AI/Autonomy Solution Architecting Workflow Modeled in Digital Engineering

Ground System Architectures Workshop (GSAW)

October 2021

Approved for public release. OTR 2022-00081.

Overview

Digital engineering supports integrating trusted AI/autonomy from inception through demonstration

AI/Autonomy Solution Architecting is part of the larger architecting and systems engineering effort that integrates the necessary AI/autonomy from inception to reduce cost, schedule, and technical risk

- *Architecting, system engineering, and modeling and simulation (M&S) iterate and inform each other in a spiral process*
- *A solution team includes all relevant specialties serving iteratively throughout the process*
- *Algorithm complexity drives/can be driven by software and hardware requirements and thus architectures within trade space and constraints*
- *Conducting co-design within a digital engineering platform fosters rapid iteration and integration to develop and test prototypes*
- *Demonstrating autonomous systems progressively reduces risk*

Once an algorithm is selected, evidence must be collected to demonstrate trust

Digital engineering supports this entire process, from design to demonstrating trust

Image Credit: NASA OSIRIS-REx

Integrating AI/autonomy from Inception Requires Building in Trust

Digital engineering supports building in trust, which requires careful design, development, & testing

- **Trusted AI/autonomy is a nascent area**
  - *Researchers are working to develop best practices & standards, but no set of practices & standards has risen to widespread adoption yet*
- **Digital engineering** tools, such as Cameo or GENESYS, **support integrated co-design of trusted AI/autonomy**
- The Aerospace Corporation has developed a trusted AI framework that maps to & includes other researchers' trust frameworks
  - *Helps illustrate which information to collect for trust*
- **Digital engineering** can **instantiate** the Aerospace **trust framework**—or any other trust framework—to **expand** its tenets **into test routines & metrics**
- This presentation introduces the Aerospace AI/autonomy solution process and its digital engineering model
  - *The design use case is a **convolutional neural network** (CNN) used for **pose estimation** in on-orbit refueling within rendezvous proximity operations (RPO)*
  - ***Digital engineering supports concrete data, tests, & metrics***

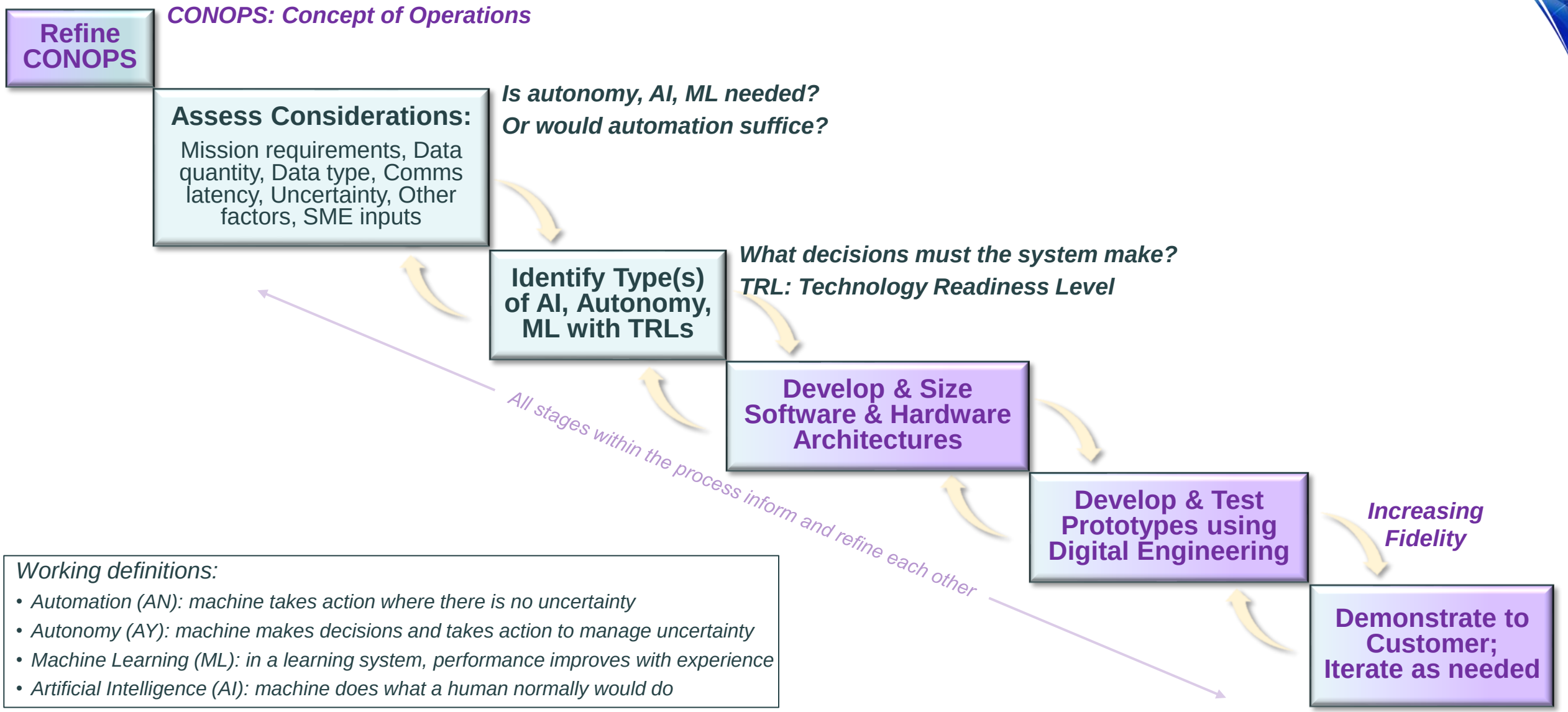
Budgeting design, development, & testing for trust up front saves time & money

Tailoring AI/autonomy solutions is key, digital engineering supports concrete measures for trust

Designing both the solution & its trust tests from inception reduces cost, technical, & schedule risks

The Aerospace AI/autonomy Solution Architecture Workflow

Digital engineering can model this process, which allows complexity requirements to drive design

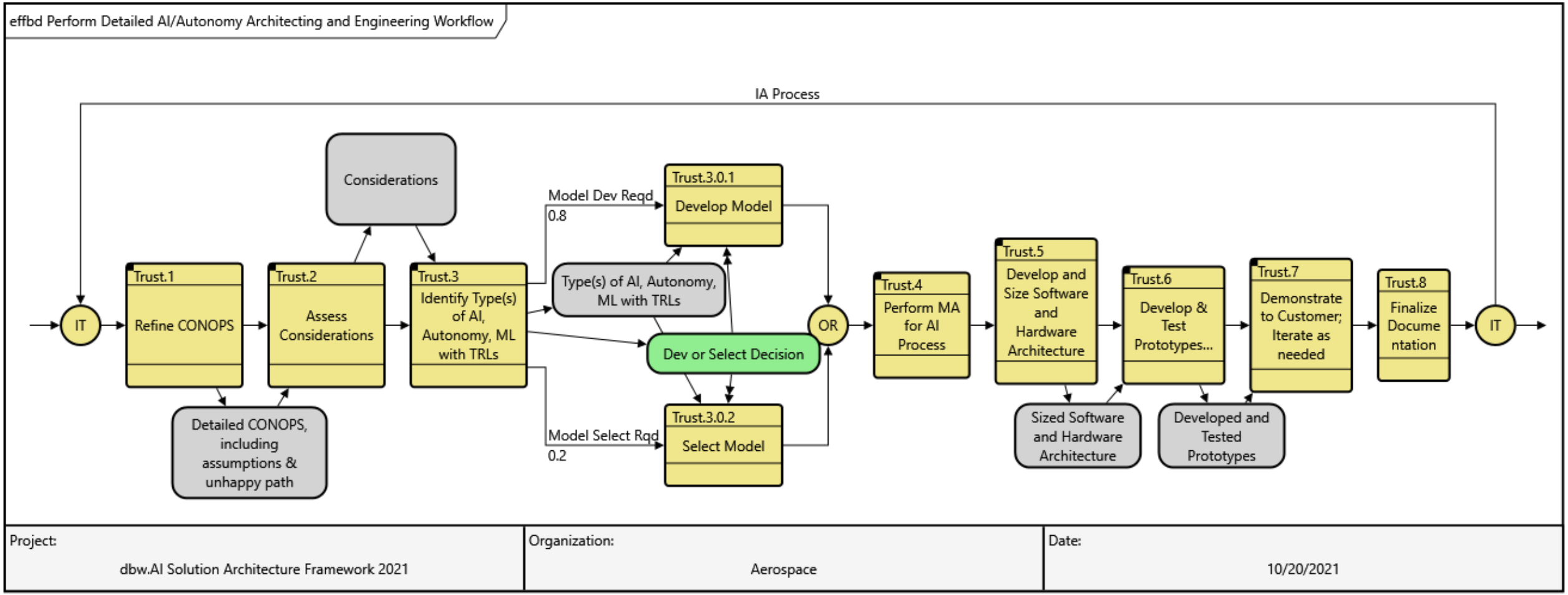


Working definitions:

- *Automation (AN): machine takes action where there is no uncertainty*
- *Autonomy (AY): machine makes decisions and takes action to manage uncertainty*
- *Machine Learning (ML): in a learning system, performance improves with experience*
- *Artificial Intelligence (AI): machine does what a human normally would do*

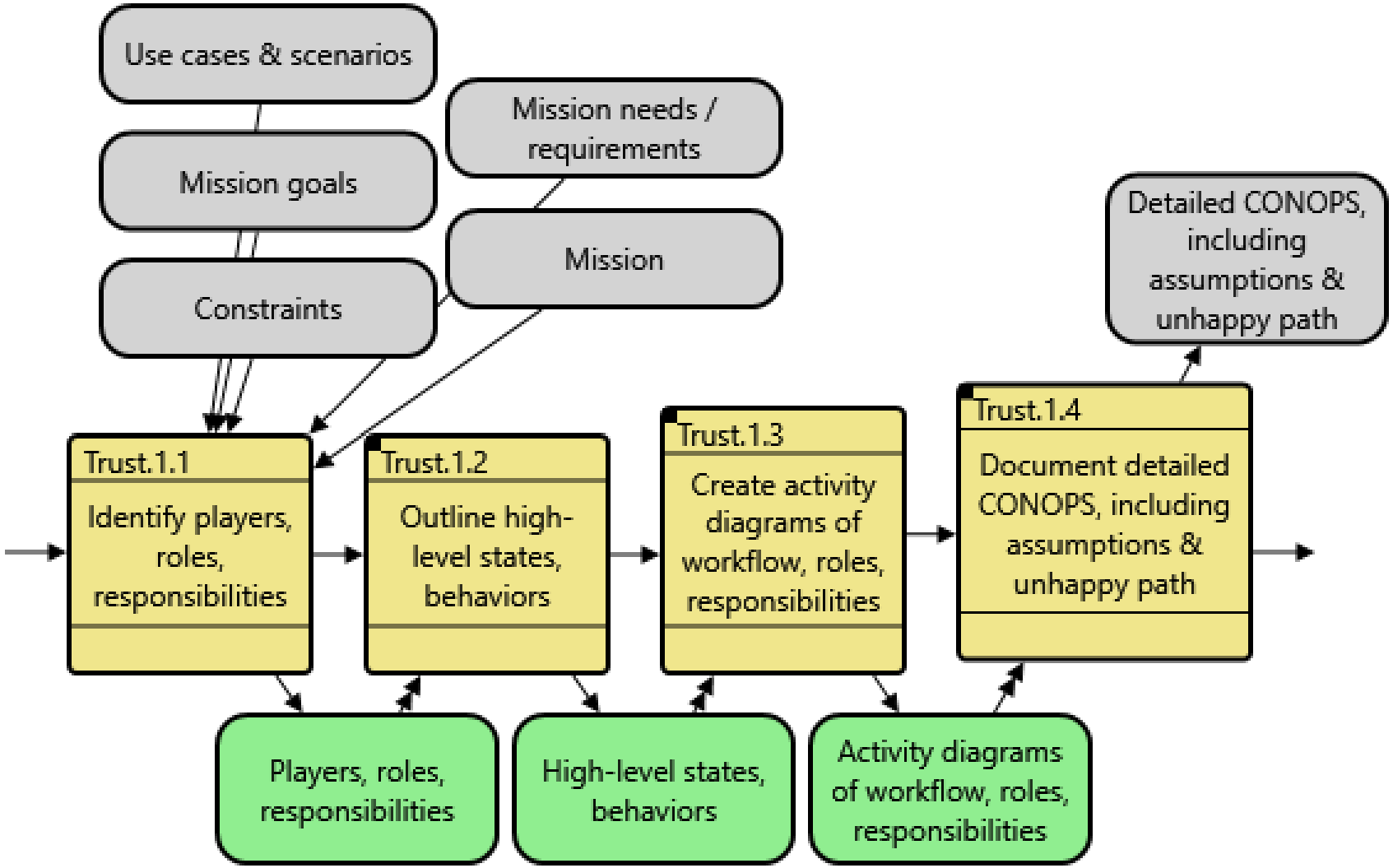
The Aerospace AI/autonomy Solution Architecture Workflow in GENESYS

