Digital Thread for Aerospace Design & Engineering

Satish Thokala
Aerospace and Defense Industry Manager

Outline

- Digital Threading Through Traditional Engineering
- Continuity Through Digital Engineering
- Examples and Case Studies
- Digital Thread in AI-ML Workflow

MathWorks

MathWorks is the leading provider of technical computing software

- **5 million users**
- Installations at **100,000+ sites** in **185 countries**
- Used for teaching and research by **6500 universities**
- **$1.25B revenue** in 2021
- **6000 staff** including **2500 engineers**
- **Private, profitable every year** since founding in 1984

MATLAB, the language of engineers and scientists, is a programming environment for algorithm development, data analysis, visualization, and numeric computation.

Simulink is a block diagram environment for simulation and Model-Based Design of multidomain and embedded engineering systems.

Headquarters
Natick, MA USA

North America
United States

Europe
France
Germany
Ireland
Italy
Netherlands
Spain
Sweden
Switzerland
UK

Asia-Pacific
Australia
China
India
Japan
Korea

Customers in many industries innovate with MathWorks software

Aerospace and Defense

Automotive

Communications

Software and Internet

Financial Services

Complex multi-domain systems, software-defined and autonomous, model-based and data-driven

Comms infrastructure, plus all types of connected systems across industries

Big Data, Agile, DevOps, integration with IT systems

Railway Systems

Energy Production

Electronics

Neuroscience

Biological Sciences

Modernization, often on legacy platforms, becoming data-centric for optimization and maintenance

Wide range of compute platforms, many kinds of HW/SW integration

Collaboration between science, engineering, and informatics

Process Industries

Industrial Machinery

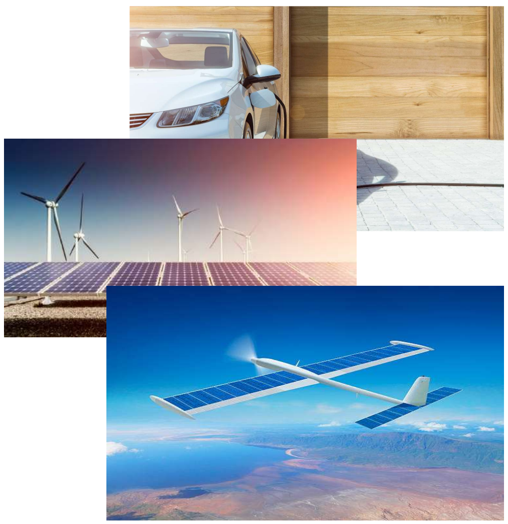
Semiconductors

Medical Devices

Biotech and Pharmaceutical

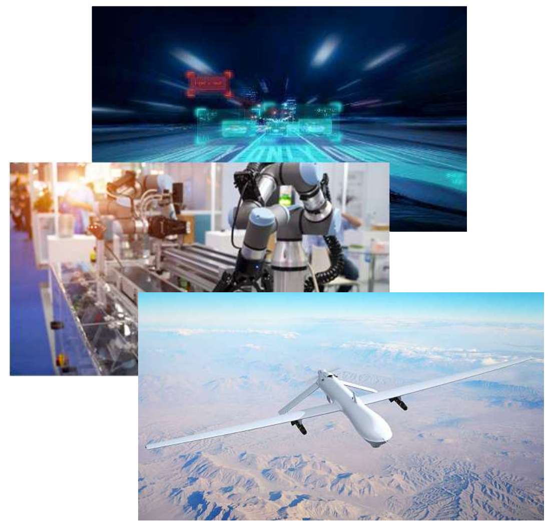
Industries megatrends

Electrification

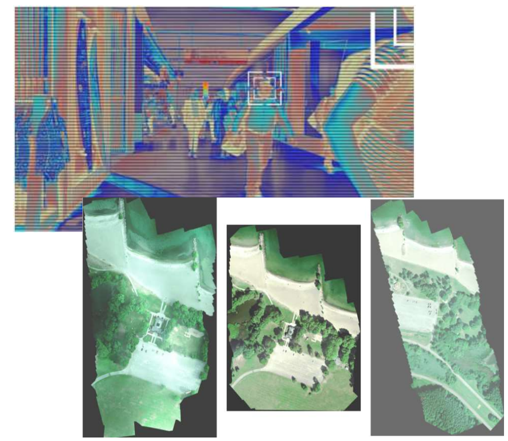


Connectivity

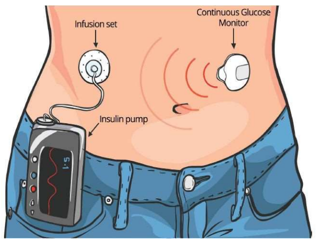
Autonomous



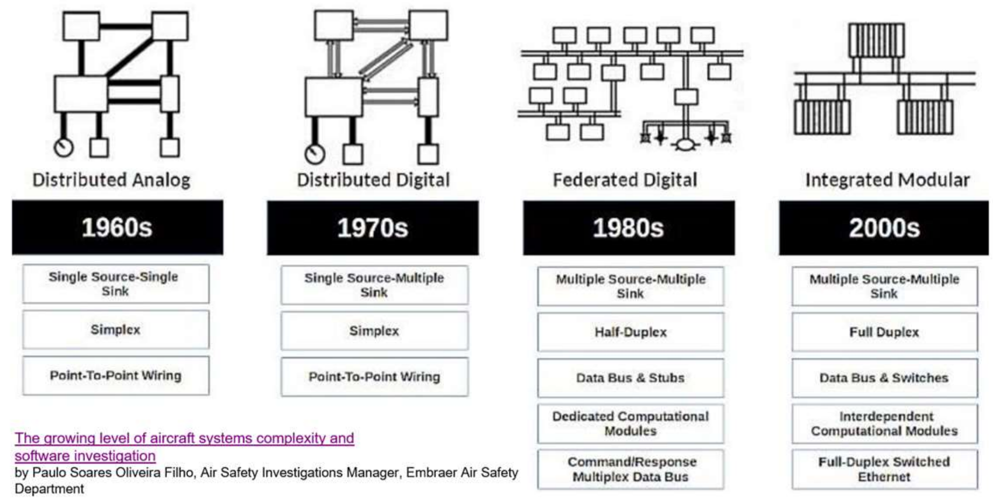
Artificial Intelligence



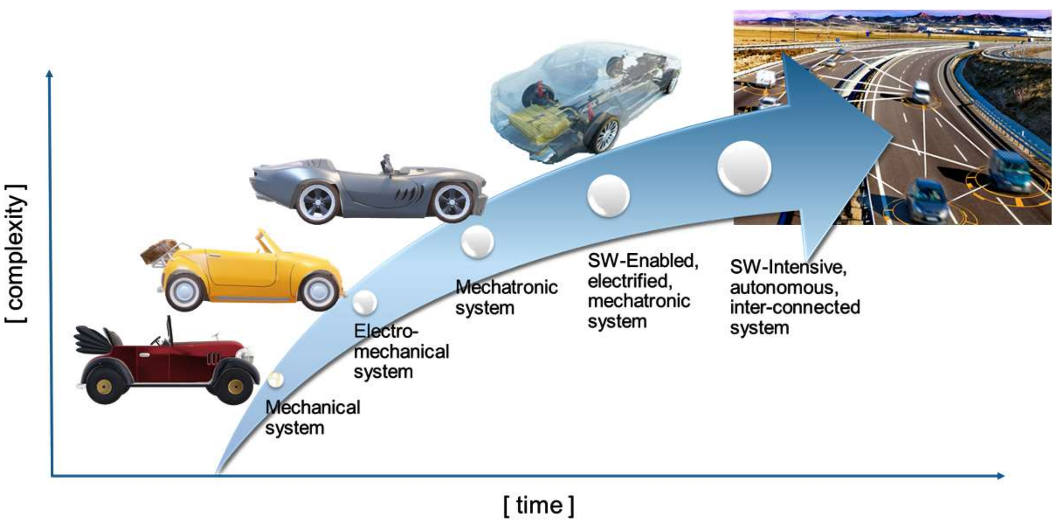
Industries challenges: Increasing systems complexity



Growing level of integration in avionics architectures

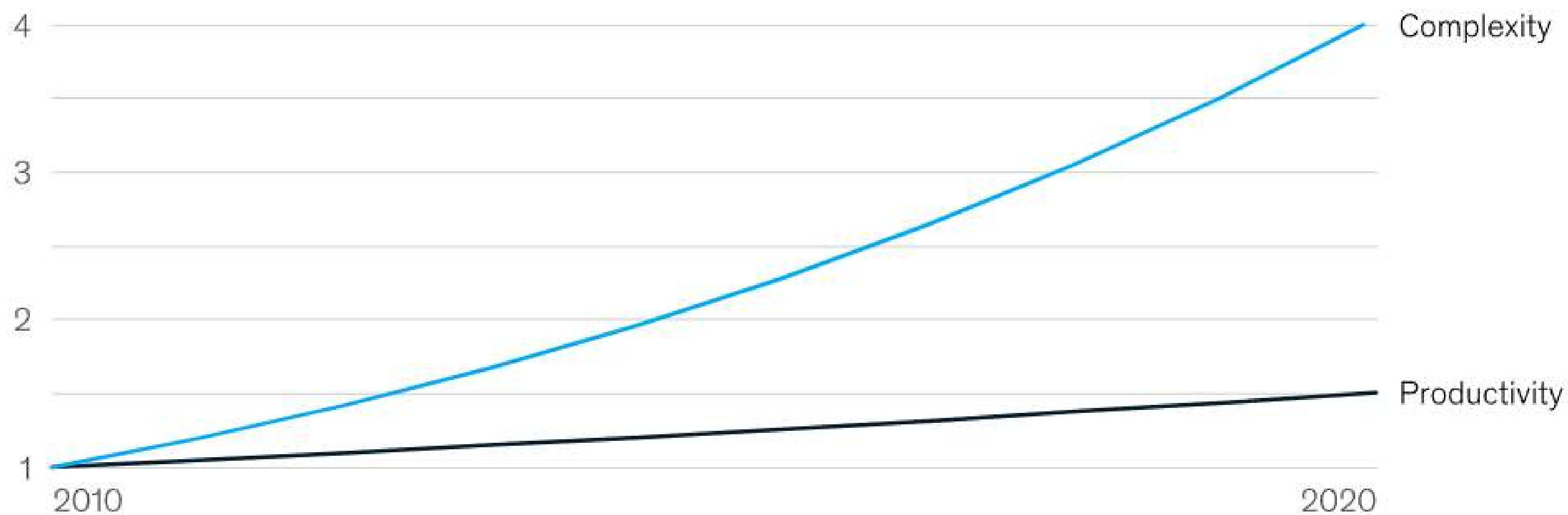


The growing level of aircraft systems complexity and software investigation
by Paulo Soares Oliveira Filho, Air Safety Investigations Manager, Embraer Air Safety Department



Falling into the complexity trap?

