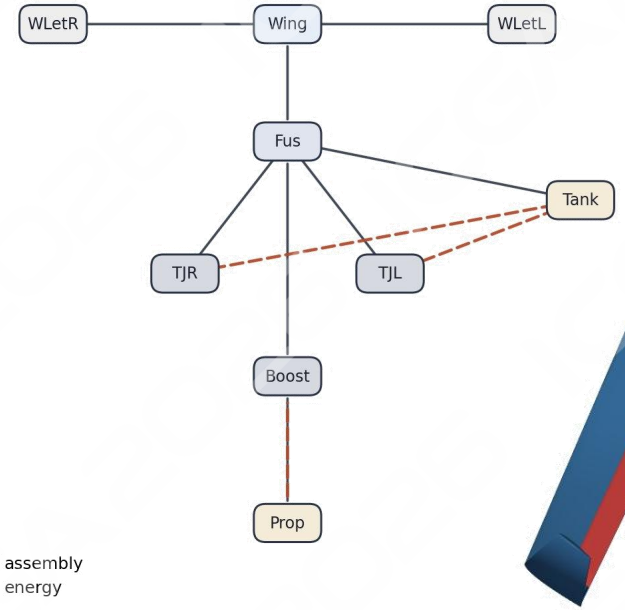
Takeoff configuration with rocket booster

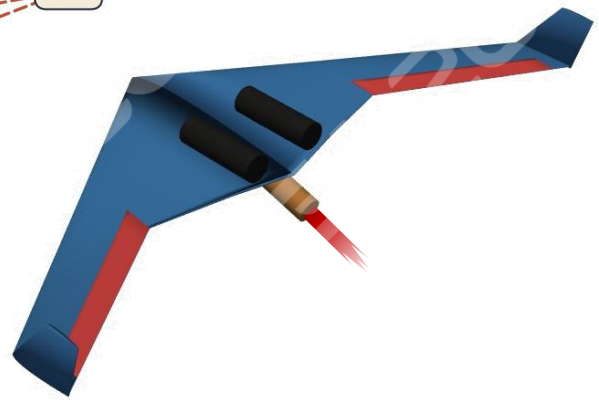


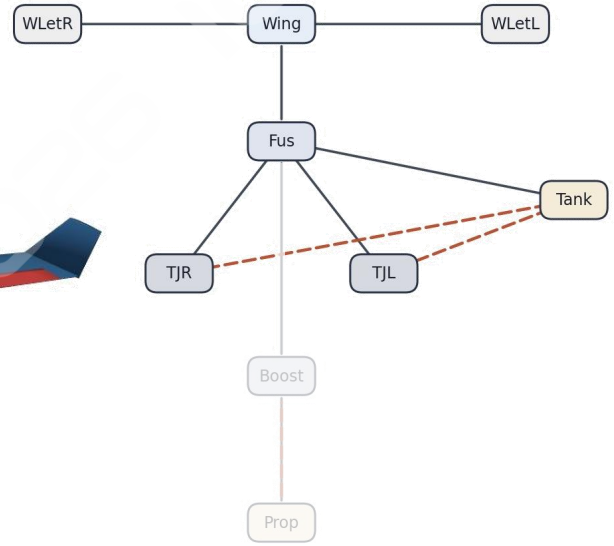
Booster case jettison

- Twin turbojet BWB UAS
  + solid-rocket booster

Parameter	Symbol	Value
Airframe		
Maximum takeoff mass	m_0	130 kg
Empty mass	m_e	70 kg
Wingspan	b	2.10 m
Wing area (planform)	S	0.75 m^2
Aspect ratio	AR	5.88
Mean aerodynamic chord	$\bar{c}$	0.357 m
Wing loading	W/S	1.70 kN/m^2
Flight condition		
Cruise speed	V_c	180 m/s