The model instantiates the inner workings of the illustrated workflow concept

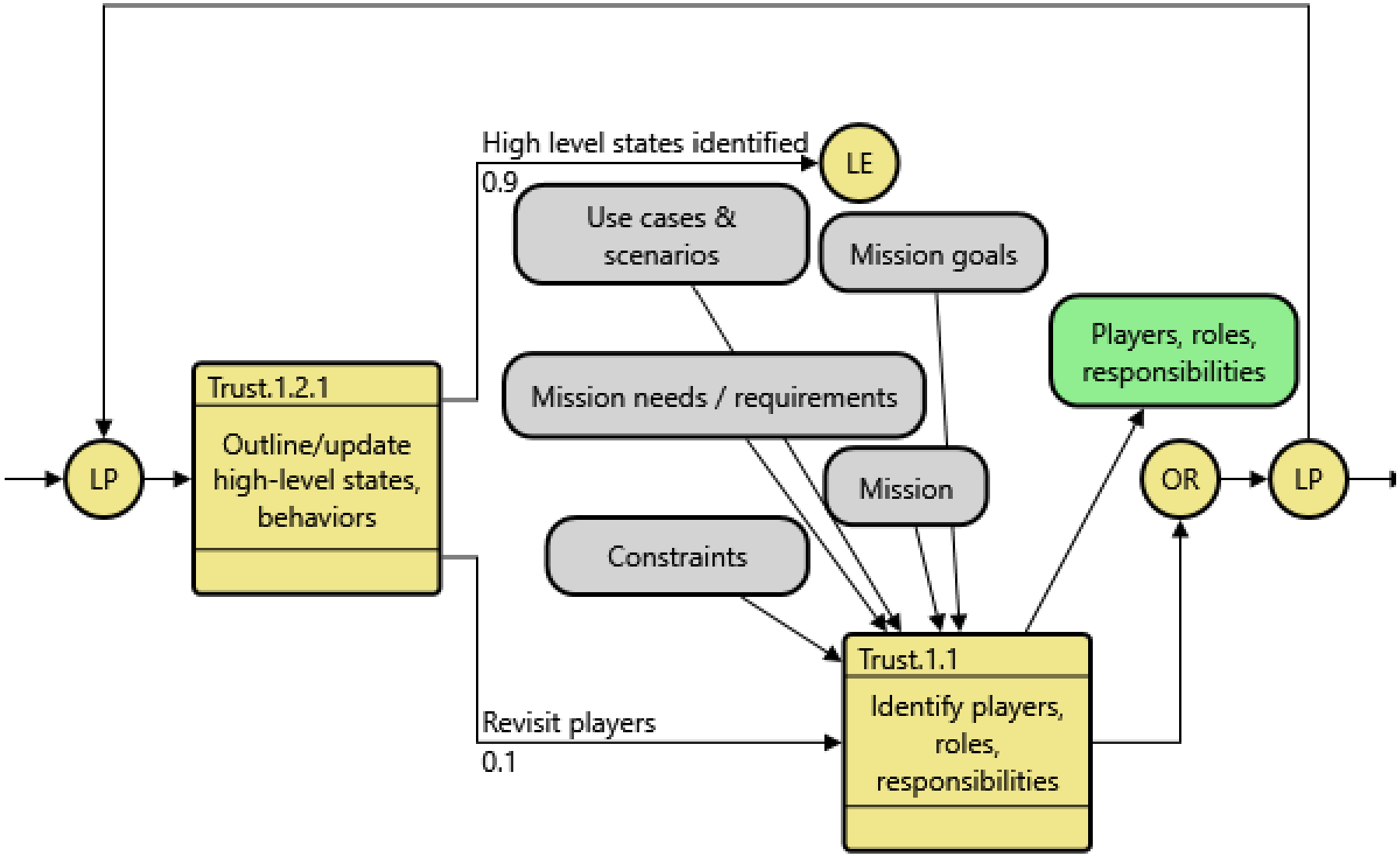


1: Refine CONOPS

The complexity required for the CONOPS and use cases drives the hardware/software design

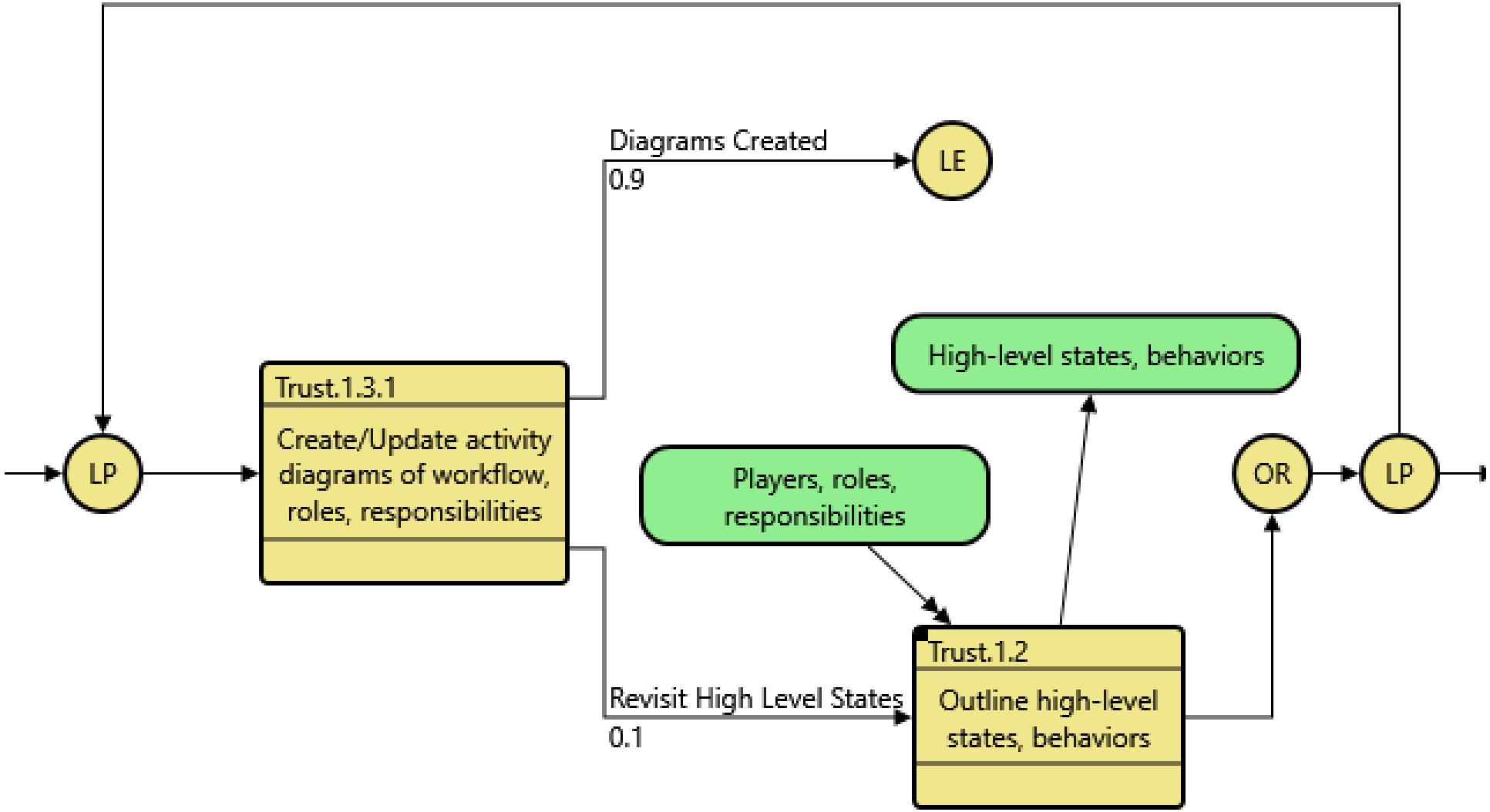


1.2: Outline High-level States, Behaviors

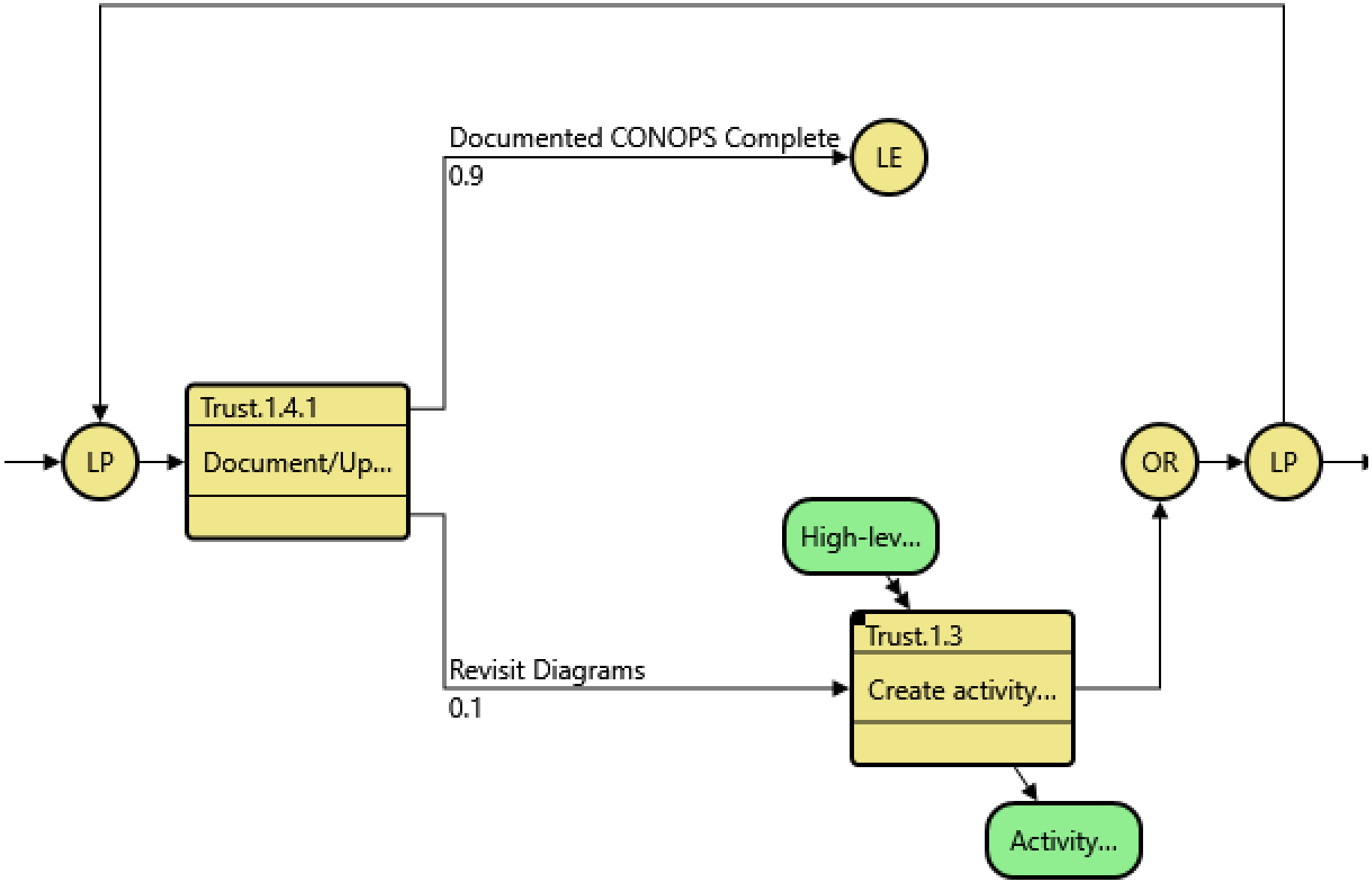


1.3: Create Activity Diagrams of Workflow, Roles, Responsibilities

Activity diagrams help illustrate the what, who, when, and why of AI/autonomy decisions & action

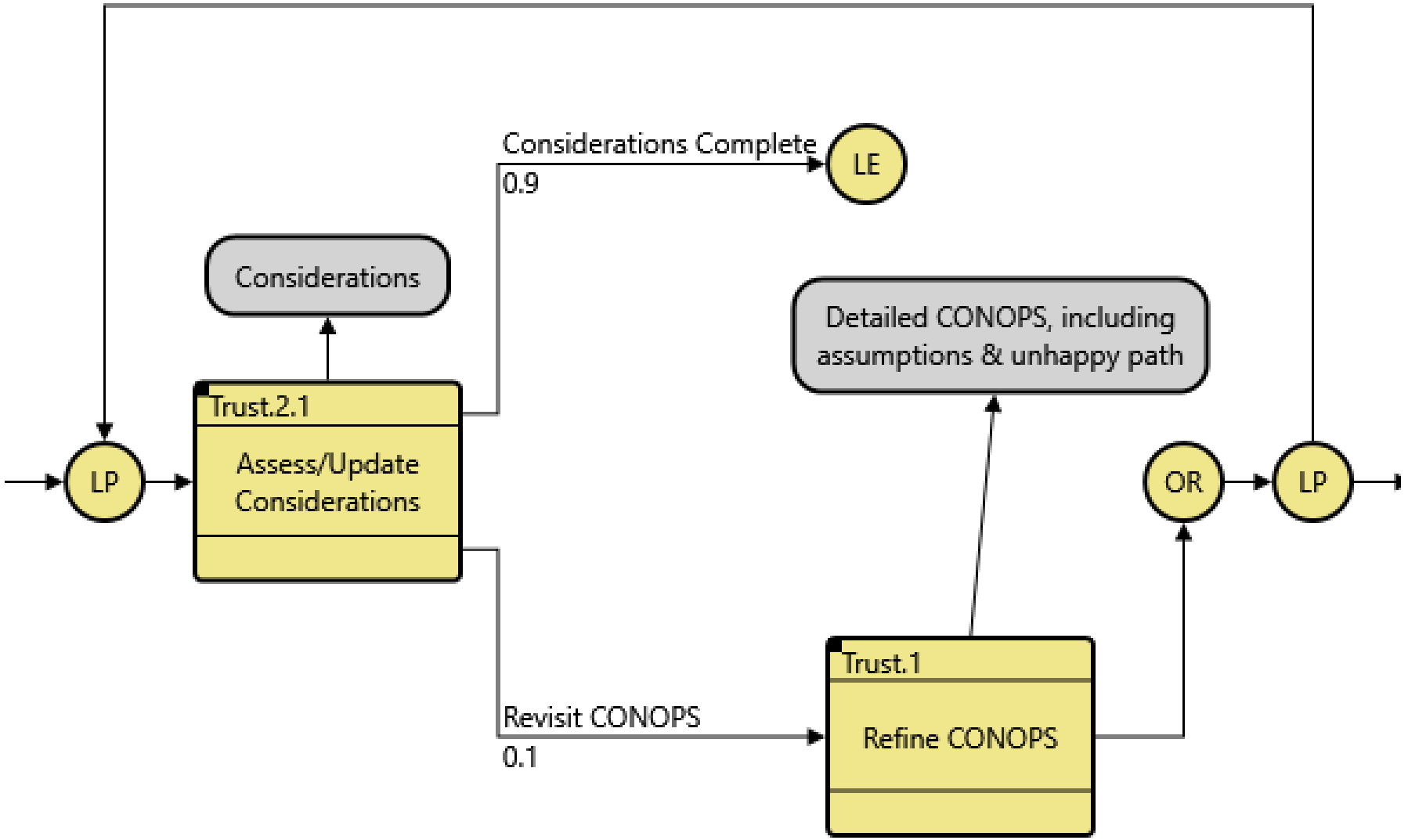


1.4: Document Detailed CONOPS, Including Assumptions & Unhappy Path



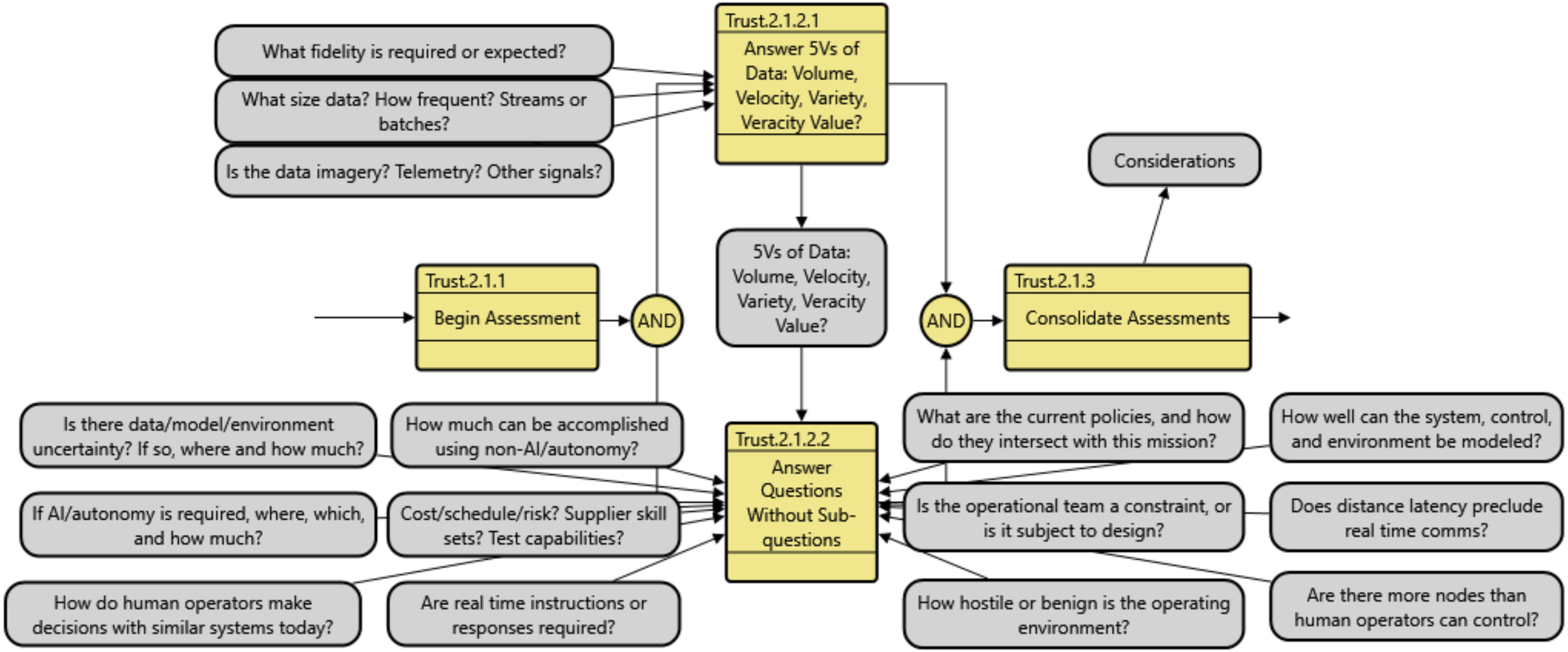
2: Assess Considerations

Which factors constrain the solution? What does the trade space look like?