Growth of software complexity and productivity in automotive systems, relative and indexed[1]



[1] "When code is king: Mastering automotive software excellence," February 17, 2021, McKinsey.com.
[2] Thousands of source lines of code.
Source: Paulo Soares Oliveira Filho, "The growing level of aircraft systems complexity and software investigation," International Society of Air Safety Investigators, 2020, isasi.org; McKinsey's SoftCoster embedded software project database

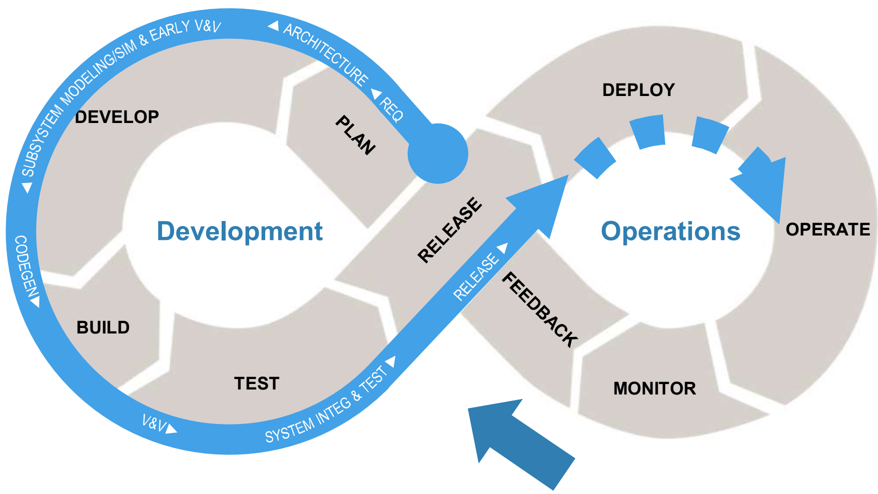
Workflow challenges

OEM | Suppliers | OEM

Mechanical system
Communications system
Electrical system

System Requirements	System Functionality and Architecture	Design	Implementation	System Integration and Test	Operations and Sustainment
REQUIREMENT DOCS	**PAPER SPECS**	**PHYSICAL PROTOTYPES**	**MANUAL CODING**	**TRADITIONAL TESTING**	**SYSTEMS IN OPERATION**
Difficult to analyze	Easy to misinterpret	Incomplete behavior	Time consuming	Design and integration issues found late	Improving uptime
Difficult to manage as they change	Difficult to integrate with design	Expensive and time-consuming to build	Introduces defects and variance	Cannot trace back to requirements	Hard to feed insights back to developers
	Slow to revise when requirements change	Prevents rapid iteration	Difficult to reuse	Difficult to verify correctness	Difficult to get value from operational data

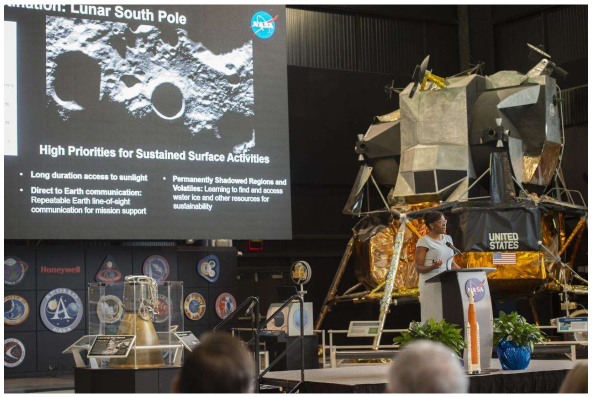
Rising demands to extend DevOps to systems, not only software



Companies are adopting various approaches to deal with these challenges

Challenge	Approach	Benefit
Communication of complex systems and requirements	SysML	Modeling
Change management	Digital Thread	Impact Analysis
Rapidly evolving systems	Agile & DevOps	Rapid delivery of value

NASA and ESA have objectives to advance digital engineering



Revision: Initial Release	Document No HLS-RQMT-001
RELEASE DATE: September 27, 2019	Page: 7 of 315
Title: HLS Requirements Document (SRD)	

2 Documents

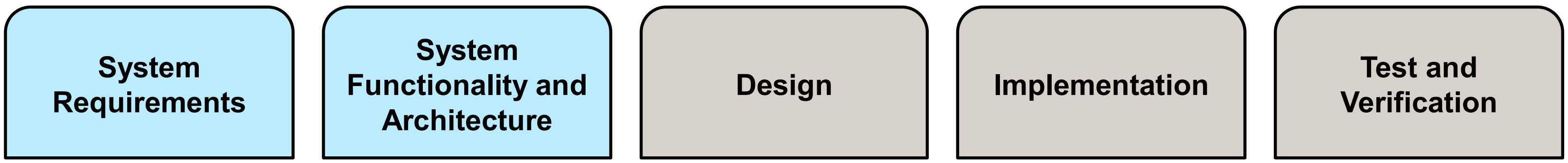
For the purpose of this document, the term ‘document’ can also refer to ‘digital artifacts,’ ‘models,’ or ‘viewpoints’ as needed to convey and exchange configuration managed data or information. An objective of the HLS Program is to advance towards a digital engineering environment and away from the traditional document-based approach for capturing data, reports and baselines.

ESA Agenda 2025:

“ESA will therefore digitalise its full project management, enabling the development of digital twins, both for engineering by using Model Based System Engineering, and for procurement and finance, achieving full digital continuity with industry.”

MathWorks®

Digital engineering in practice



Digital engineering in practice

System Requirements

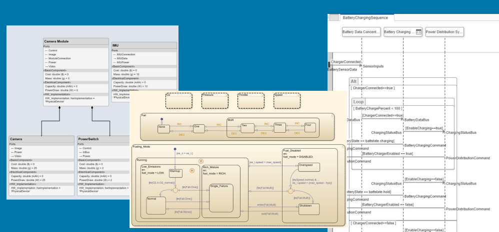
System Functionality and Architecture

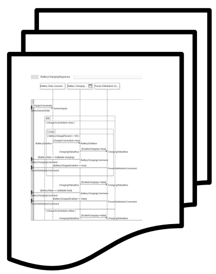
Design

Implementation

Test and Verification

SysML



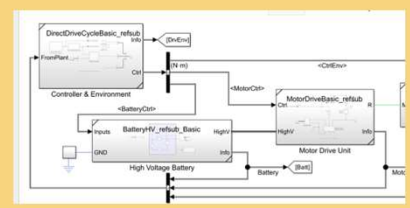


Model-Based Design

Software

Mechanical

Electronics



Digital engineering in practice

System Requirements

System Functionality and Architecture

Design

Implementation

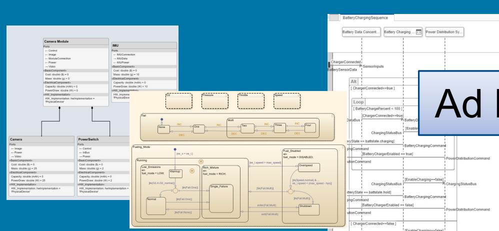
Test and Verification

Inconsistent semantics of models breaks the digital thread and loses valuable engineering information

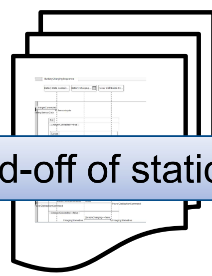
Inability to automate tasks such as requirements verification

Difficult to leverage simulation for systems analysis

SysML



Ad hoc hand-off of static documents

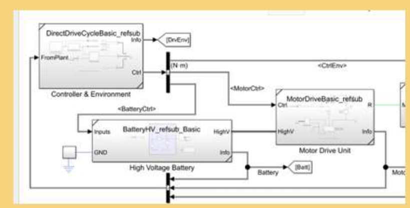


Model-Based Design

Software

Mechanical

Electronics



Difficult to trace authority between systems and design

Companies are adopting various approaches to deal with these challenges

Challenge	Approach	Benefit
Communication of complex systems and requirements	**SysML**	**Modeling**
Change management	**Digital Thread**	**Traceability**
Rapidly evolving systems	**Agile & DevOps**	**Rapid delivery of value**

Companies are adopting various approaches to deal with these challenges

Challenge	Approach	Benefit	Drawback
Communication of complex systems and requirements	**SysML**	**Modeling**	Static diagrams must be updated manually.
Change management	**Digital Thread**	**Traceability**	No automation of resolving differences.
Rapidly evolving systems	**Agile & DevOps**	**Rapid delivery of value**	May not leverage traceability or automation.