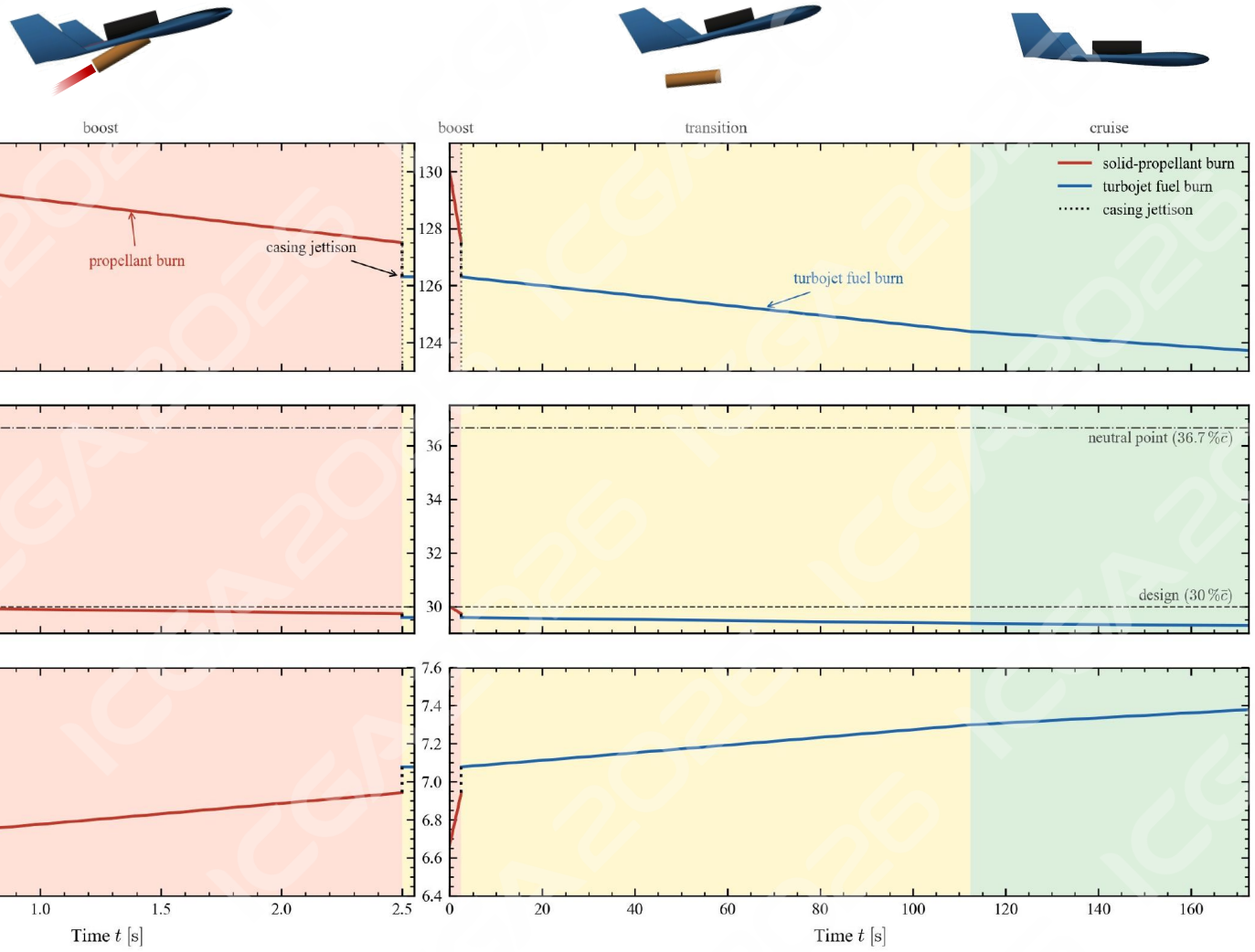


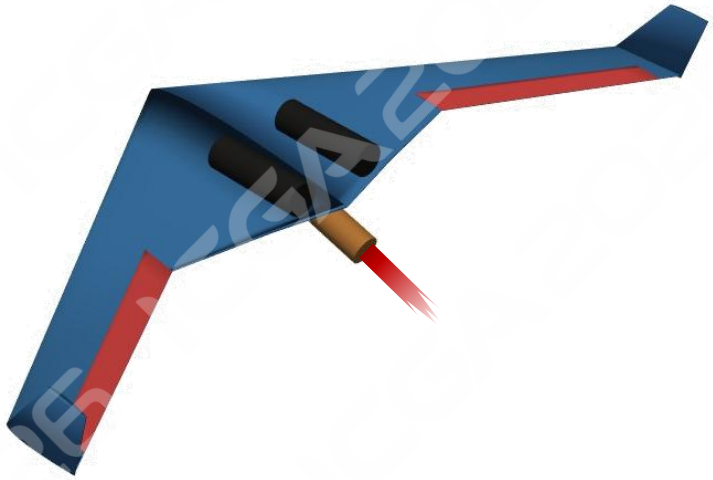


APPLICATIONS & DEMONSTRATION

Application I — Architecture-Agnostic Trim Analysis of RATO BWB UAS

- Trim analysis for eat mission segment was performed successfully without any reformulation.