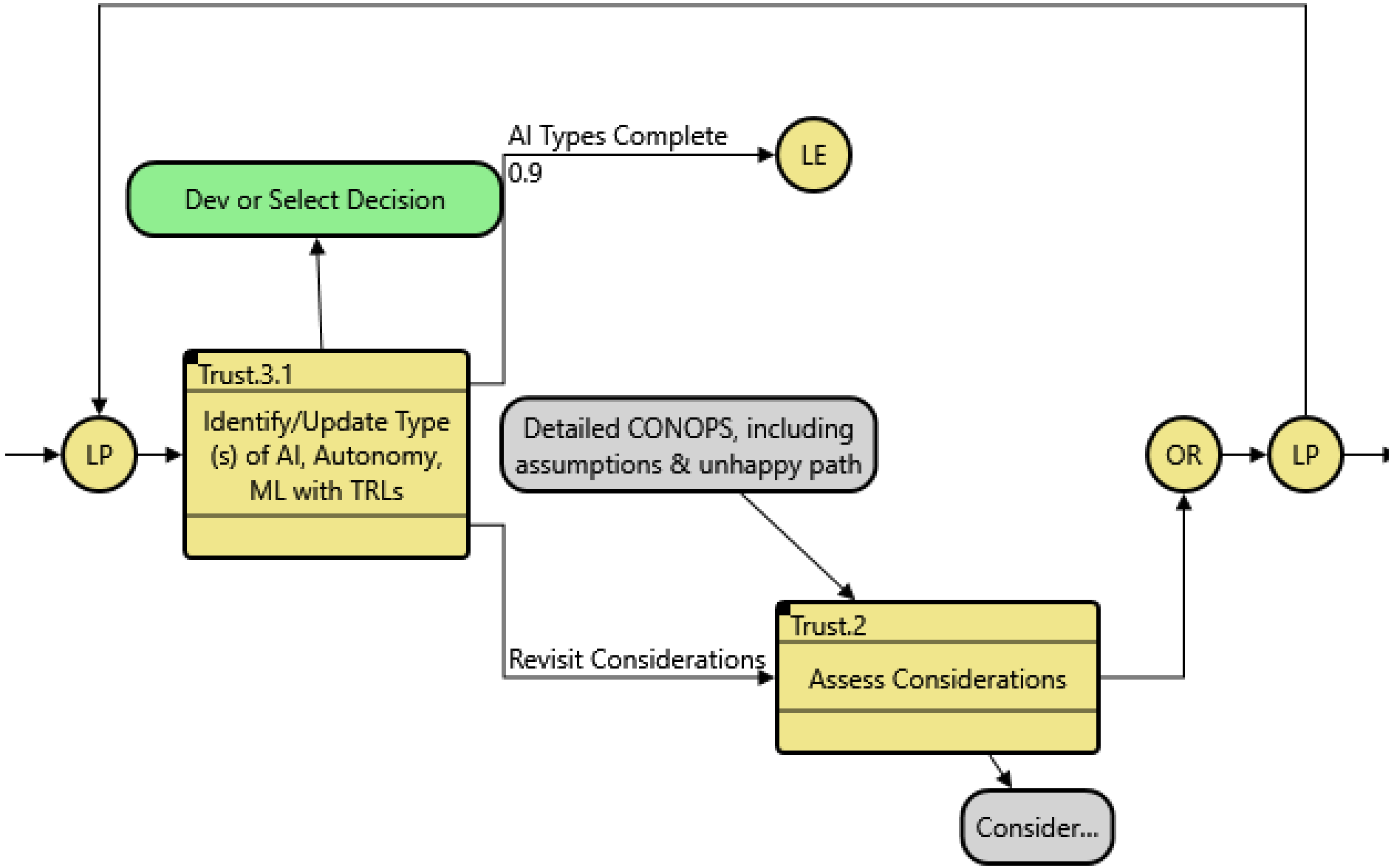


2.1: Assess Update Considerations

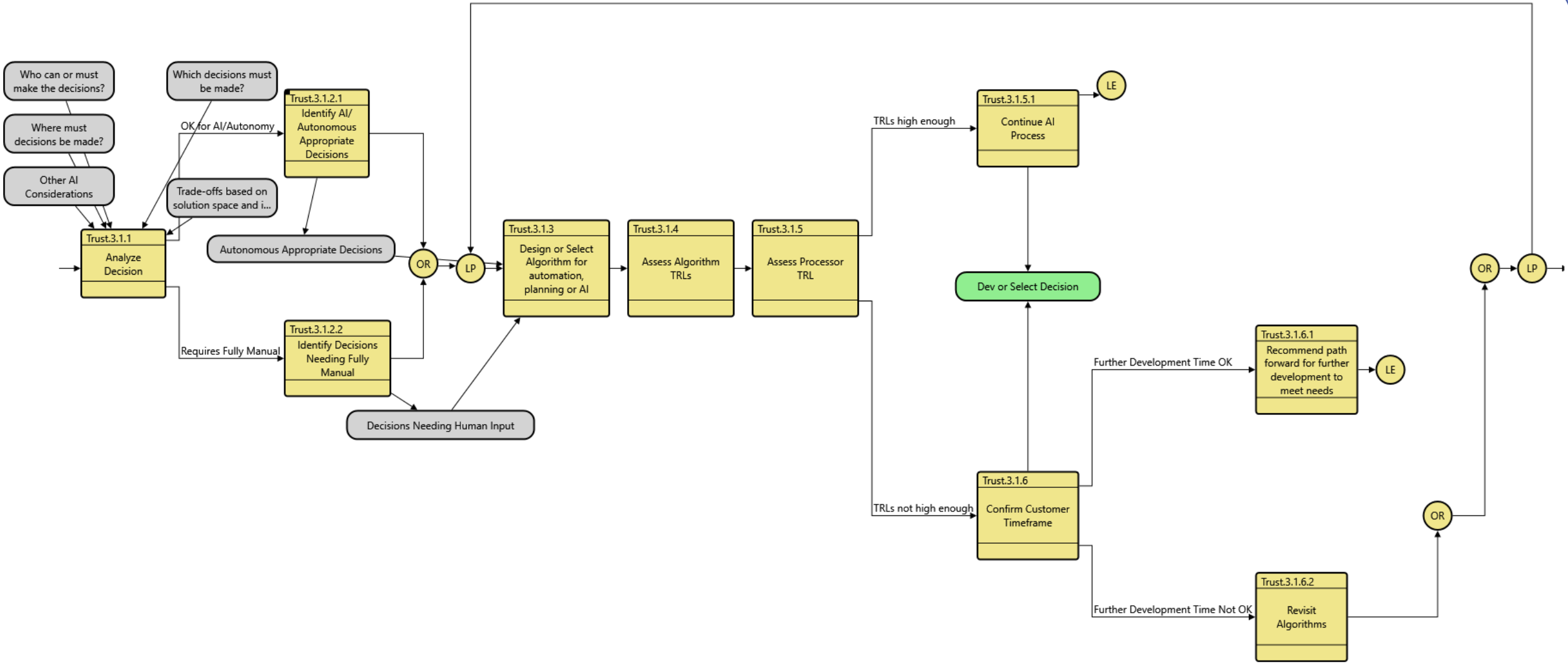
These considerations influence design re which AI/autonomy, why, & where within the solution