Requires Human-in-the-loop

Companies are adopting various approaches to deal with these challenges

Common Modeling Semantics

Authoritative Source of Truth

Automation

Modeling · Automation · Authoritative Source of Truth

Building a digital engineering ecosystem

System Requirements	System Functionality and Architecture	Design	Implementation	Test and Verification

Common Modeling Semantics

Authoritative Source of Truth

Automation

Modeling ▪ Automation ▪ Authoritative Source of Truth

Building a digital engineering ecosystem

Common Modeling Semantics

Authoritative Source of Truth

Automation

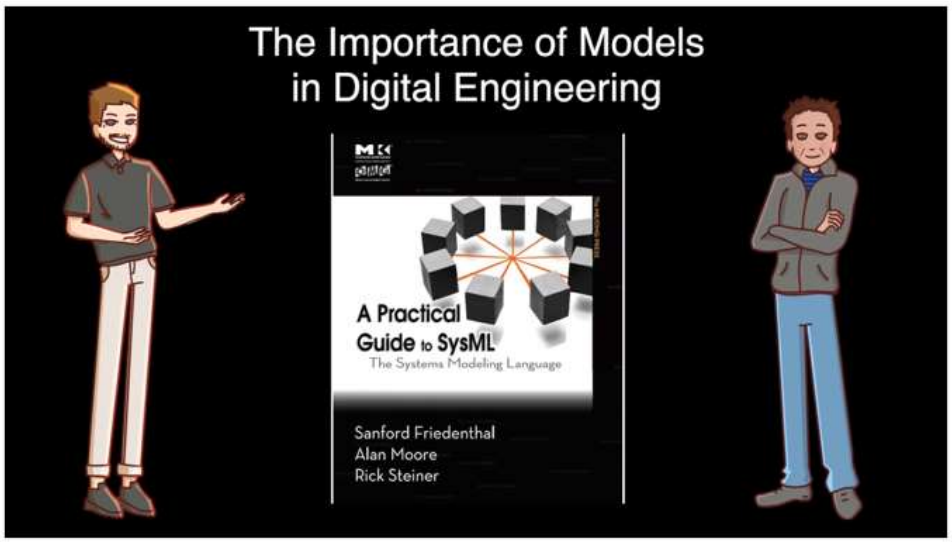
Digital Engineering Ecosystem

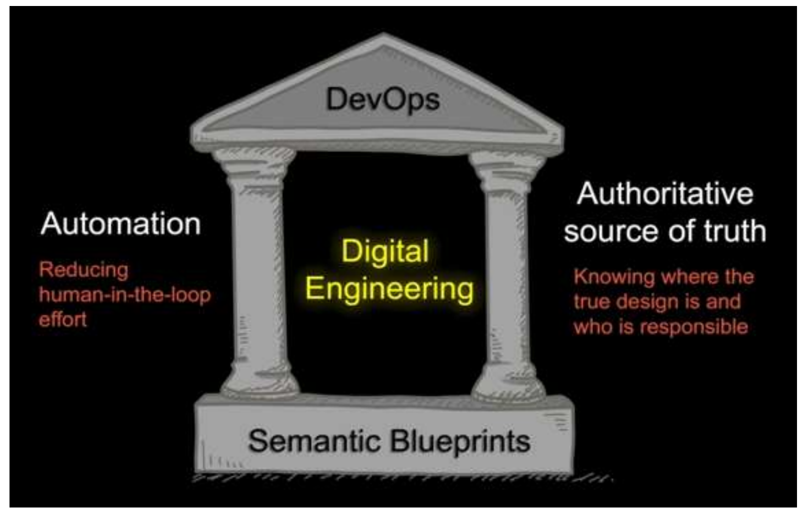
System Requirements | System Functionality and Architecture | Design | Implementation | Test and Verification

MATLAB EXPO

Why Models Are Essential to Digital Engineering

Digital engineering is a trending industry buzzword. It's something that organizations strive to embrace and tool vendors claim to implement. But what is the practical reality behind the buzz? What are some of the essential aspects of an engineering ecosystem that actually provide the value promised? In this talk, Brian Douglas of Control Systems Lectures and MATLAB® Tech Talks, and Alan Moore, one of the original authors of SysML and co-author of "A Practical Guide to SysML," discuss exactly these questions and show how models are a central and essential element of digital engineering.





https://www.mathworks.com/videos/why-models-are-essential-to-digital-engineering-1652969543566.html

Modeling ▪ Automation ▪ Authoritative Source of Truth

Building a digital engineering ecosystem

Common Modeling Semantics

Authoritative Source of Truth

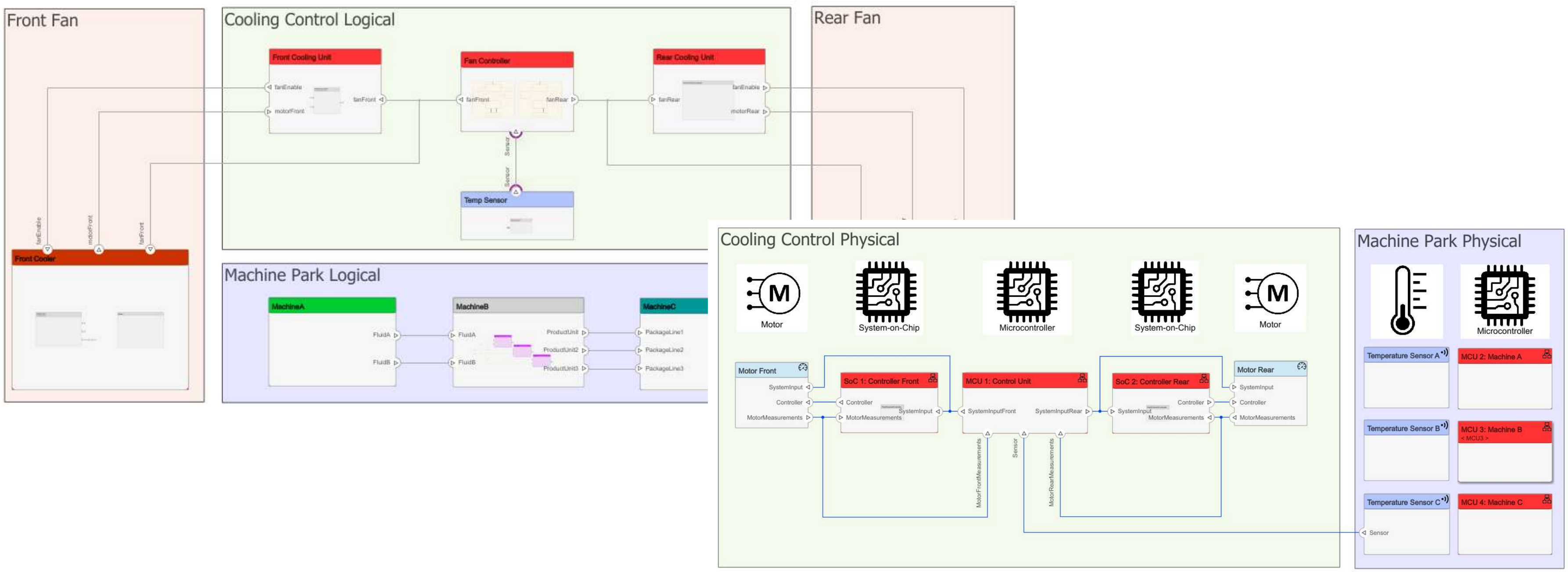
Automation

Digital Engineering Ecosystem

System Requirements | System Functionality and Architecture | Design | Implementation | Test and Verification

Modeling

Modeling semantics rich enough for descriptive modeling...

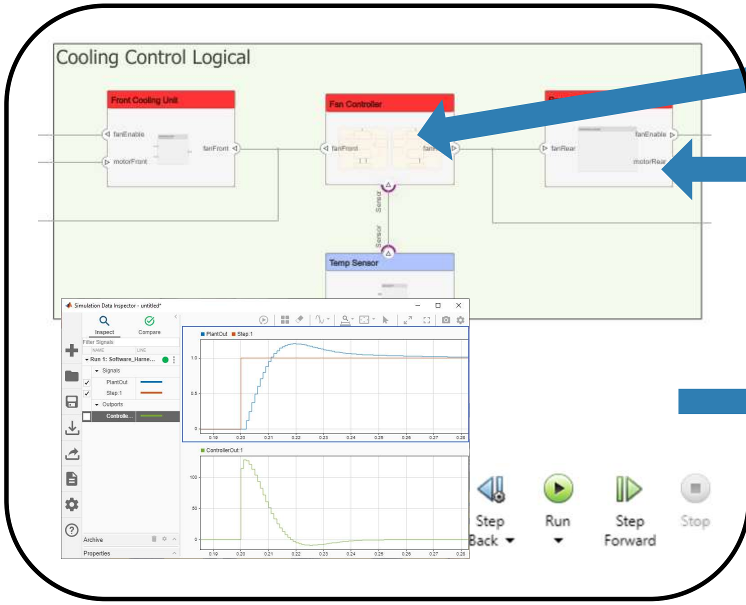


Digital Engineering Ecosystem

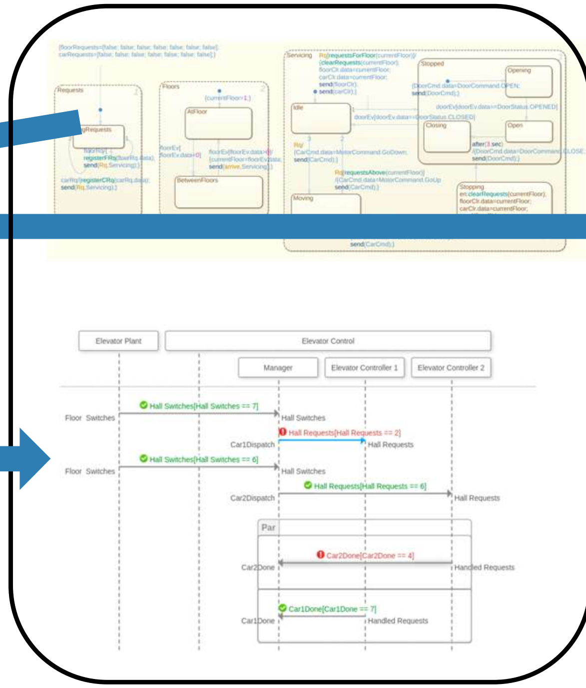
System Requirements | System Functionality and Architecture | Design | Implementation | Test and Verification