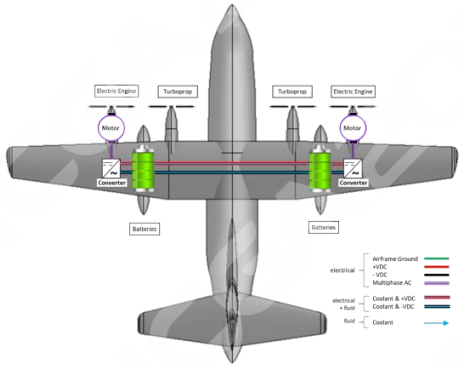


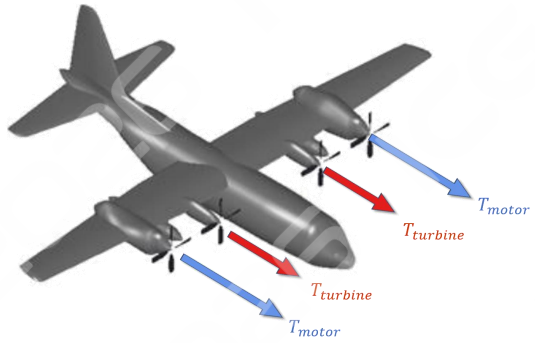
Module	Change scope
Component catalog (booster factory)	New file
Force evaluator (time-variable thrust)	New file
Energy source (propellant consumable)	New file
Evaluator registry	+1 entry
Graph engine	None
Pose propagation	None
Force-summation / trim solver	None
Workflow adapters	None
Energy-balance propagation	None

[1] Pham, Dahlia, et al. "Parametric Modeling and Mission Performance Analysis of a True Parallel Hybrid Turboprop Aircraft for Freighter Operations." *AIAA AVIATION FORUM AND ASCEND 2024*. 2024.

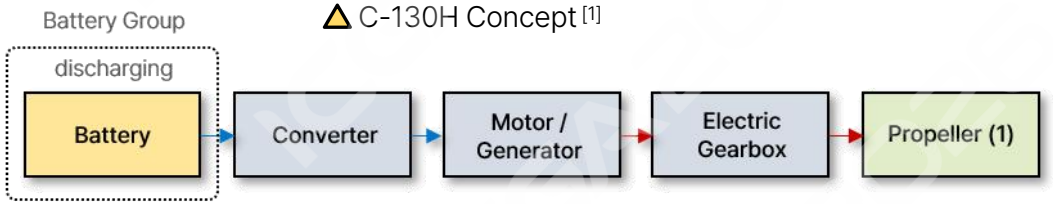
Application II — Architecture-Agnostic Mission Analysis of Cargo Aircraft

- A retrofitted C-130H freighter aircraft with a parallel hybrid electric powertrain[1]
  - ✓ **MTOW**: 155,000 lb. / **Payload** : 25,000 lb.
  - ✓ Outboard turboprop engines replaced with electric motors.
- Four propulsion system architectures were studied
  - ✓ V1 (battery-electric) / V2 (series hybrid) / V3 (parallel hybrid) / **V4 (true-parallel hybrid)**