3: Identify Type(s) of AI, Autonomy, ML with TRLs

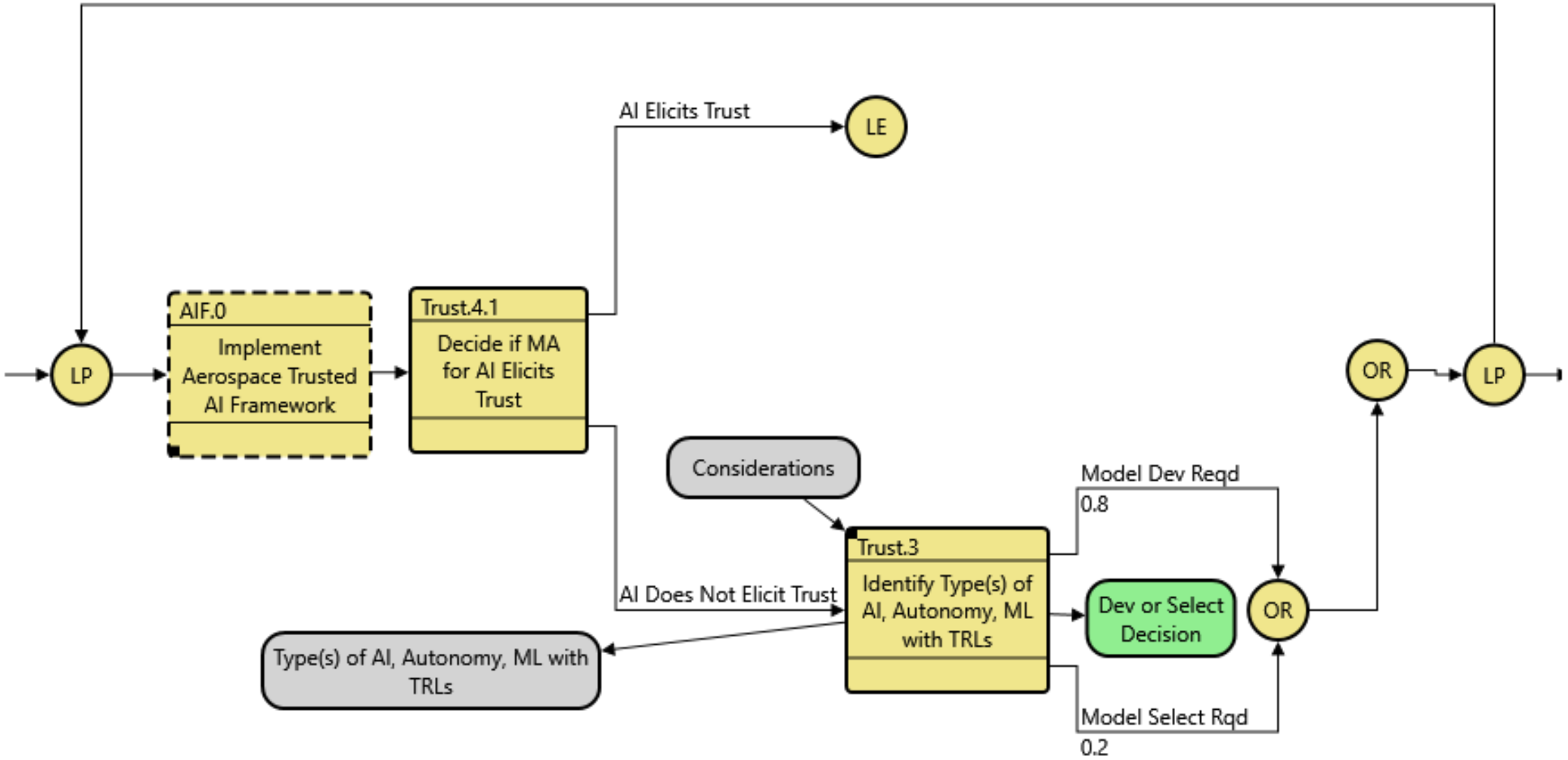


3.1: Identify Update Type(s) of AI, Autonomy, ML with TRLs



4: Perform Trusted AI ("MA for AI") Process

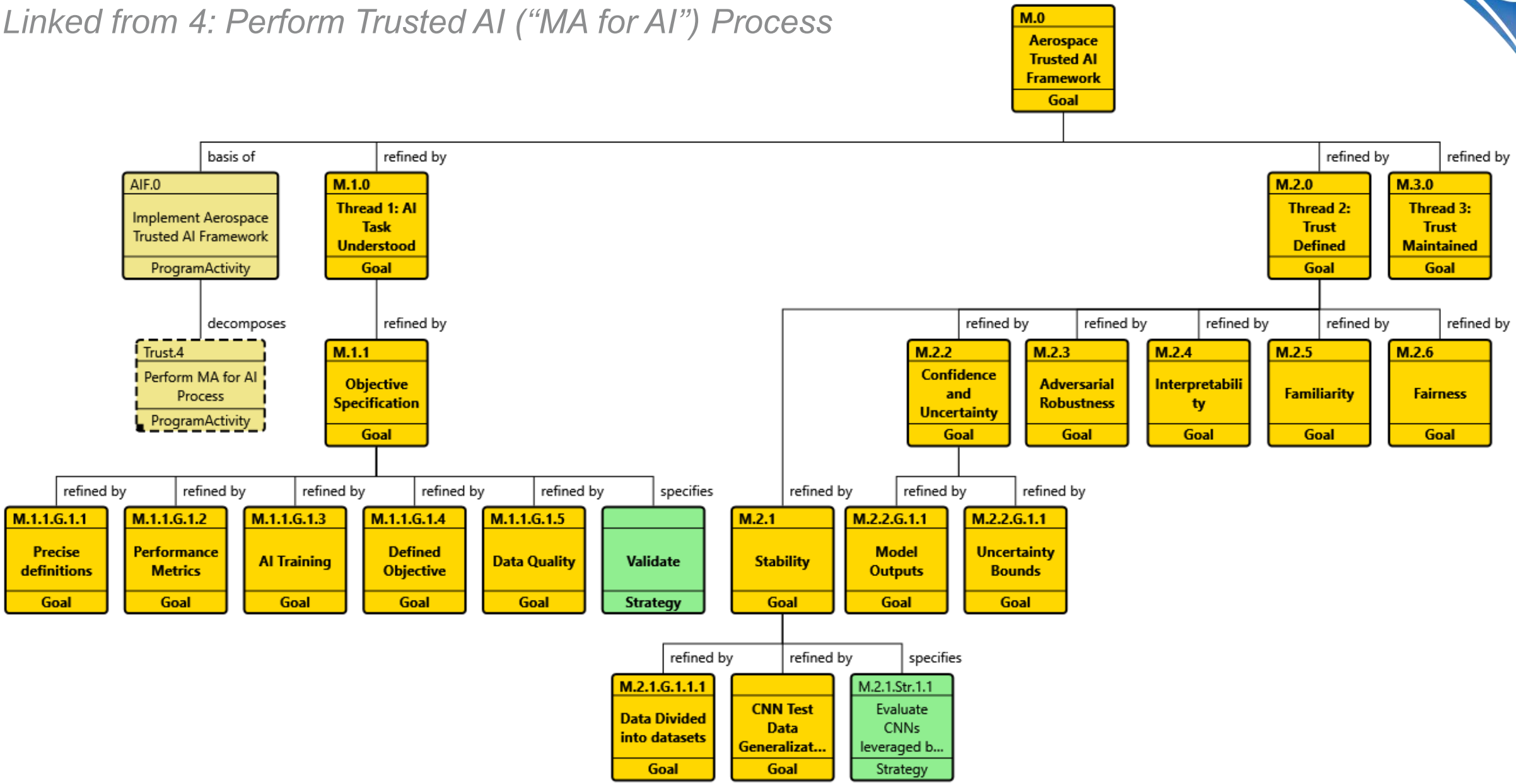
This stage of the solution architecting process links to the Aerospace Trusted AI Framework



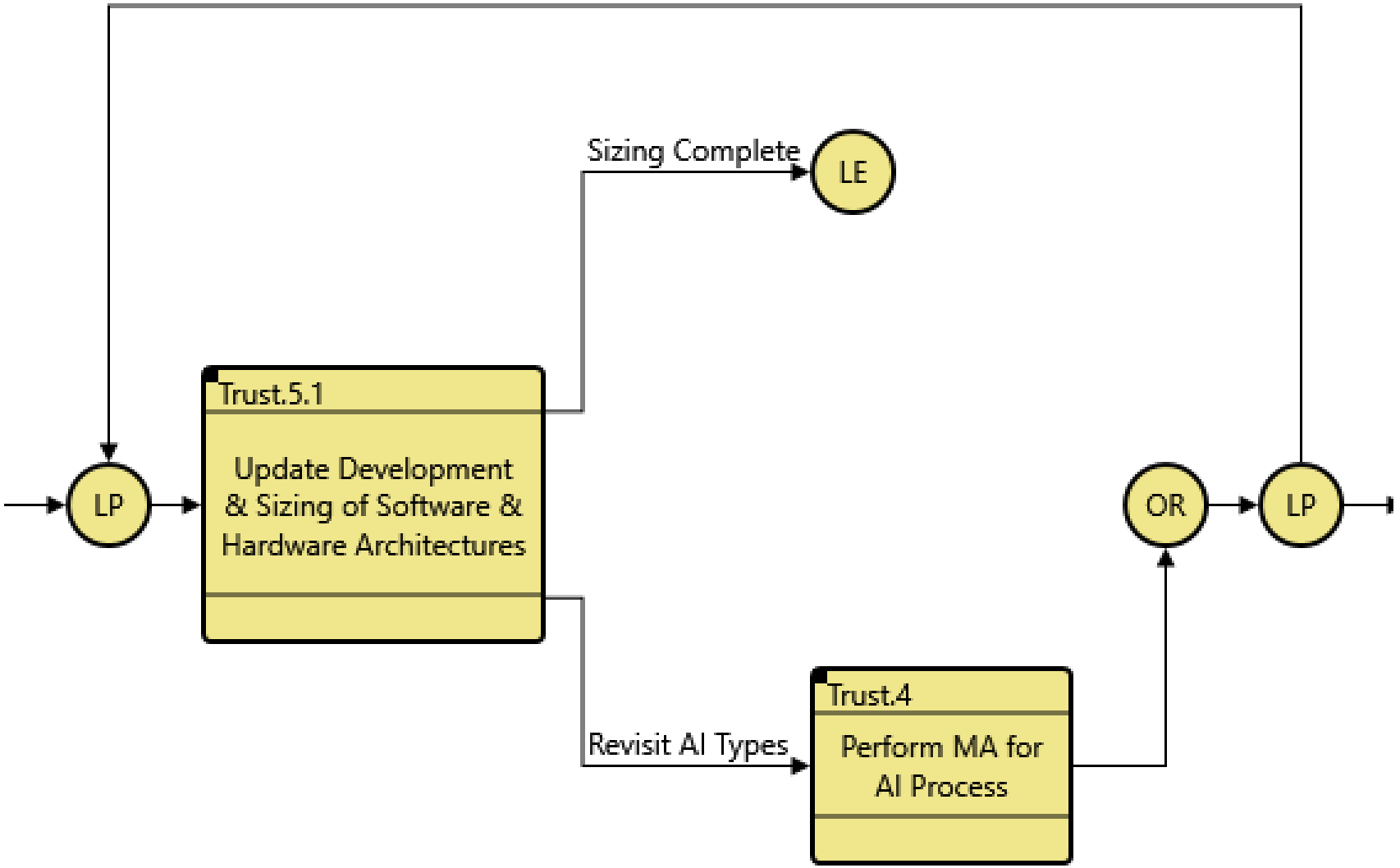
Stage AIF.0, the Aerospace Trusted AI Framework, is captured in a separate GENESYS model/project

The Aerospace Trusted AI Framework, Modeled in GENESYS

Linked from 4: Perform Trusted AI ("MA for AI") Process



5: Develop and Size Software and Hardware Architecture



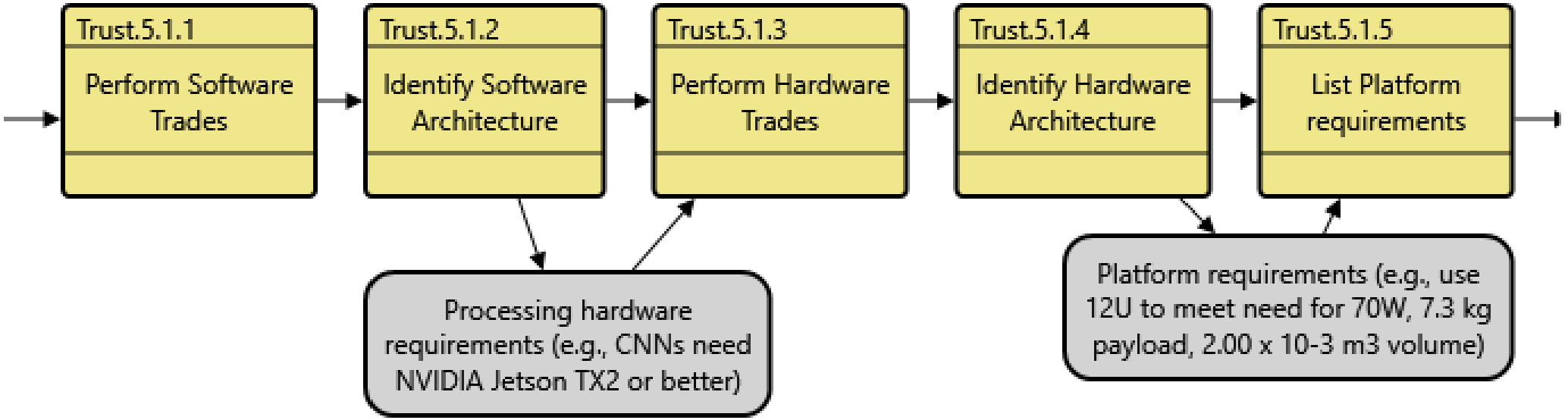
5.1: Update Development & Sizing of Software & Hardware Architectures

Trade studies inform this stage of the co-design process,

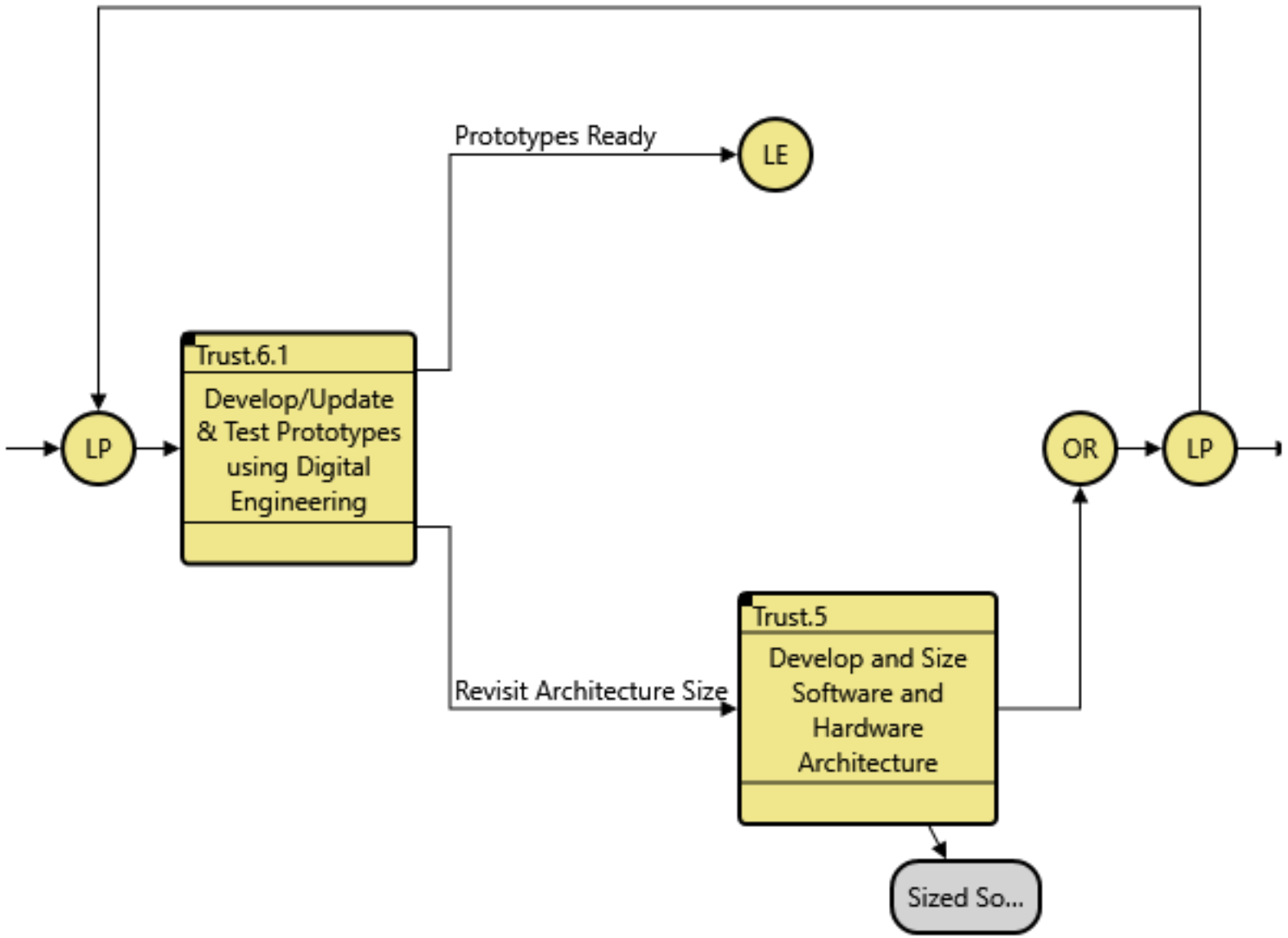
in which AI/autonomy software is part of the larger software architecture

and AI/autonomy hardware is part of the larger hardware architecture

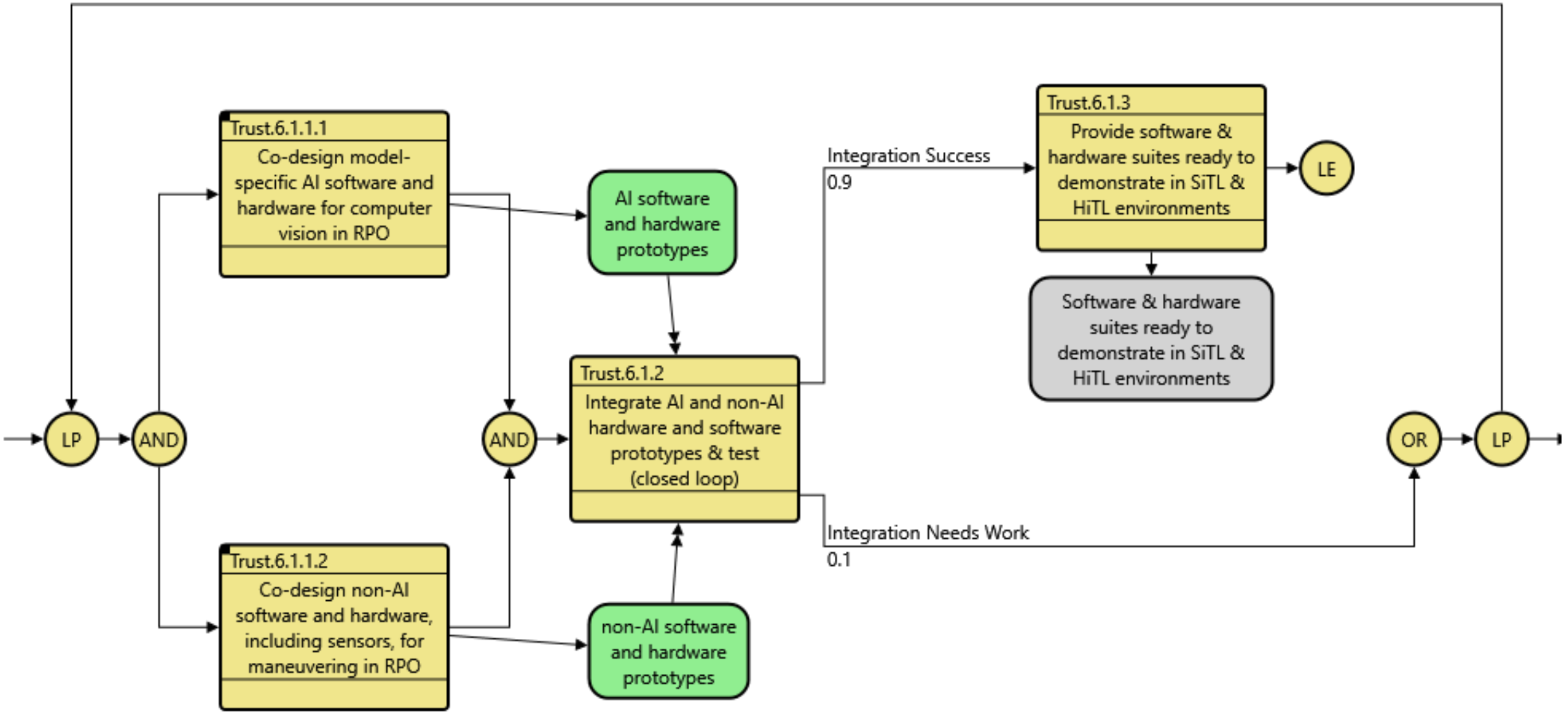
The overall hardware architecture must support the entire software-hardware co-design