Modeling

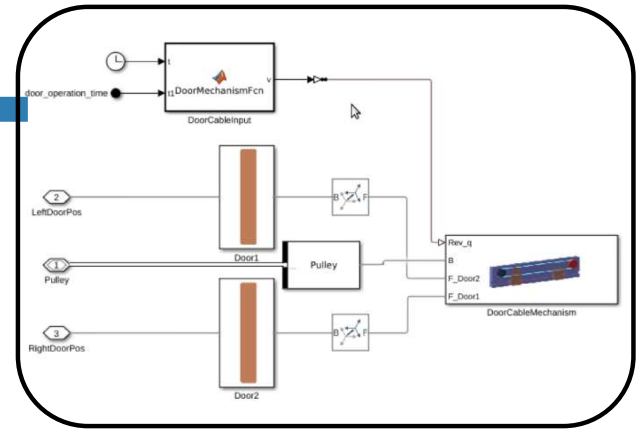
…and precise enough for detailed design, simulation and analysis.



Descriptive Diagrams



Behavior Diagrams



Design Models

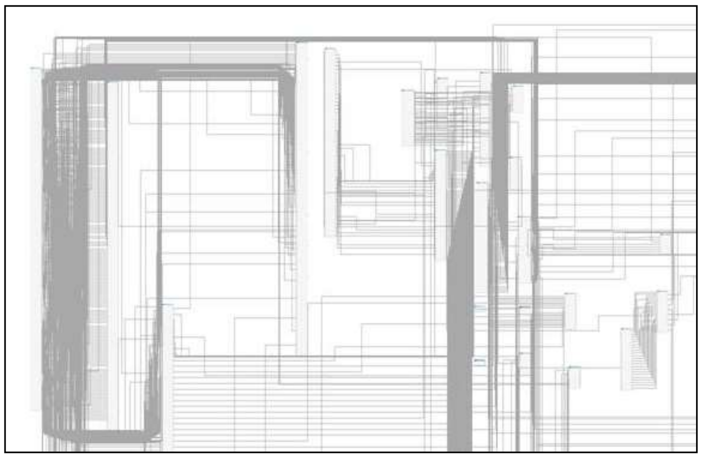
Digital Engineering Ecosystem

System Requirements	System Functionality and Architecture	Design	Implementation	Test and Verification

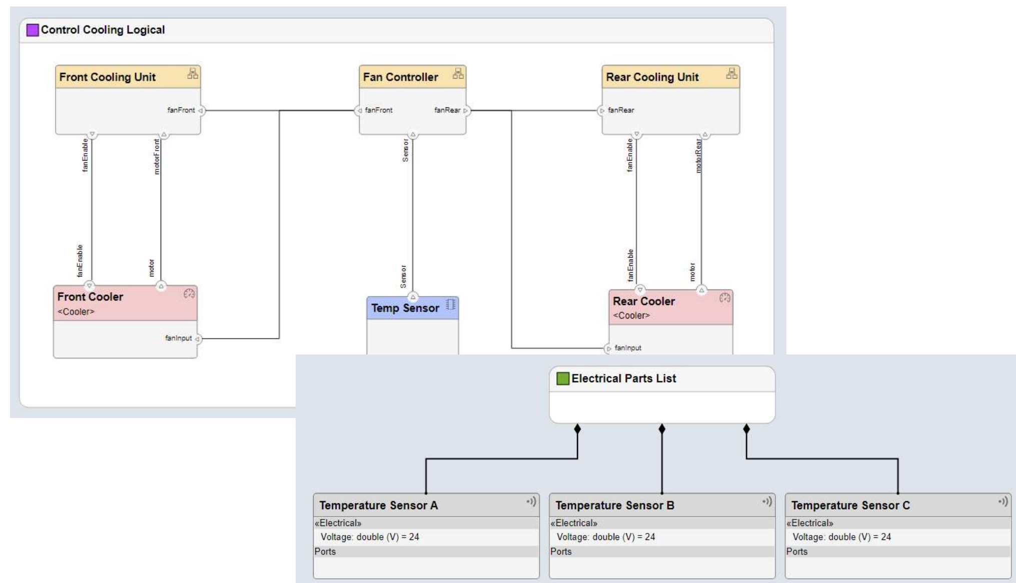
Modeling

Modeling semantics rich enough for descriptive modeling...

Full System Model



Filtered Views

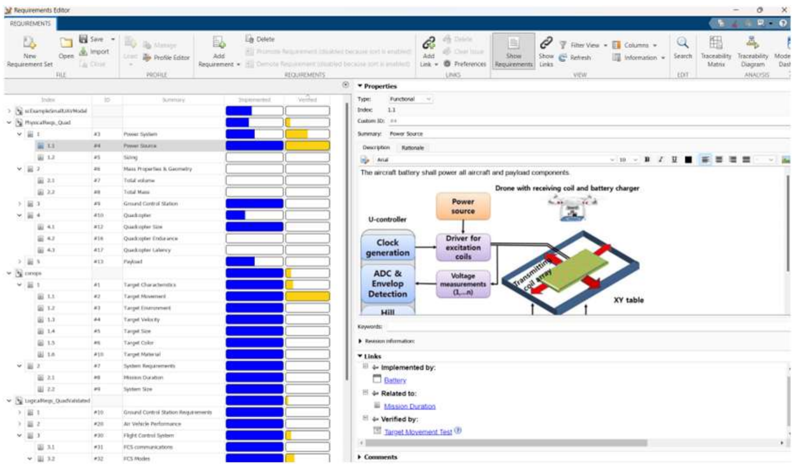


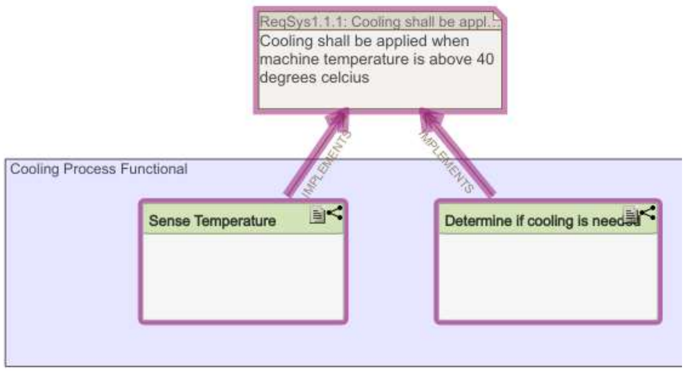
Digital Engineering Ecosystem

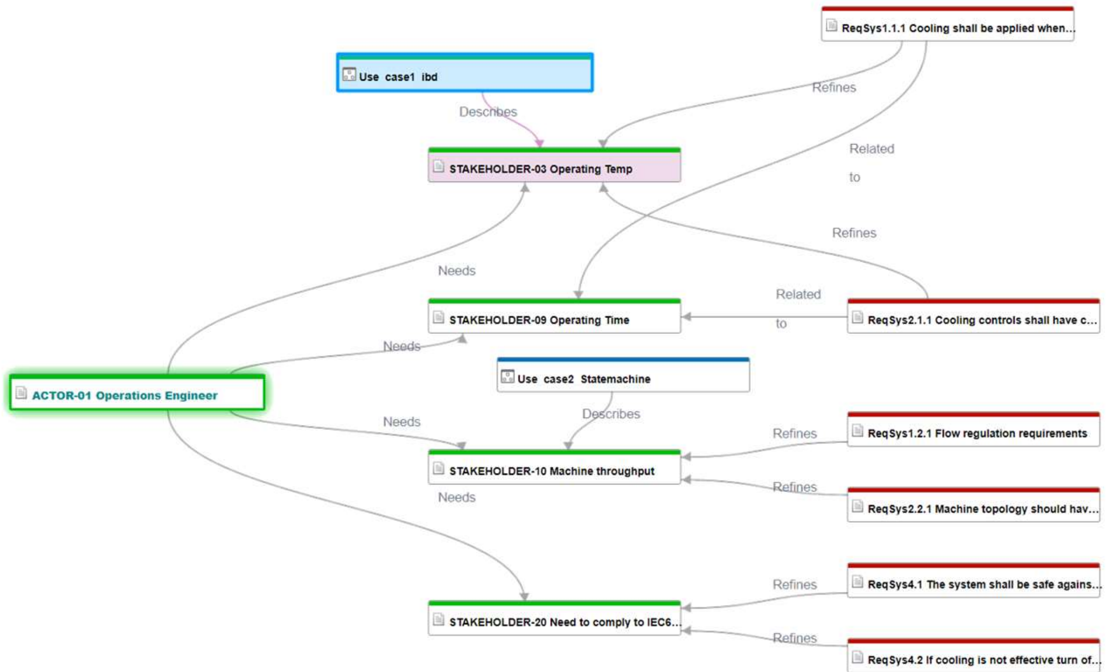
System Requirements | System Functionality and Architecture | Design | Implementation | Test and Verification