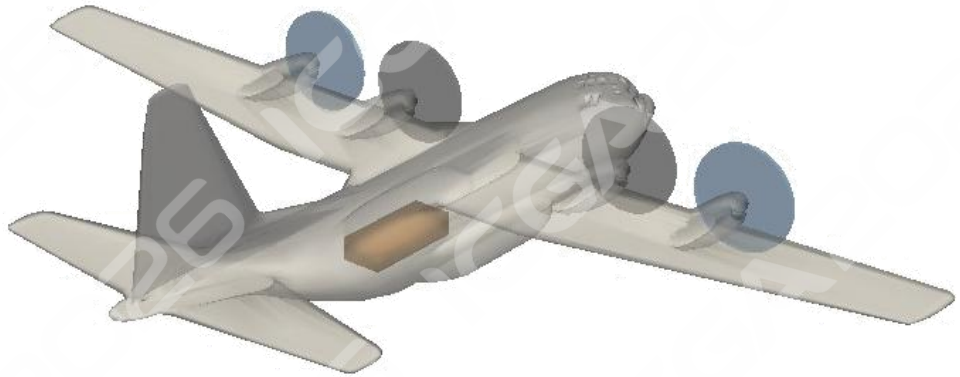


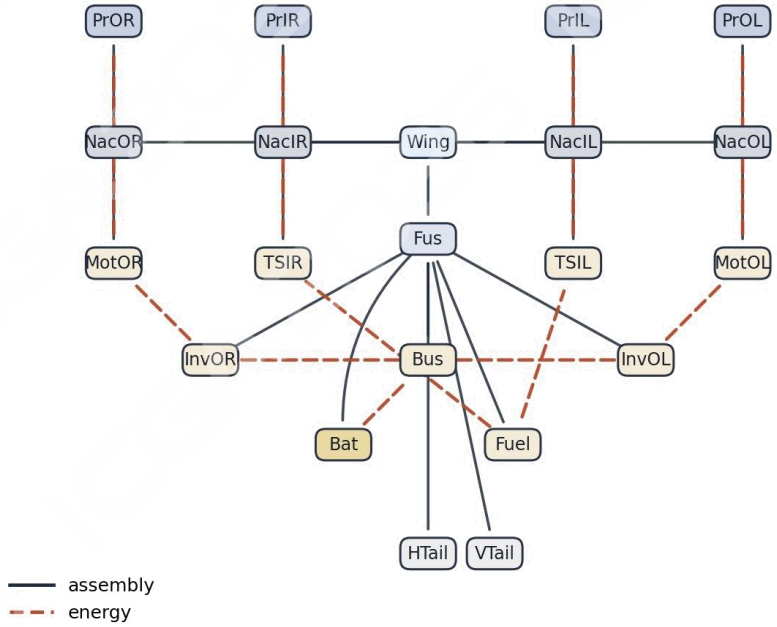


△ C-130H Concept [1]