6: Develop & Test Prototypes using Digital Engineering

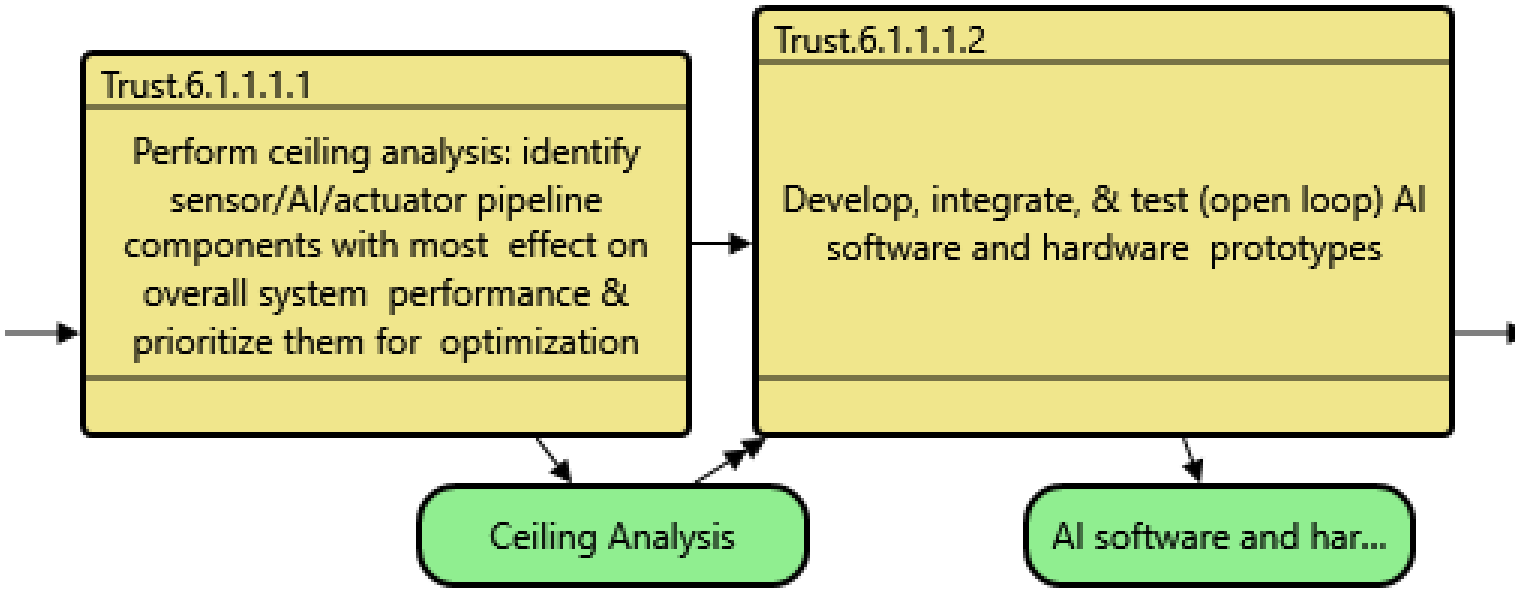


6.1: Develop Update & Test Prototypes using Digital Engineering



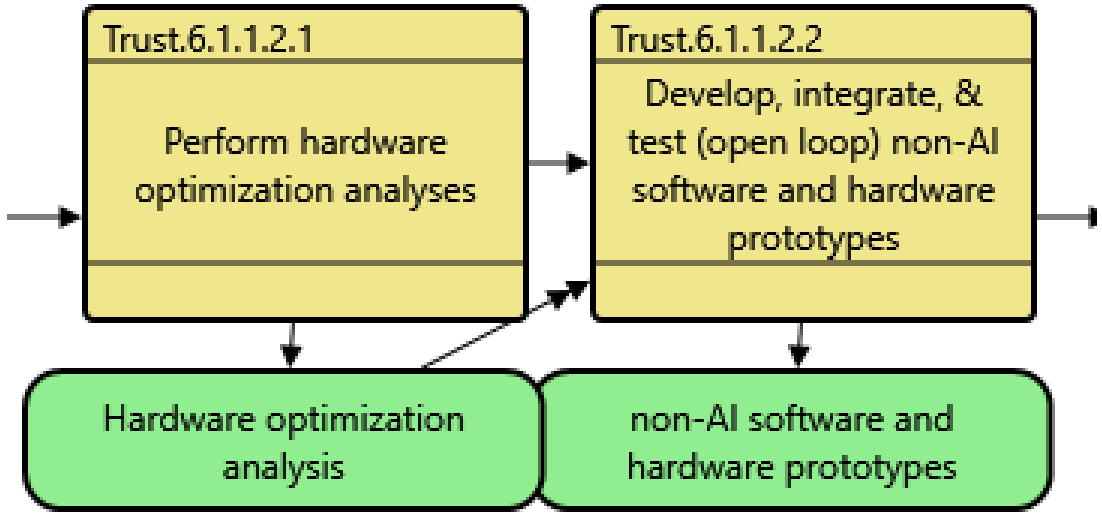
6.1.1.1: Co-design Model-specific AI Software and Hardware