Authoritative Source of Truth

Full traceability of requirements, architectures, and design







Digital Engineering Ecosystem

System Requirements	System Functionality and Architecture	Design	Implementation	Test and Verification

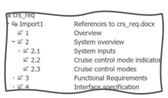
Authoritative Source of Truth

Automate analysis and assessments of linked artifacts

Link and Trace

Requirements

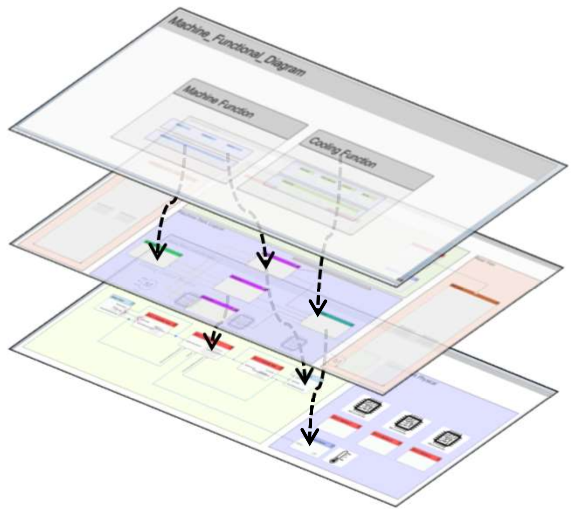
IBM DOORS



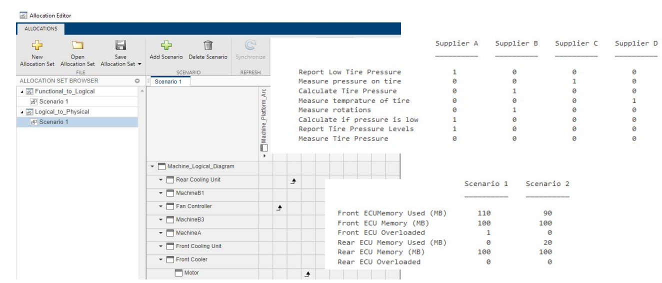
REQUIREMENTS TOOLBOX

Allocate

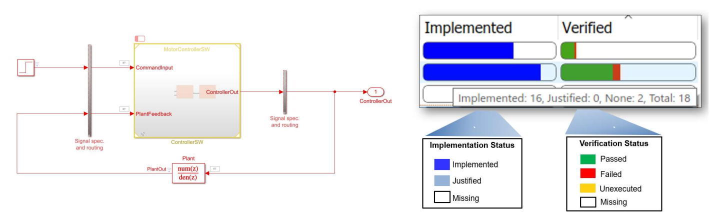
Architecture models
(functional, logical and physical)



View, Assess, and Compare



Design and Test

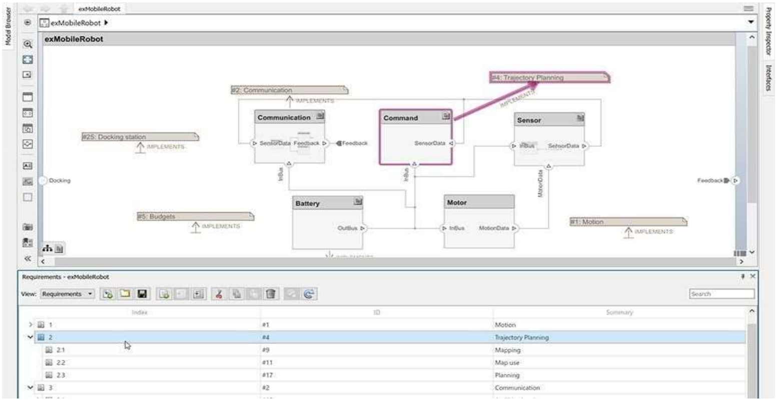


Digital Engineering Ecosystem

System Requirements | System Functionality and Architecture | Design | Implementation | Test and Verification