△ True Parallel Hybrid-Electric Architecture

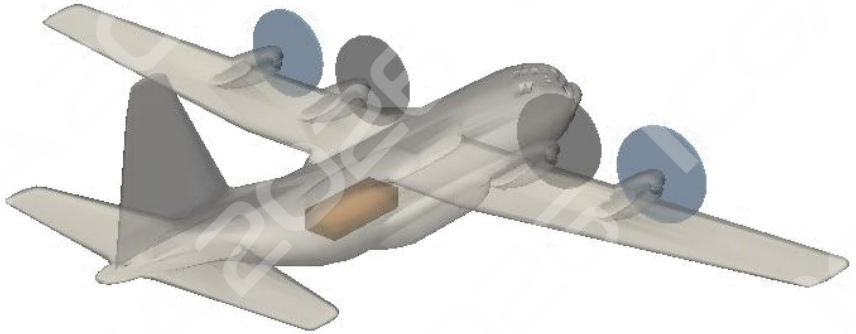


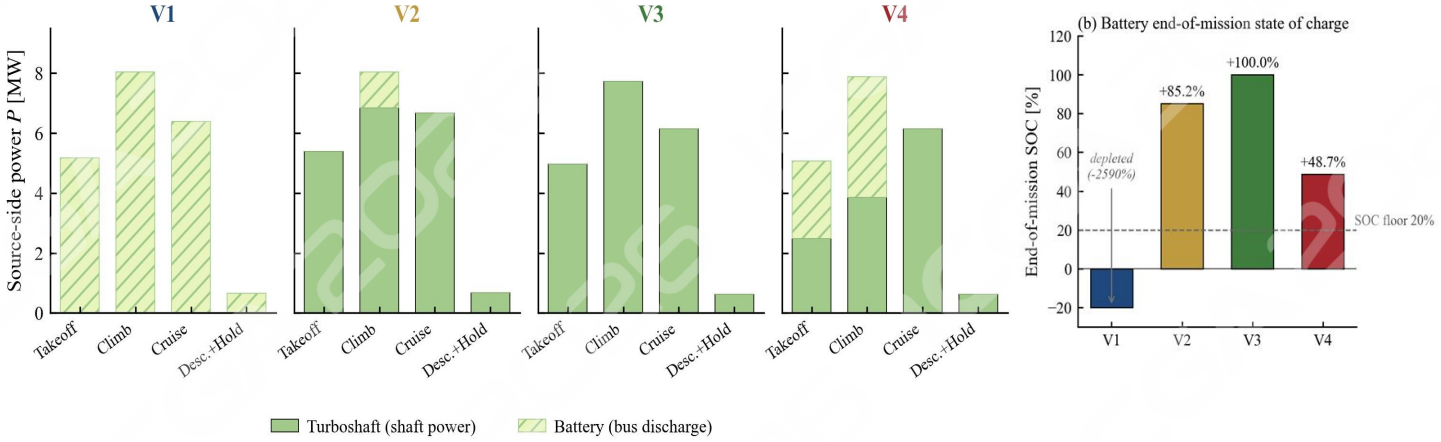


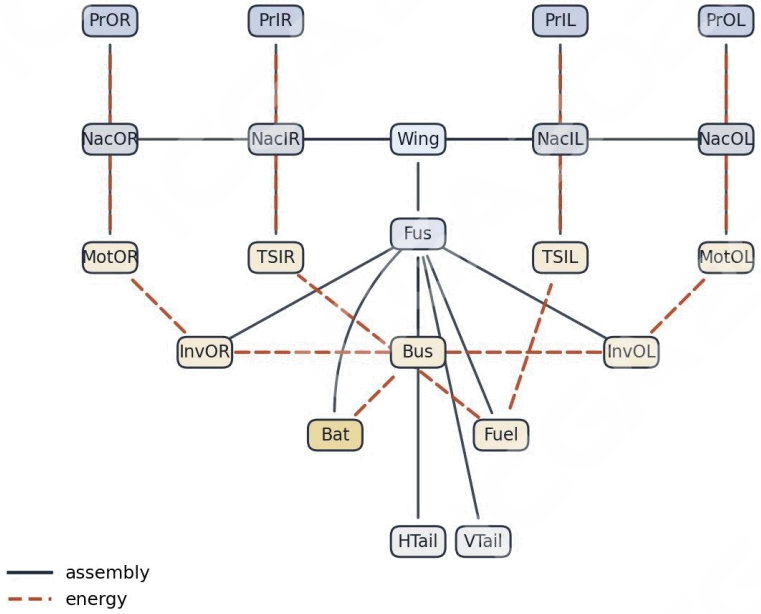
Application II

Architecture-Agnostic Mission Analysis of Cargo Aircraft

- Architecture-agnostic trim & mission analysis worked without any code-level rewriting.



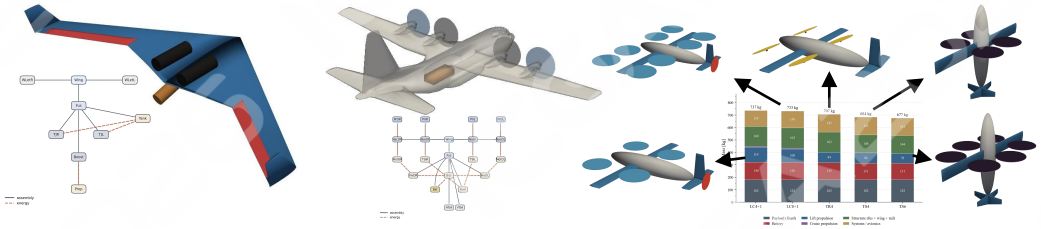




Transition	Graph changes	Framework code changes
Baseline → V1	−4 turboshafts, +4 motors, +1 battery, +1 bus, +9 edges	None
V1 → V2	+1 turboshaft, +1 generator, +2 edges	None
V1 → V3	+4 turboshafts, +4 gearboxes, +8 edges	None
V1 → V4	+2 turboshafts (inboard), −2 motors (inboard), rewired	None

SUMMARY: Methodological Requirements to Applications Mapping

- All requirements were satisfied and demonstrated.



Requirement	I (RATO)	II (TPH)	III (VTOL)
R1: Unified representation	•	•	•
R2: Model–viewpoint separation	•	•	•
R3: Topology-invariant workflow	•	•	•
R4: Explicit model–method decomposition	•	•	•
R5: Machine-readable state			•
R6: Declarative modification	•	•	•
R7: Structured diagnostics	•	•	•
R8: Human-oversight iteration			•