for computer vision in RPO

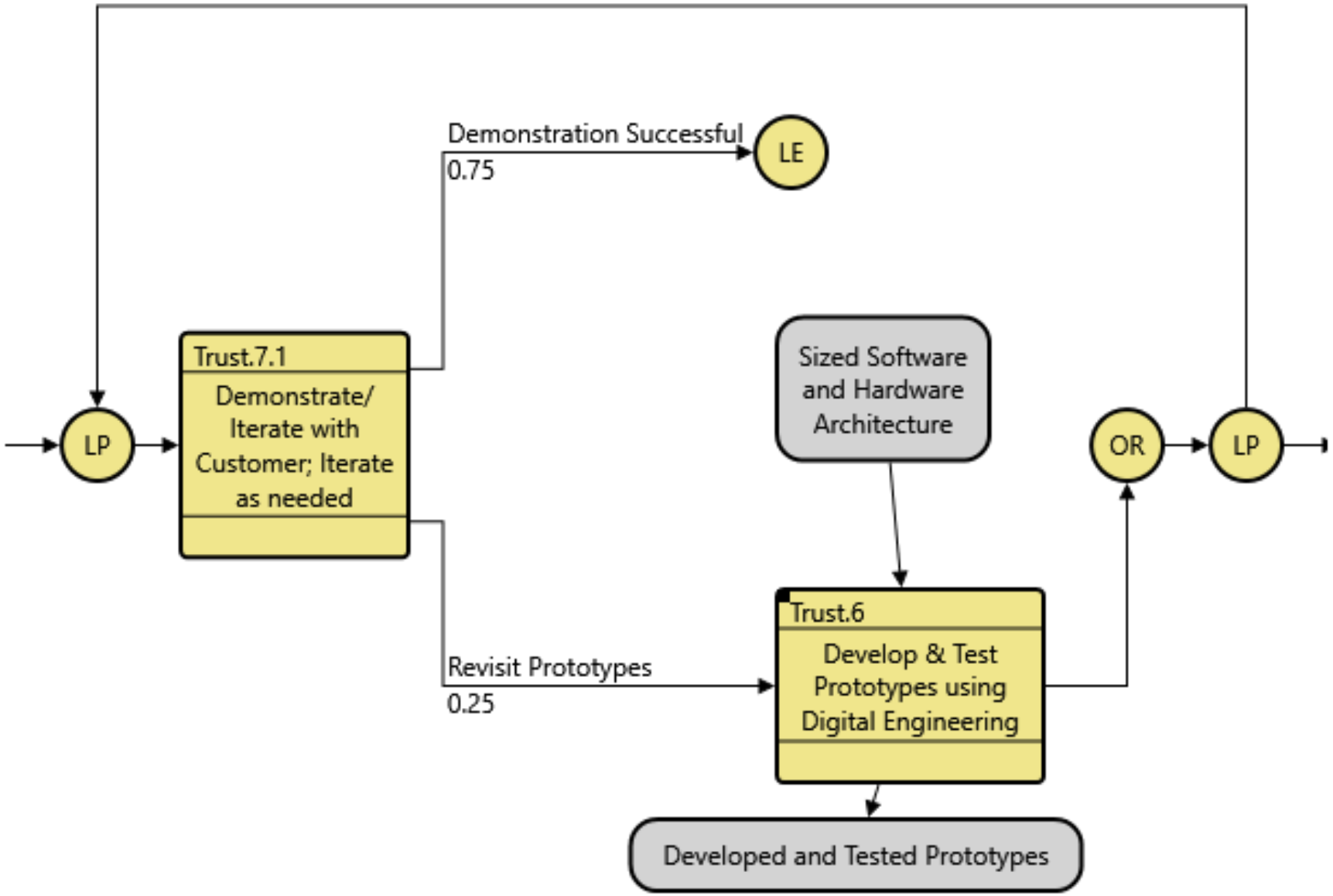


6.1.1.2: Co-design Non-AI Software and Hardware, Including Sensors

for maneuvering in RPO

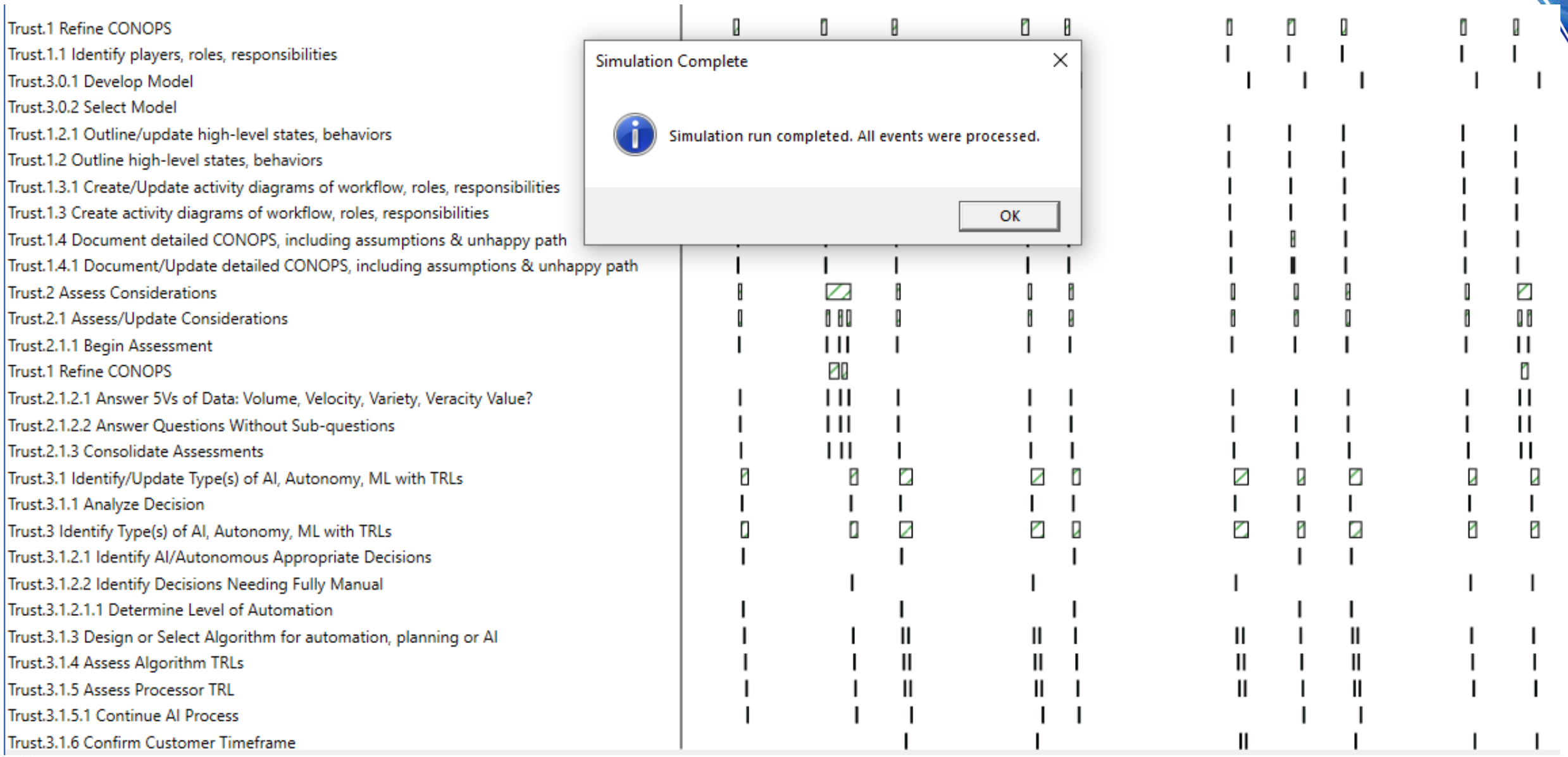


7: Demonstrate Prototype to Customer; Iterate as Needed



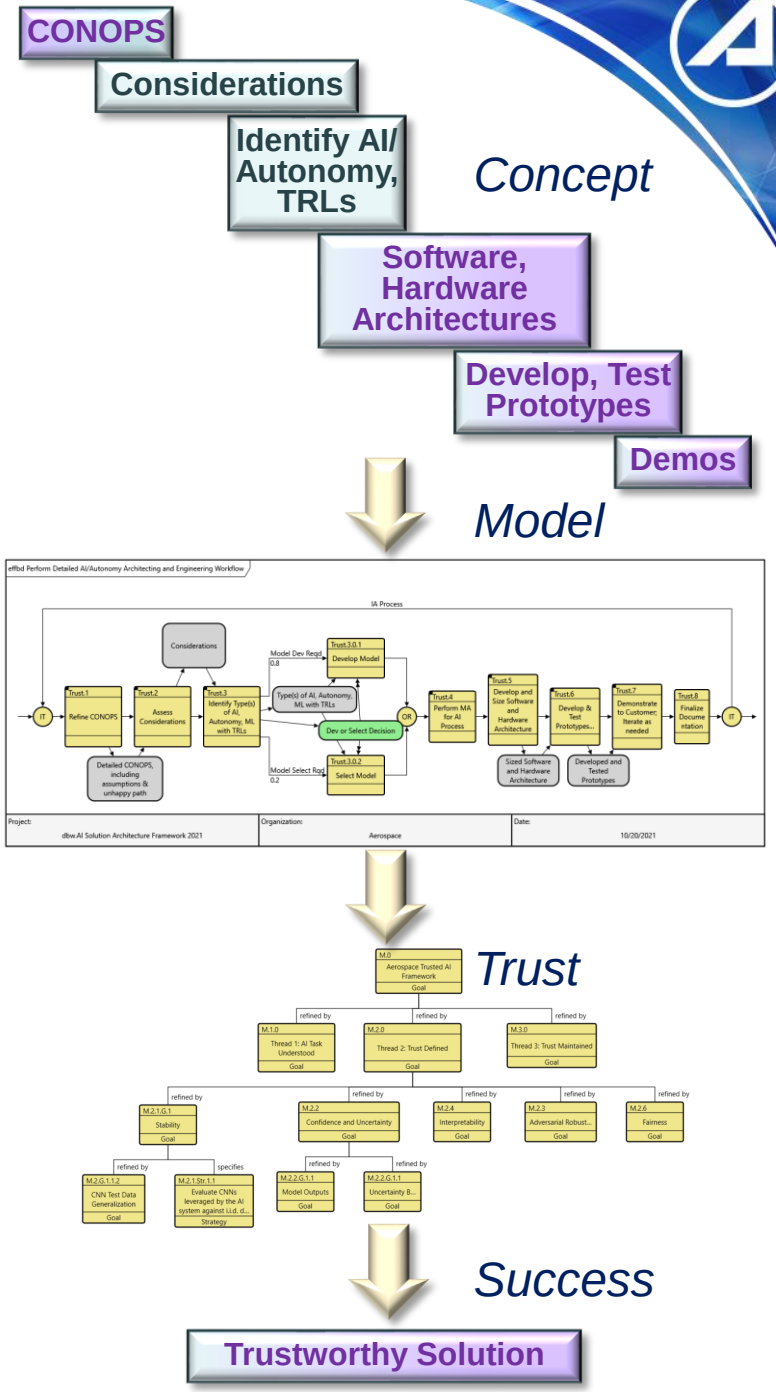
Simulation Based on the GENESYS Model

10 iterations



Summary

- Digital engineering supports the process of designing integrated AI/autonomy as well as demonstrating that such solutions are worthy of trust
  - *Digital engineering can be used to model The Aerospace Corporation's AI/autonomy Solution Architecting workflow*
    - This process uses the customer's complexity requirements to drive the hardware-software co-design
  - *The trusted AI part of the process uses The Aerospace Corporation's Trusted AI Framework*
    - This framework can be broken down into a Goal Structuring Notation, which can be instantiated in digital engineering for the design and demonstration of tests & metrics for trust
- The overall approach reduces cost, schedule, and technical risk by promoting a tailored AI/autonomy solution and helping customers and suppliers understand which information and evidence to collect to demonstrate that the resulting solution is trustworthy



Additional Information

- Introduction to The Aerospace Corporation's Trusted AI Framework
- How to begin collecting evidence for trusting a CNN to perform pose estimation

A Trusted AI Framework from Multiple Sources for Everyone

Explicit guidance on how to accomplish trust in relevant applications

The Aerospace Trusted AI Framework maps to and covers key industry frameworks as well as additional trust frameworks, including the IDA Roadmap to Assurance (May 2020) and the NIST Workshop on AI Trustworthiness (Aug 2020) (not shown in the table below)

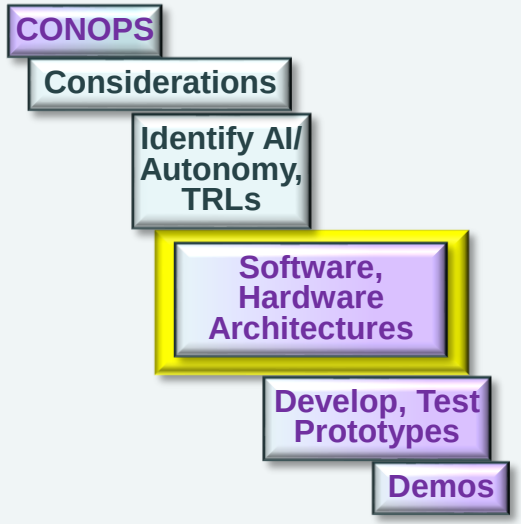
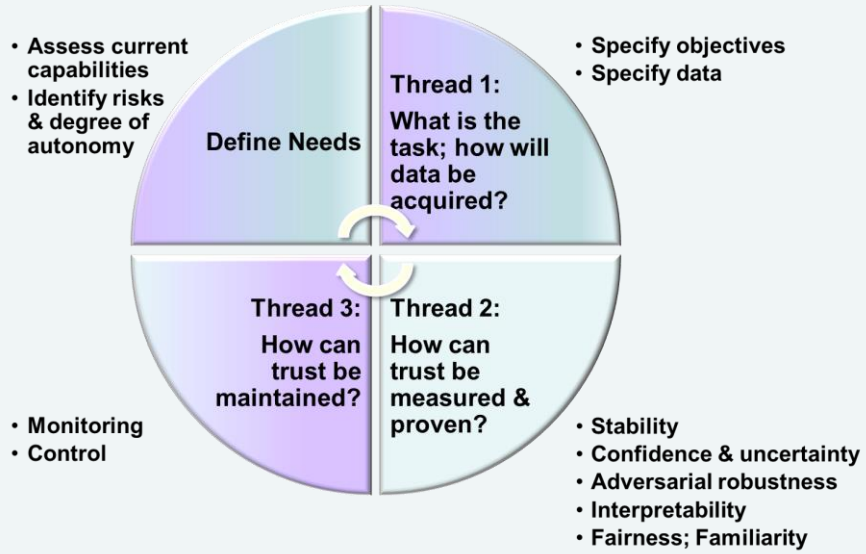
Aerospace's Trusted AI Framework		National AI Initiative Office Characteristics of Trust	DoD's Principles of AI Ethics		AI Ethics Framework for the Intelligence Community		Deloitte's Trustworthy AI Framework	IBM - Trusting AI		Microsoft Responsible and Trusted AI	U.S. Air Force Research Laboratory
Thread 1	Objective Specification	Ethical use of AI	Responsible, Traceable, Reliable		Testing, Version Control (builds, models, data), Stewardship	Governing the AI and the data; Documentation of Purpose, Parameters, Limitations, and Design Outcomes		Transparency and Accountability	Value Alignment		Ethics
	Data Specification	Privacy					Privacy			Privacy and Security	
Thread 2	Stability	Accuracy, Reliability					Robust/Reliable				Testing, Formal Methods
	Confidence and Uncertainty										
	Adversarial Robustness	Robustness, Resilience					Robust/Reliable		Robustness		
	Interpretability	Explainability and Interpretability, Transparency				Transparency: Explainability and Interpretability	Transparent/ Explainable		Explainability	Transparency	Transparency
	Familiarity										
	Fairness	Fairness, Bias Mitigation		Equitable		Mitigating Undesired Bias and Ensuring Objectivity	Fair/Impartial		Fairness	Fairness, Inclusiveness	
Thread 3	Monitoring	Security		Governable		Periodic Review	Safe/Secure	Transparency and Accountability		Reliability and Safety	Reliability and Security
	Control	Safety				Human Judgement and Accountability	Responsible/ Accountable			Accountability	

This framework facilitates development of concrete requirements, test routines, and performance metrics that provide evidence for trust

Digital Engineering Supports Designing a Trusted Software Architecture

AI-ML models, such as CNNs, require specific trade analyses and trust

Follow the AI Solution Architecture Process	Perform the model trade analyses	Select or design the model	Use the Trusted AI Framework to build trust in the model in from inception

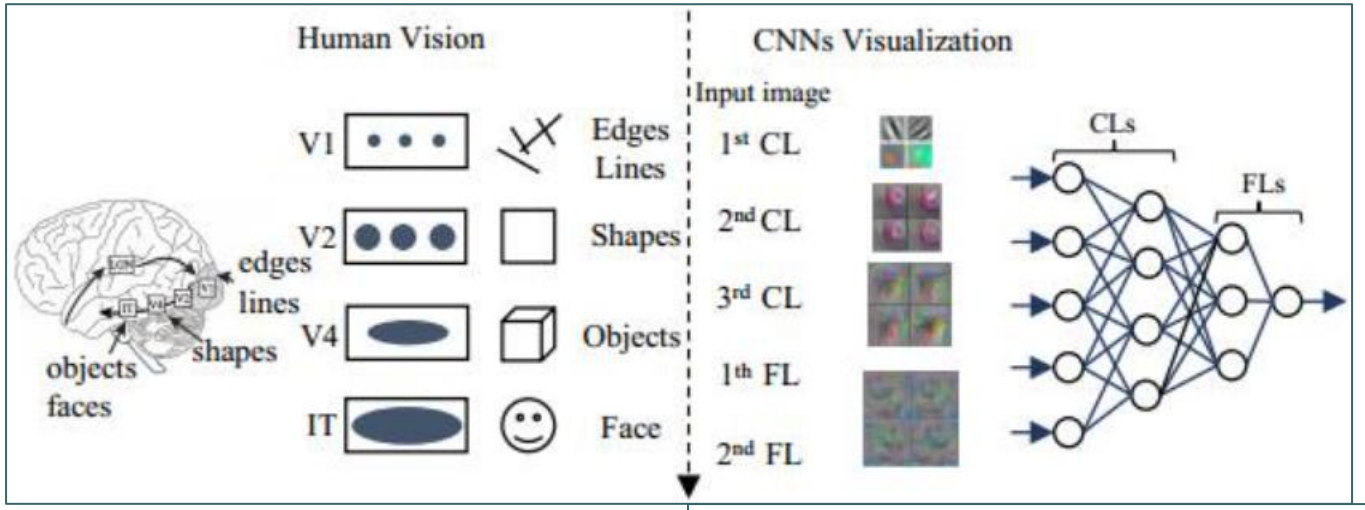

Software and hardware architectures require iterative co-design to build in tests, monitoring, and control for trust
	Model architecture requires optimization:
	• **Model size, Data sources**
	• **Languages, Libraries**
	• **Accuracy**
	  • More complex model
	  • Harder to train
	  • More data, compute resources
	• **Simplicity**
	  • Smaller model
	  • Fewer parameters
	  • Less data, compute resources
	  • Might reduce overfitting
	• **Transfer learning vs learning from scratch**
		Use AI Solution Architecture inputs, e.g., trades, to select/design a feasible, appropriately sized model:
		• CONOPS
		• Considerations
		• AI/Autonomy needs
		• Model size
		• Data needs and availability
		  • Data sources, quality, etc. (5Vs)
		  • Design of and 5V requirements for simulated data
		• Ceiling analysis
		• Hardware constraints
			
			Trusted AI is a nascent field requiring explicit definitions into meaningful, generalizable, measurable, and testable attributes. High consequence environments often entail high risk in mission criticality, algorithm complexity to meet mission criticality and complexity, and level of autonomy to meet issues like communications latency. data volume, etc.; technical, cost, and schedule risks must be quantified so they can be mitigated

The notional selection of a model, such as a CNN, for this use case is only the beginning of a solution

How do you know you can trust an artificial neural network like a CNN?

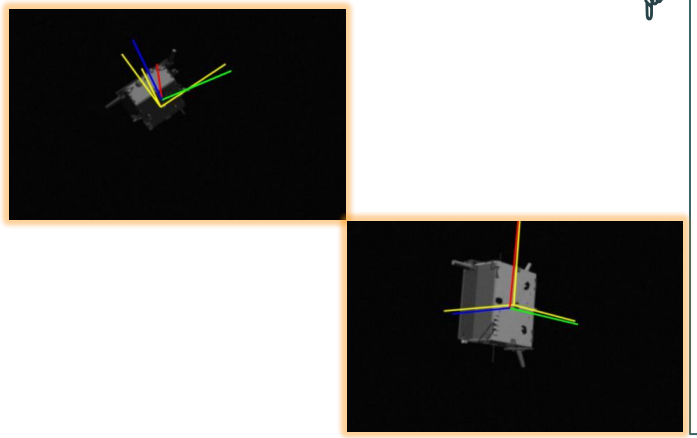
Trusting neural networks is not a solved problem—there is no established set of tests or metrics

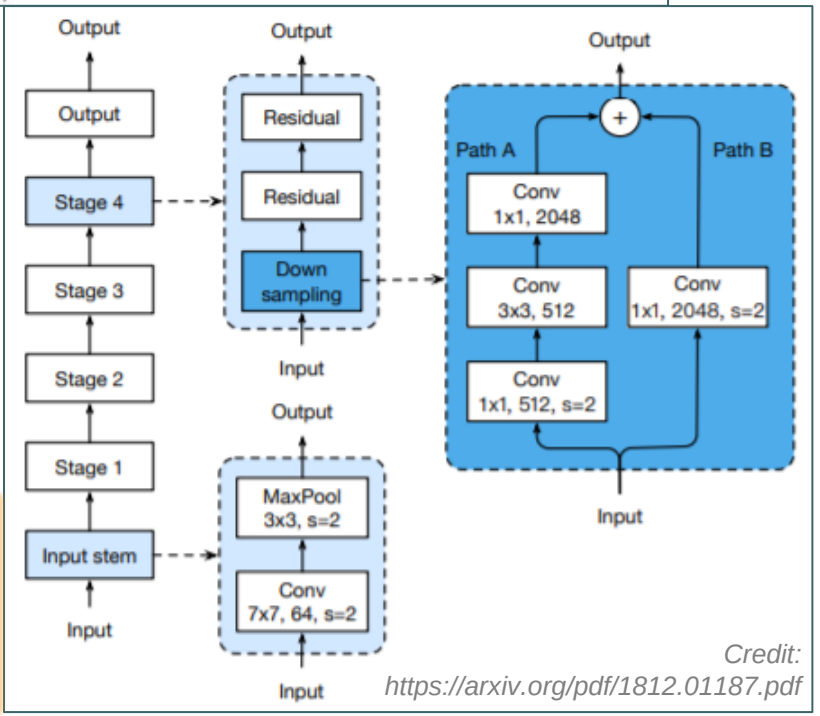
- Convolutional neural networks (CNNs) are inspired by the mammalian brain and vision system
  - *CNNs use multiple hidden layers & thus qualify as deep neural networks (DNNs)*
  - *DNNs are non-procedural and non-linear*
- CNNs are well established and accepted for applications like computer vision, including pose estimation
- CNN architecture is feed forward, but testing for trust is not intuitively obvious
  - *Trade analyses are required to validate the best model for an application*
  - *Metrics & tests must be developed*
- The Aerospace trusted AI framework is a helpful starting point for developing these metrics & tests



Credit: https://www.aimsciences.org/article/doi/10.3934/mfc.2018008

ResNet-50: ☞ a CNN Architecture for pose estimation ☜





Credit: https://arxiv.org/pdf/1812.01187.pdf

Trusting the Model via the Trusted AI Framework in Digital Engineering

*Goal Structuring Notations help define what's needed for **actionable confidence** that a model meets our objectives **quantifiably and understandably over the system lifetime***

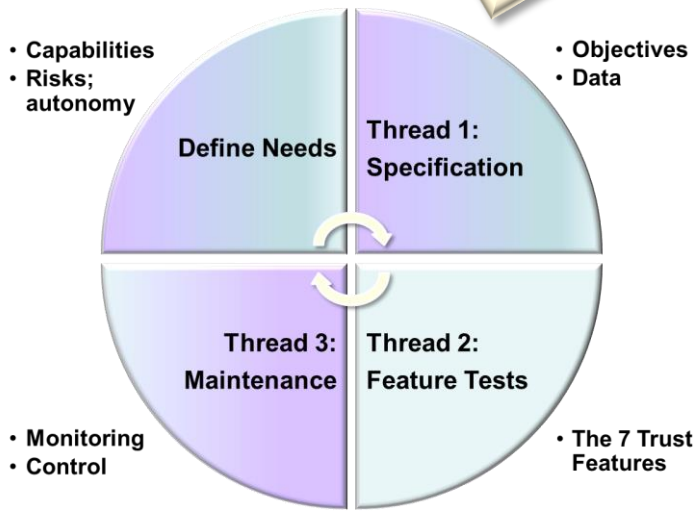
The Solution Team can use a Goal Structuring Notation (GSN) visual representation to decompose the threads of the trusted AI framework into sub-threads

M0/G1. *Aerospace Trusted AI Framework*:
The AI system and its characteristics meet user defined objectives in a proper and understandable way over the lifetime of the system.