UAV design using digital engineering

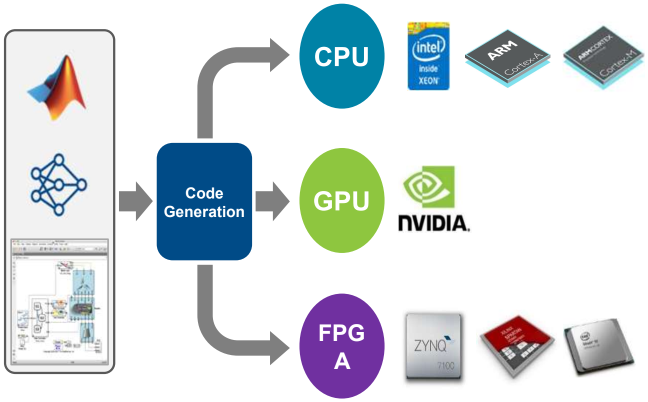
Digital Thread



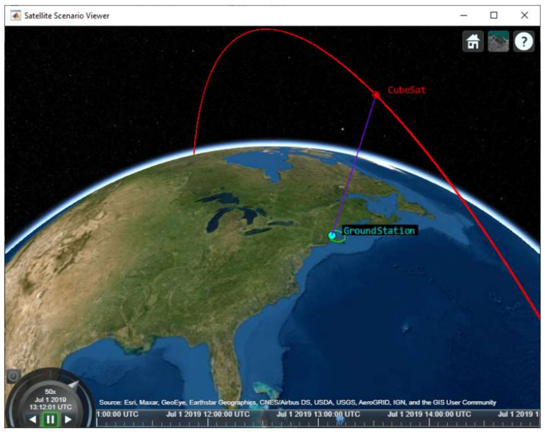
Real-time testing on hardware

3D visualization

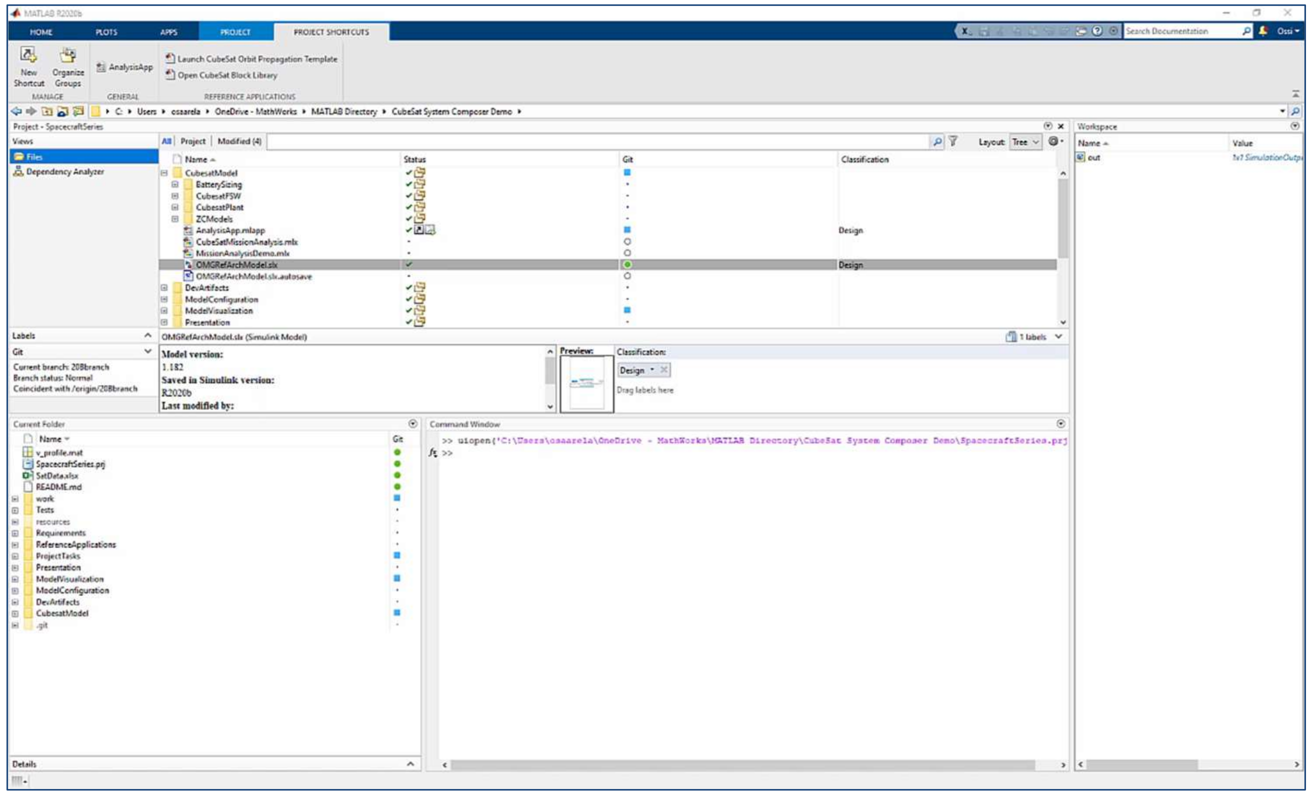
Deployment



Digital engineering for space systems

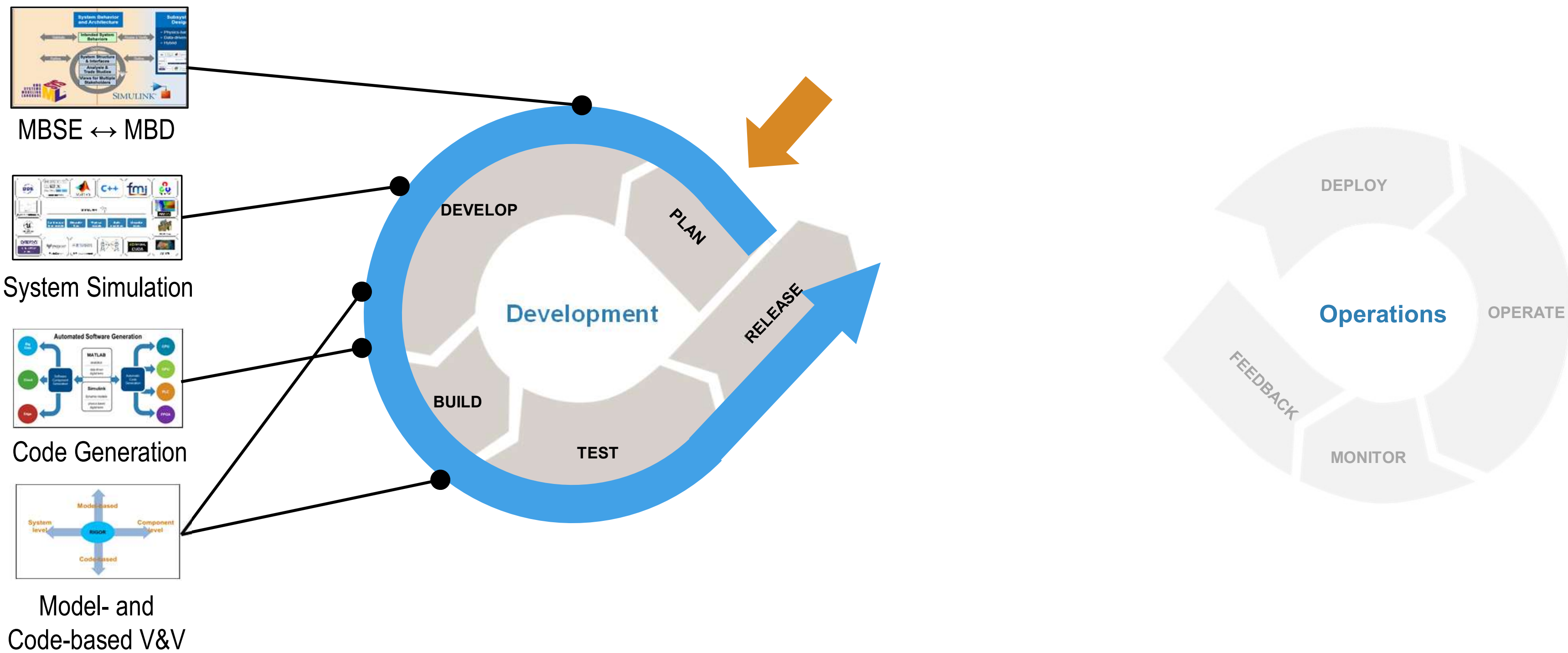


The system shall provide and store visual imagery of MathWorks headquarters [42.2775 N, 71.2468 W] 1 time daily at 10 meters resolution.



Automation

Integrate models at every level into CI pipeline

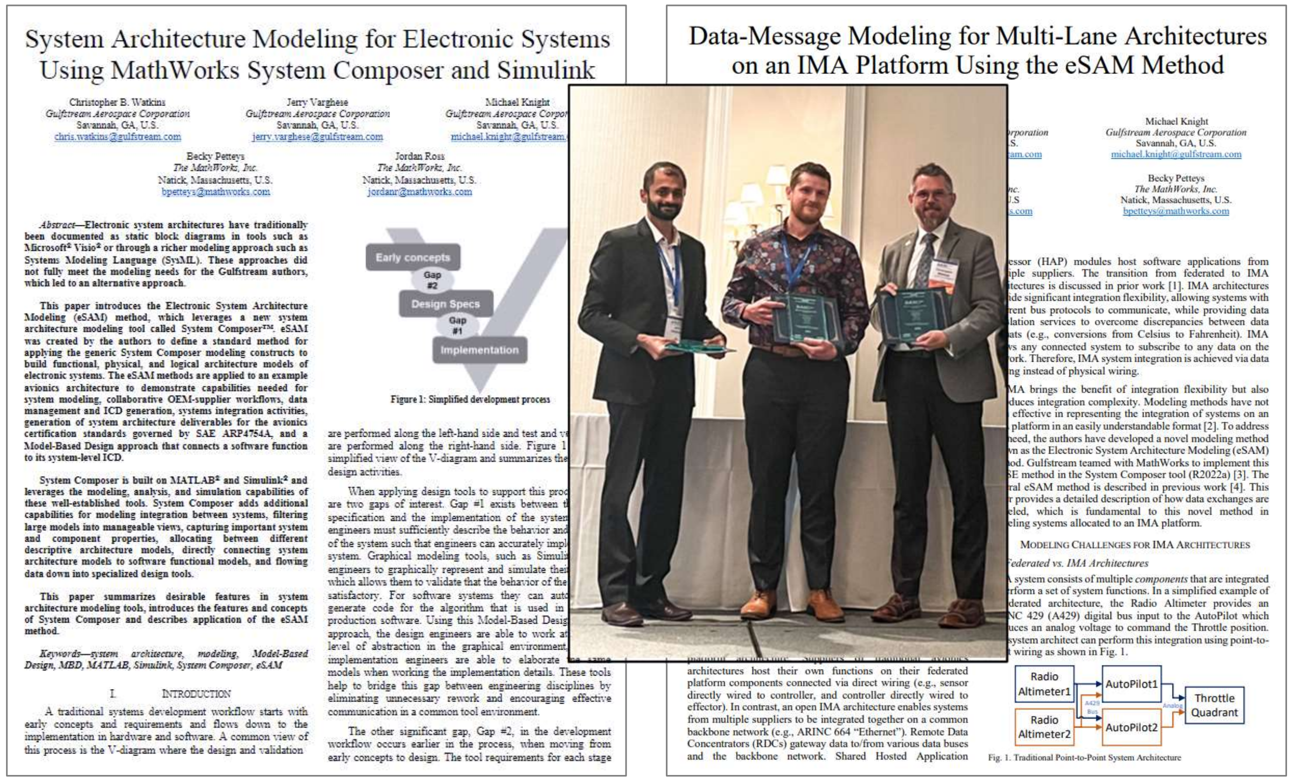


Digital Engineering Ecosystem

System Requirements | System Functionality and Architecture | Design | Implementation | Test and Verification

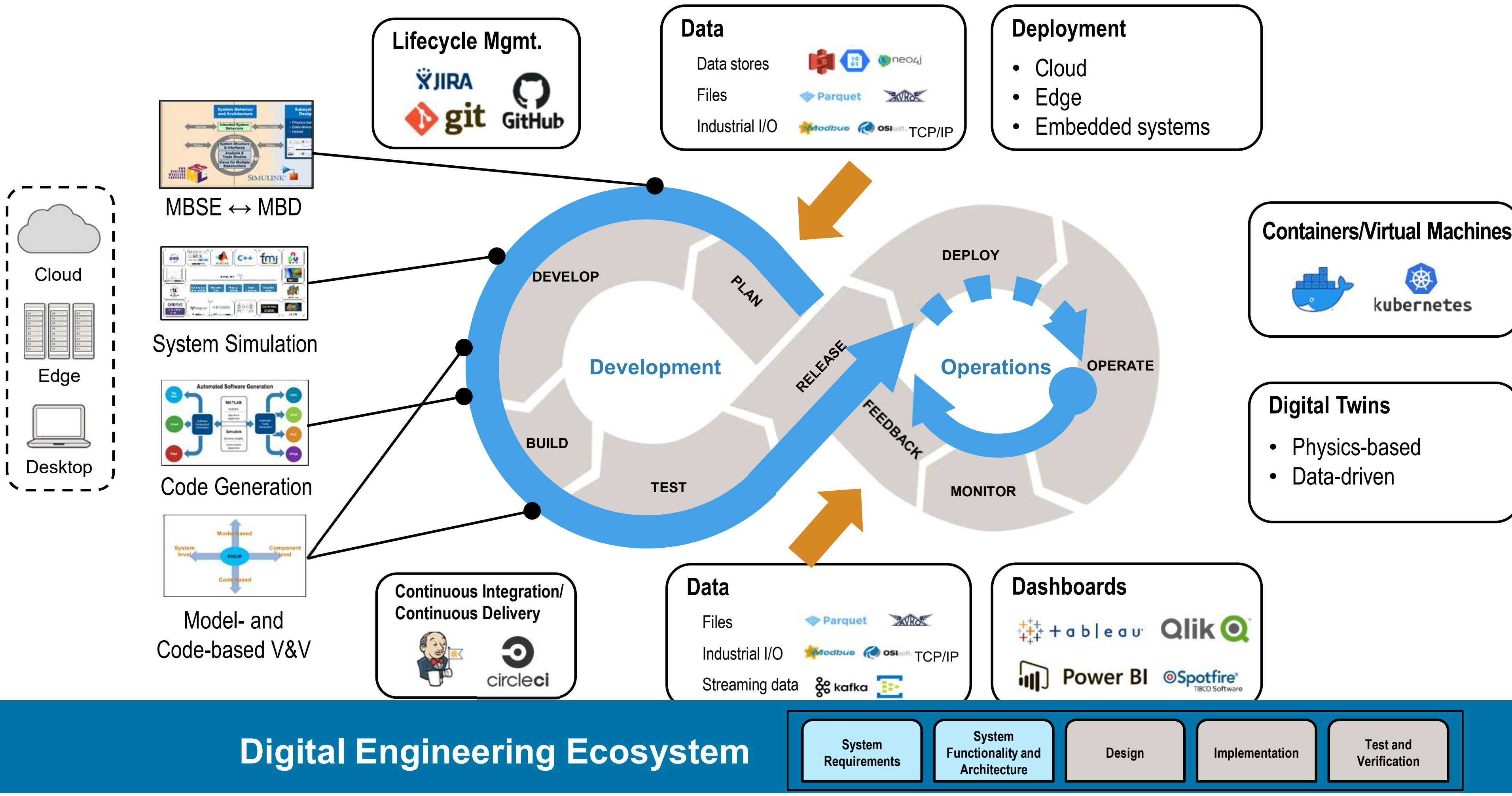
Gulfstream: Electronic System Architecture Modeling using Digital Engineering