M1
***Thread 1*:**
The AI system's task objective is understood and sufficient data has been acquired to accomplish the task.

M2
***Thread 2*:**
Trust has been explicitly defined for the AI system and can be measured and proven.

M3
***Thread 3*:**
The AI system maintains attributes of trust while deployed.



M1/G1. *Thread 1*:
The AI system's task objective is understood and sufficient data has been acquired to accomplish the task.

M1.1
***Objective Specification*:**
The task objective is accomplished in a manner consistent with user need and expectations.

M1.2
***Data Specification*:**
Assumptions made during selection of training data are understood to ensure accurate representation of deployed environment data.

M2/G1. *Thread 2*:
Trust has been explicitly defined for the AI system and can be measured/proven.

M2.1
***Stability*:**
The AI system's predictions are consistent when provided inputs fall within routine parameter ranges.

M2.2
***Confidence and Uncertainty*:**
Levels of confidence and uncertainty bounds can be discerned for the AI system.

M2.3
***Adversarial Robustness*:**
The AI system is not only robust to a wide variety of known inputs but also robust to purposefully misleading inputs.

M2.4
***Interpretability*:**
The AI system is instrumented in a way for users to easily understand the underlying causes of how responses were formulated.

M2.5
***Familiarity*:**
Users are able to understand when to trust and when not to trust the AI system.

M2.6
***Fairness*:**
The AI system ensures that decisions made are not unfair or do not cause unintentional negative consequences due to bias.

M3/G1. *Thread 3*:
The AI system maintains attributes of trust while deployed.

M3.1
***Monitoring*:**
The AI system is instrumented so that data can be regularly and easily collected for assessment.

M3.2
***Control*:**
When unexpected behavior occurs, an automated means of user notification and/or system interruption is provided.

GSNs help systematically decompose the sub-threads into "branches" of evidence to collect—foundations for test routine "leaves"—to make the argument for trust

Any drawing package (e.g., Visio) can be used to develop the GSN

Test routines that provide evidence to help argue for trusting this CNN

Trusted AI Framework → GSN → Test Routines & Metrics → Models & Simulations using Digital Engineering tools

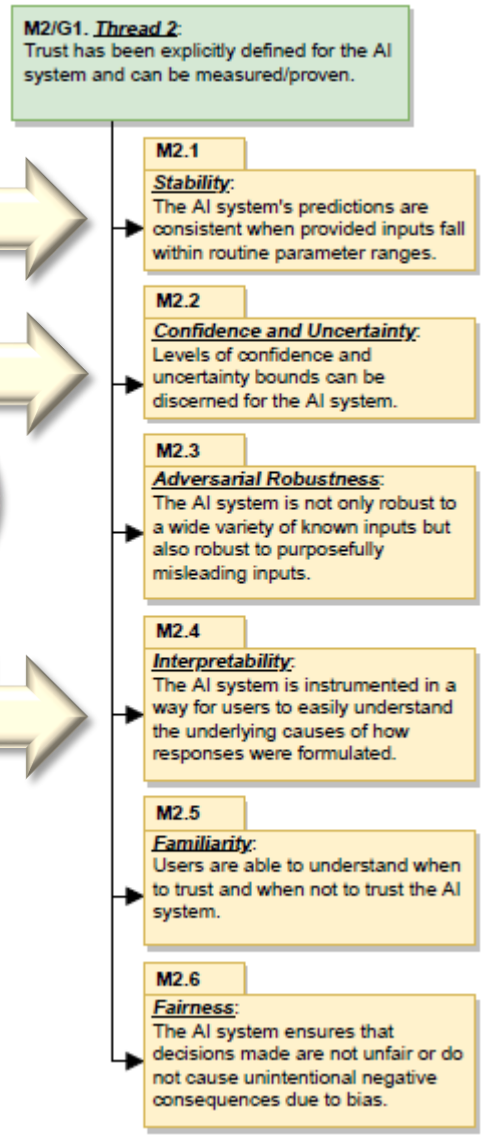
Threads of Trusted AI: Feature Tests

The example in this presentation focuses on a subset of trust features listed under Thread #2

For this demonstration, the GSN expounds on three features, or sub-threads

- **Stability**
- **Confidence and Uncertainty**
- **Interpretability**

for modeling and simulation in GENESYS, a digital engineering environment



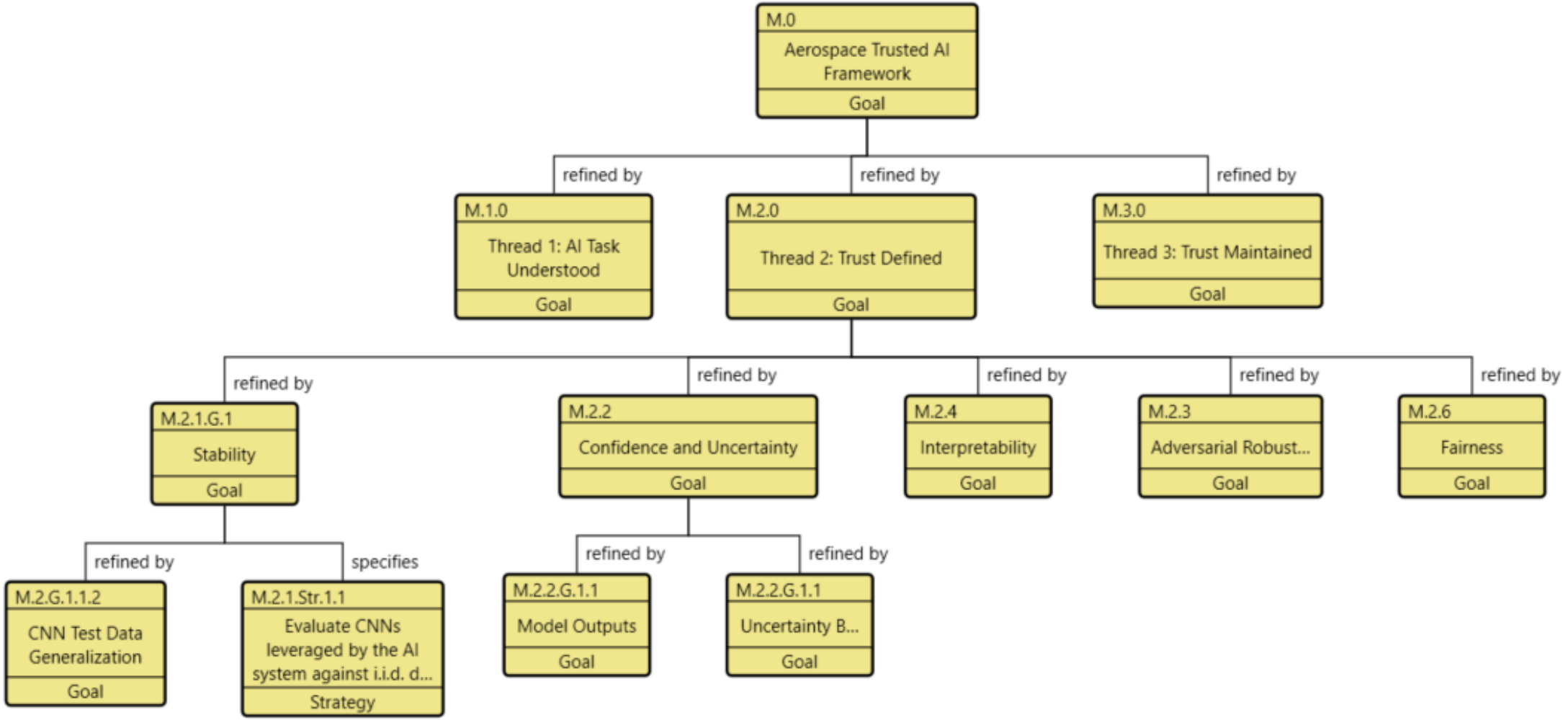
Stability: predictions are consistent when provided inputs fall within routine ranges

Confidence and Uncertainty: levels of confidence and uncertainty bounds can be discerned

Interpretability: users can easily understand the underlying causes of how responses were formulated

Integrating the Satellite Pose CNN into Digital Engineering

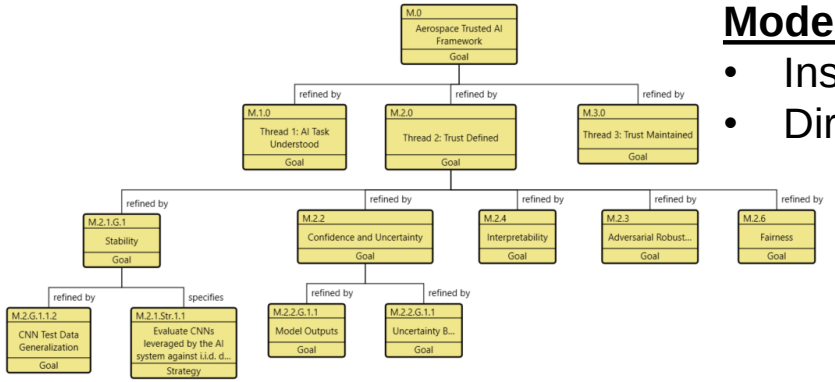
GSN elements of interest



This is the GENESYS instantiation of the GSN that was developed using a drawing tool like Visio

Digital Engineering: MBSE AI-ML Evaluation Flow

Elements of a GSN instantiated in GENESYS can access external code, like Python scripts

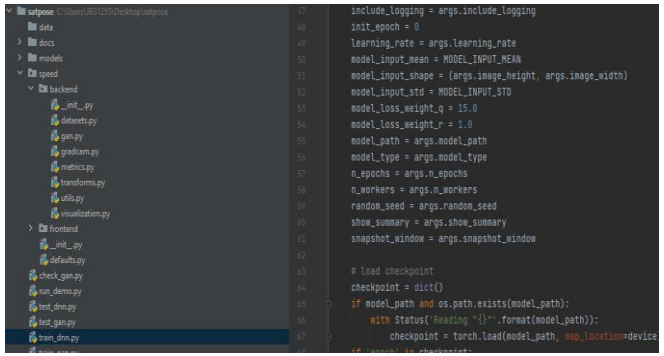


Model: Hierarchy

- Instructs analysis sequence
- Directs to relevant code repository

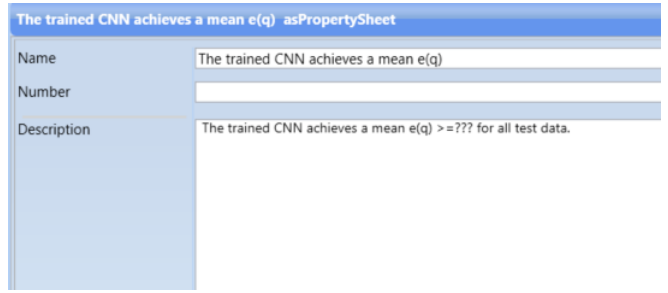
Code Repository

- Runs specified analysis
- Records results as intermediary



Model: Repository

- Records analysis results
- Checks against requirements



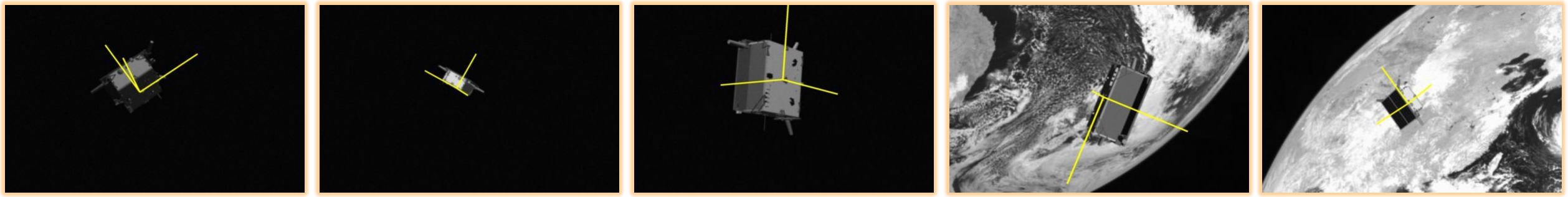
The Solution Team selected GENESYS for this use case, but other digital engineering environments should work

Trusted AI Framework Example: Thread 2 Pose Estimation Trust Features

Can a ResNet-50 CNN estimate the pose of a client spacecraft well enough for trust?

- Satellite Pose Estimation
  - *Estimate the 3D position and orientation of a client spacecraft from 2D imagery*
  - *Deep Learning with a ResNet-50 Convolutional Neural Network (CNN) from the literature*

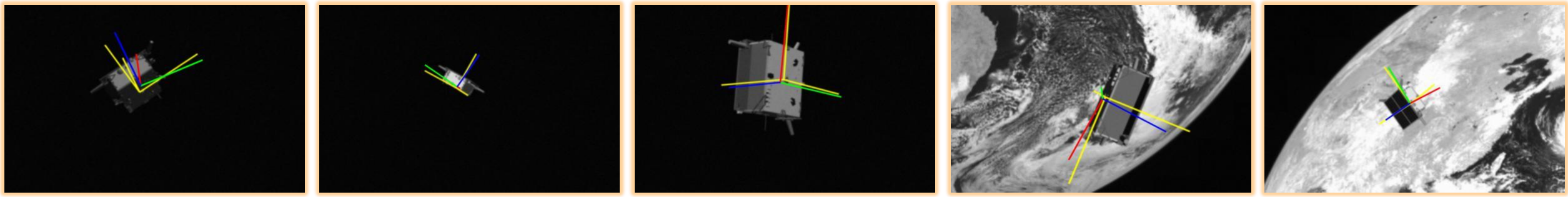
- Spacecraft Pose Estimation Dataset (SPEED)
  - *Created for Satellite Pose Estimation Challenge [Kisantal 2020]*
  - *Stanford University, Space Rendezvous Laboratory (SLAB)*
  - *Images of Tango spacecraft from PRISMA mission*
  - *First publicly available dataset for spacecraft pose estimation*
  - *Includes both synthetic and real images (mostly synthetic images)*
    - 12,000 training images (labeled); 2,998 synthetic test images (not labeled)



Prototype Satellite Pose CNN

Overview

- Training a Regression Model in Python (train_dnn.py) (speed_resnet50.pth)
  - *ResNet-50 architecture [He 2015]*
  - *Pose estimation is a regression problem (not classification)*
    - Estimate orientation *q* and position *r*
  - *Training objective (loss): minimize L_1 error between truth and predictions*
    - Orientation L_1 loss: $\theta_q = |\hat{q} - q|$
    - Position L_1 loss: $\theta_r = |\hat{r} - r|$
    - Combined L_1 loss: $\theta = 15 \times \theta_q + \theta_r$
    - Implemented in Python with PyTorch machine learning library
  - *Model pre-trained with ImageNet weights [Russakovsky 2015]*



Stability

Predictions are consistent when provided inputs fall within routine ranges

- Evaluation (test_dnn.py)
  - *Evaluate performance statistics against an independent and identically distributed dataset*
  - *Training dataset split: 80% training, 20% validation*
    - Model trained with 9,600 images from training set
    - Model evaluated against 2,400 unseen images from training set

- Metrics
  - *L_1 loss for both orientation and position*
  - *Error metrics from Satellite Pose Estimation Challenge [Kisantal 2020]*
    - Orientation error: $e_q = 2 \times \arccos(\langle \hat{q}, q \rangle)$
    - Normalized position error: $e_r = {|\hat{r} - r|_2}/{|r|_2}$
    - Combined pose error: $e = e_q + e_r$

Stability

How stable is the prototype CNN?

- Prototype Results
  - *Orientation L_1 loss (θ_q):* 0.3180
  - *Position L_1 loss (θ_r):* 0.3977
  - ***Orientation error (e_q):*** **1.1732**
  - ***Normalized position error (e_r):*** **0.0936**
  - *Combined pose error (e):* 1.2667
- Comparison
  - *State-of-the-art (2020) comparison appears below:*

DETAILED RESULTS OF THE TOP TEN SUBMISSIONS COMPARED TO THE SLAB'S BASELINE PERFORMANCE

Team	E_{syn}	E_{real}	e_q [°]	e_t [m]	PnP
1. UniAdelaide [47]	**0.0094**	0.3752	**0.41 ± 1.50**	**0.032 ± 0.095**	Yes
2. EPFL_cvlab	0.0215	**0.1140**	0.91 ± 1.29	0.073 ± 0.587	Yes
3. pedro_fairspace [48]	0.0571	0.1555	2.49 ± 3.02	0.145 ± 0.239	No
SLAB Baseline [49]	0.0626	0.3951	2.62 ± 2.90	0.209 ± 1.133	Yes
4. Team_Platypus					No
5. motokimura1					No
6. Magpies					No
7. GabrielA					No
8. stainsby					No
9. VSI_Feeney					No
10. jblumenkamp					Yes

Best results for each metric are highlighted with bold fonts. The mean and the standard deviation of the orientation errors (e_q) as in (3) and position errors (e_t) as in (1) are measured on the synthetic test set.

Performance falls within **Top 3** for Satellite Pose Estimation Challenge

Confidence and Uncertainty

Levels of confidence and uncertainty bounds can be discerned

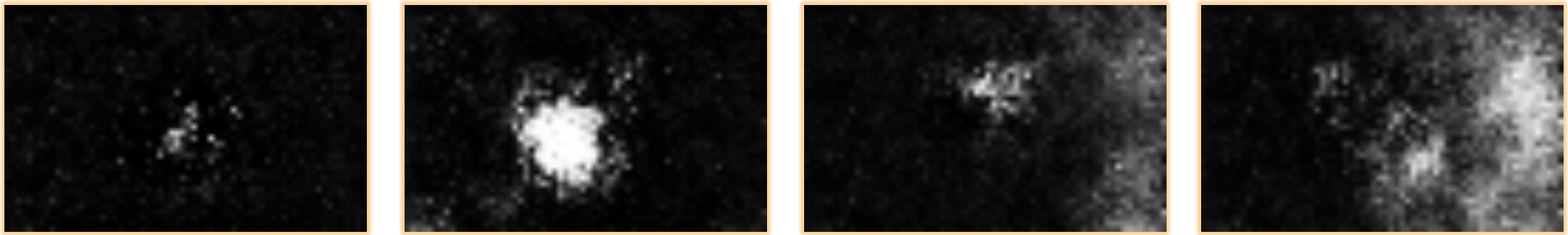
- Determining Out-of-distribution Inputs [Gawlikowski 2021]
  - *Performance metrics are only known for training/validation data*
  - *Performance for data that deviate from training/validation is uncertain*
  - *Method needed for determining if given images are close enough to training images*

- Generative Adversarial Network (GAN) (train_gan.py) (speed_dcgan.pth)
  - *Deep Convolutional Generative Adversarial Network (DCGAN) [Radford 2016]*
  - *Separate DNN with two sub-networks: a Generator & Discriminator*
  - *Generator: creates similar (but fake) images*
  - *Discriminator: determines if an image is real or fake*

Confidence and Uncertainty: the Role of Generators and Discriminators

The Discriminator helps identify Uncertainty by determining out-of-distribution inputs

- DCGAN Generator
  - *Attempts to create images similar to training set from random noise*
  - *Note: these are low fidelity images*



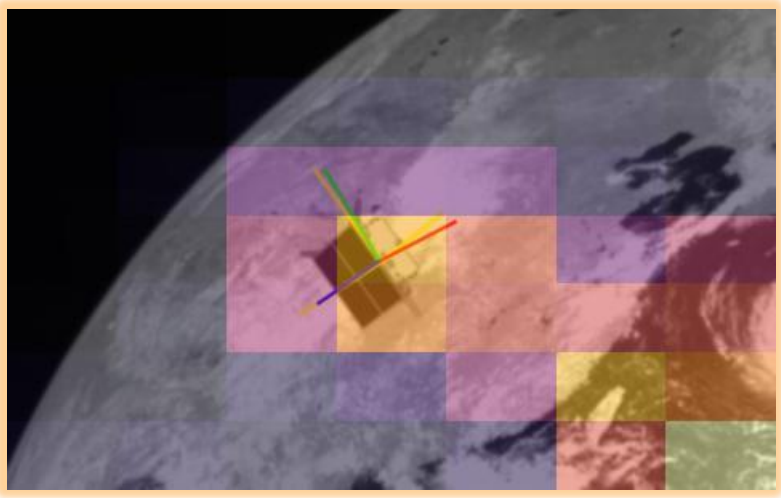
- DCGAN Discriminator
  - *Can classify images that are real (matching distribution) or fake (out of distribution)*
  - *Evaluation results (test_gan.py)*
    - Identifying *real training* images (*79% accuracy*)
    - Identifying *fake generated* images (*66% accuracy*)

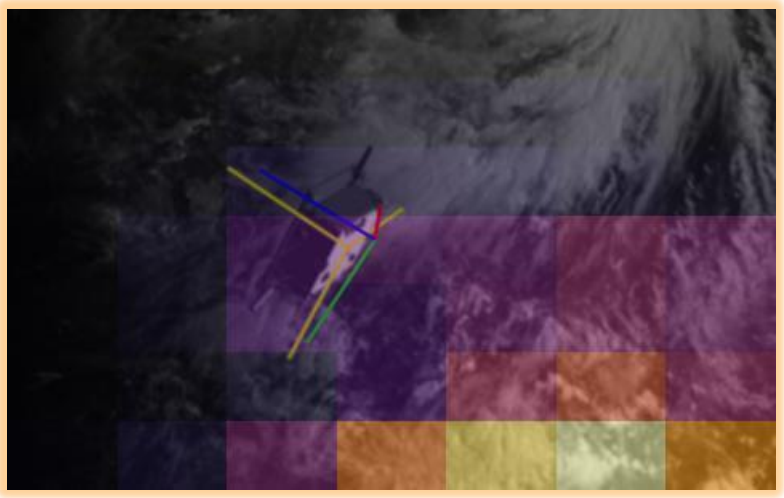
Quantifying Confidence is challenging for regression problems like this: there's no general assessment that a neural network's prediction is "correct." It is necessary to define margins of error and instead measure confidence relative to those margins of error.

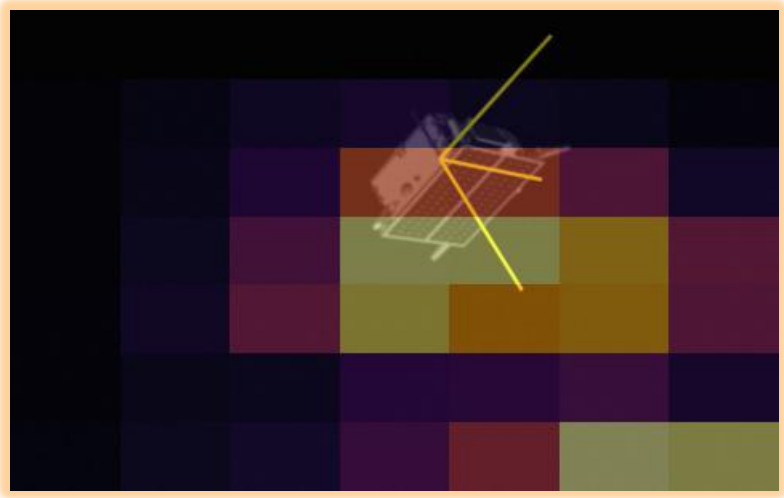
Interpretability

Users can easily understand the underlying causes of how responses were formulated

- Visualizing Focal Regions (viz_gradcam.py)
  - *Gradient-based Localization (Grad-CAM) [Selvaraju 2019]*
  - *Highlights image regions that most significantly impacted predictions*
  - *Enables users to reject predictions when focal regions seem suspect*





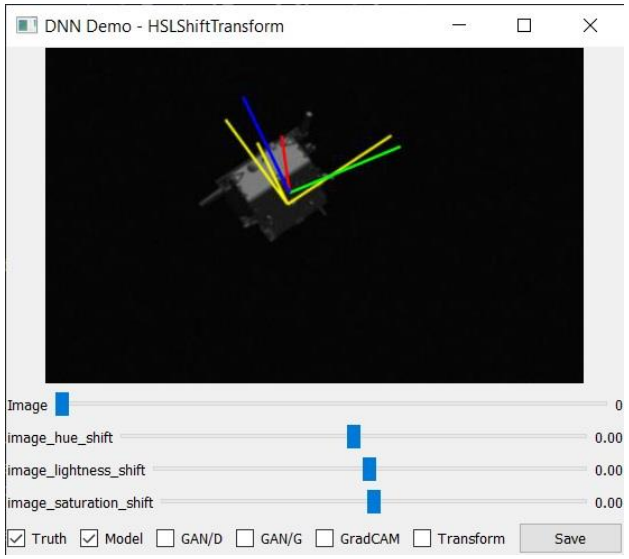


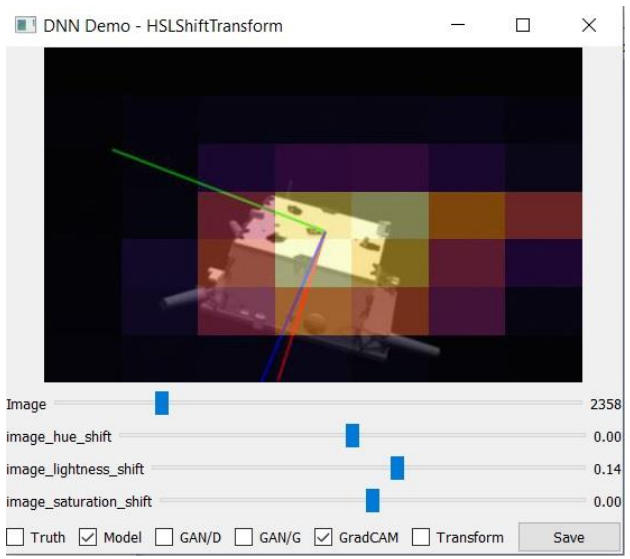
Tools like this illustrate to human users where within the imagery the CNN focuses to make its determinations

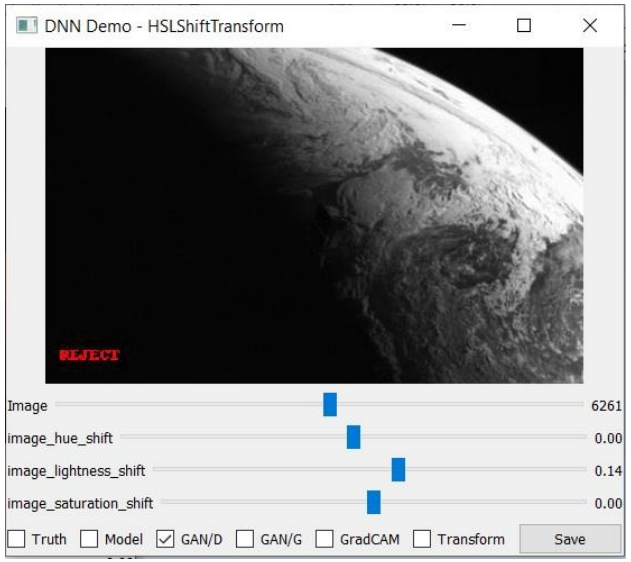
Demonstrating the Prototype Satellite Pose CNN in Python

Now the Solution Team can integrate the prototype CNN into digital engineering to illustrate the trusted AI framework and connect its tenets with requirements and other program artifacts

- Graphical Interface (run_demo.py)
  - *Enables user to process images in real time with Satellite Pose CNN*
    - Displays both “true” and “estimated” pose (*computed in real time*)
    - Enables processing with *DCGAN* to determine “accept” / “reject”
    - Enables Grad-CAM projection for *interpretability*
    - Enables image transformations to observe *robustness* to change







The dashboard with its slider bars facilitates Interpretability by showing the user how performance changes based on manual parameter adjustments