> "System Composer adds additional capabilities for modeling integration between systems, …capturing important system and component properties, …directly connecting system architecture models to software functional models, and flowing data down into specialized design tools."



https://ieeexplore.ieee.org/document/9256753 https://ieeexplore.ieee.org/document/9925816

Software and systems – Agile to DevOps



Machine Learning workflow follows the same cycle

Lifecycle Mgmt

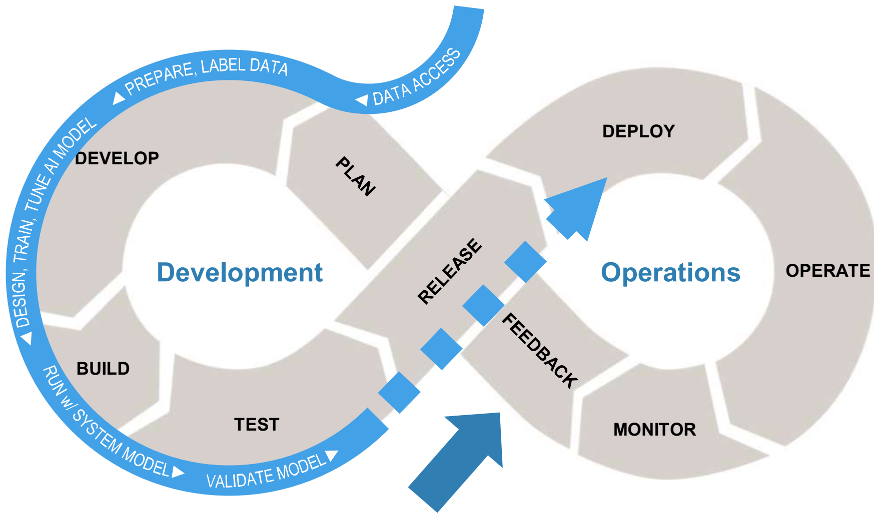
JIRA git GitHub

Data

Data stores neo4j
Files Parquet AVRO
Industrial I/O Modbus OSIsoft TCP/IP

Deployment

- Cloud
- Edge
- Embedded systems



Containers/Virtual Machines

kubernetes

Digital Twins

- Physics-based
- Data-driven

CI/CD

circleci

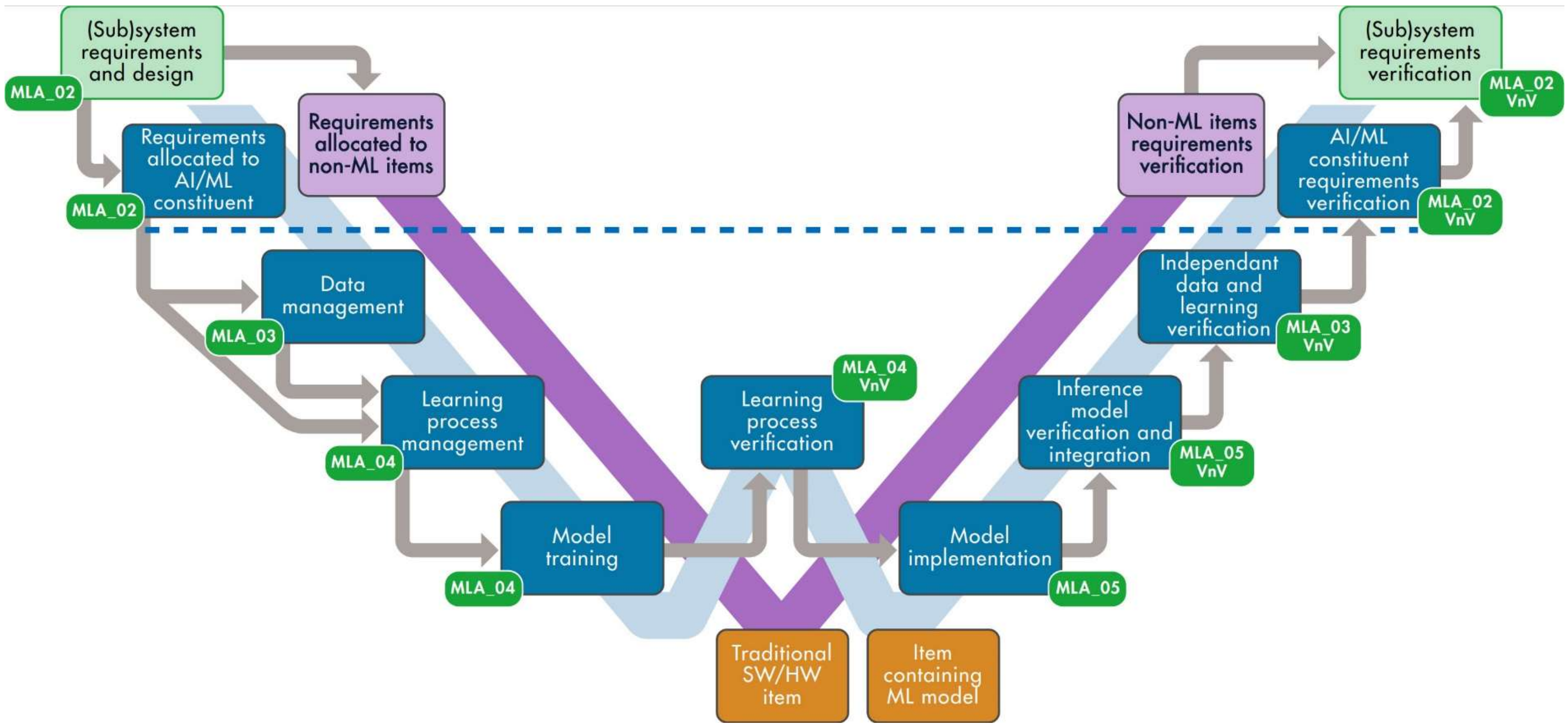
Data

Files Parquet AVRO
Industrial I/O Modbus OSIsoft TCP/IP
Streaming data kafka

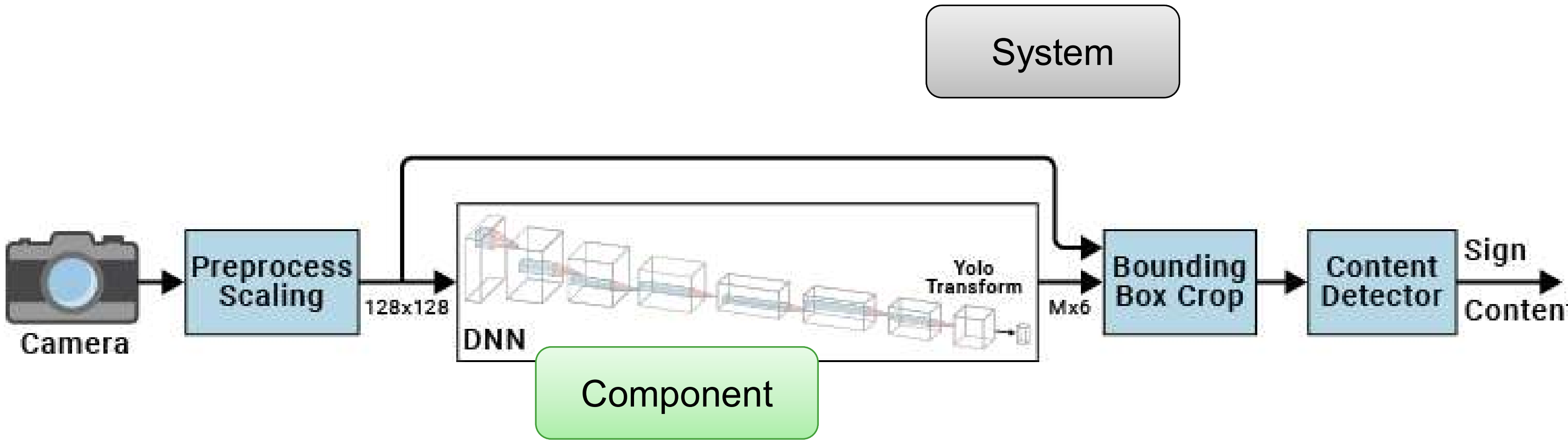
Dashboards

tableau Qlik
Power BI Spotfire TIBCO Software

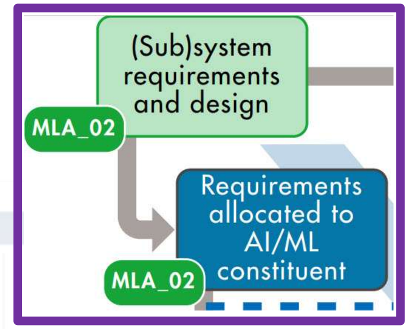
Digital Thread in an Airborne Deep Learning System

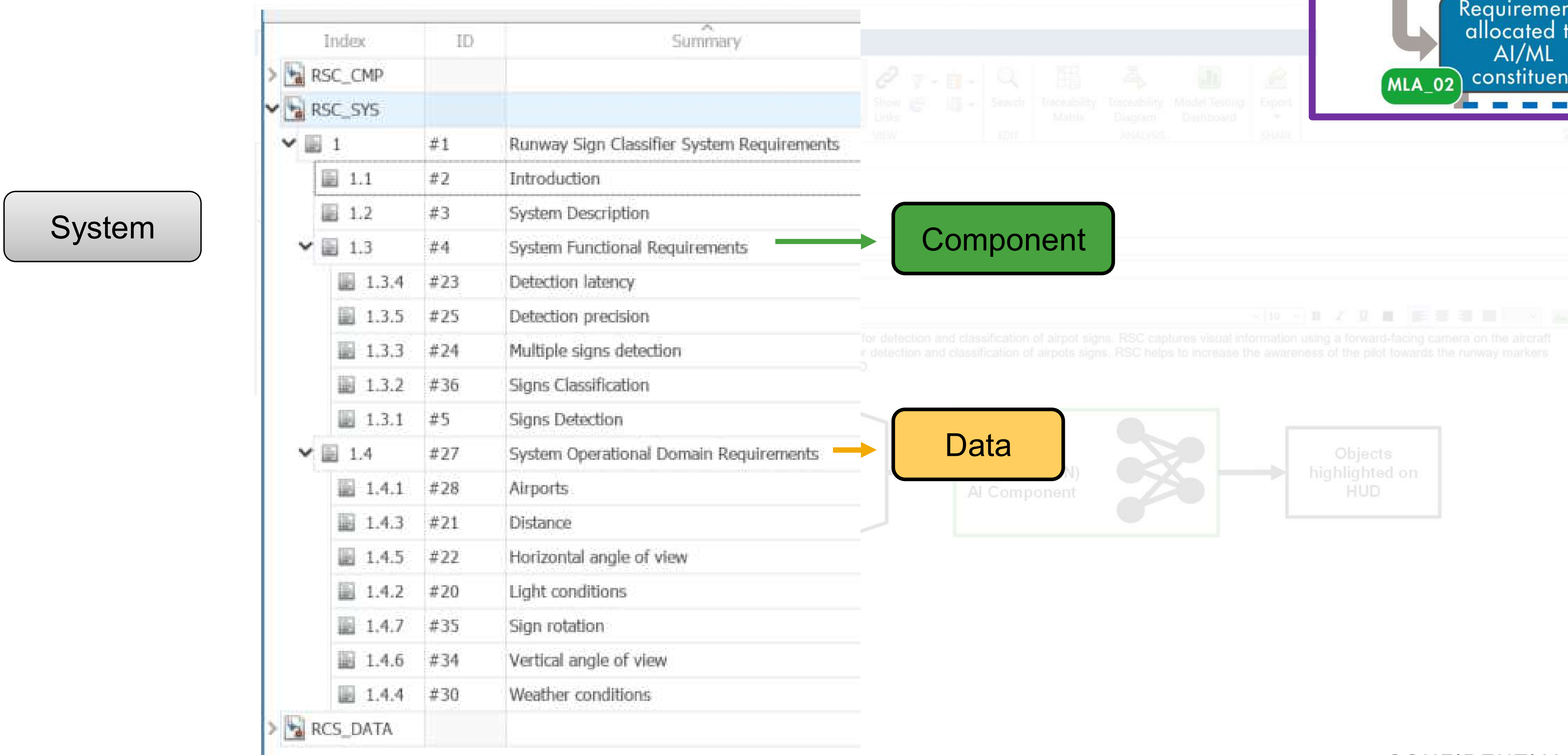


Case Study: A Visual Sign Recognition System



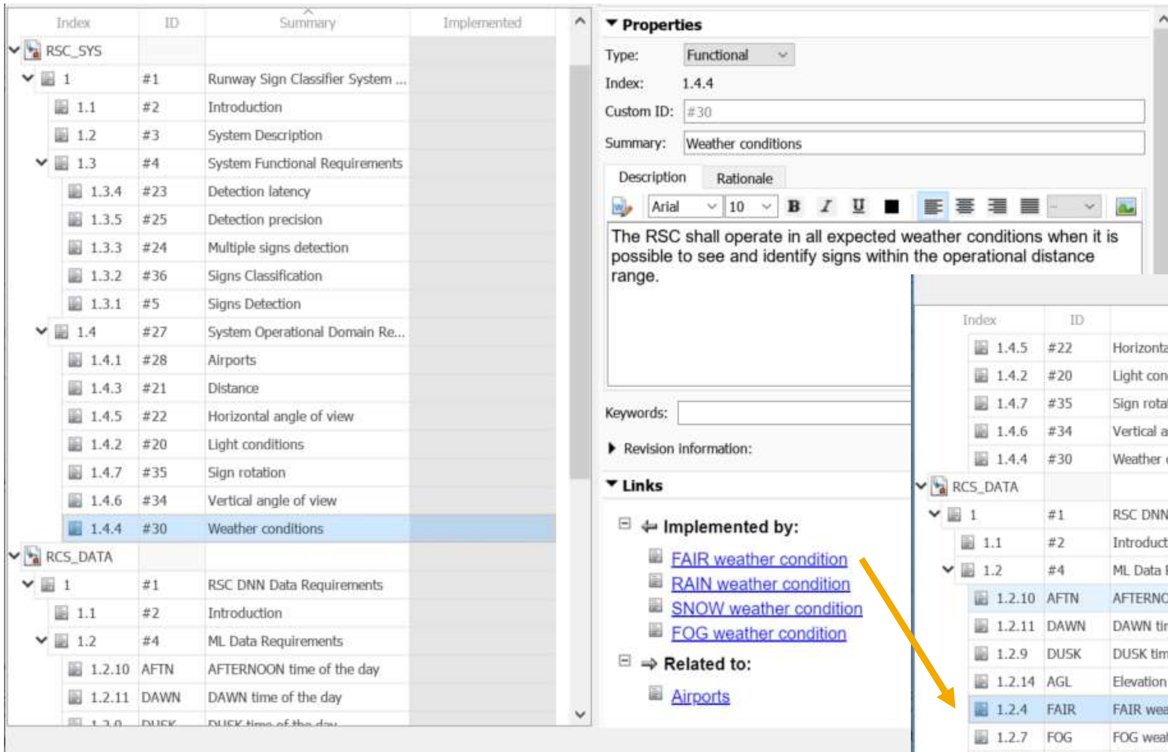
Defining system requirements and allocate to the AI constituent

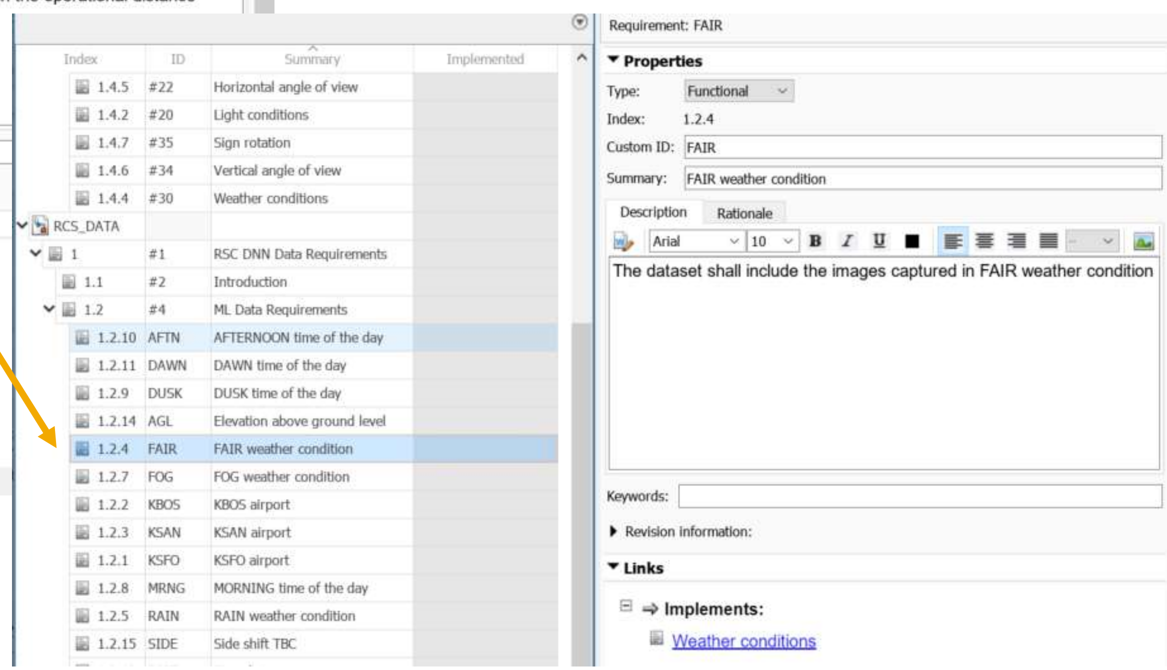


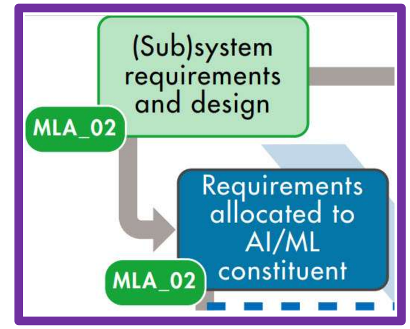


Index	ID	Summary
RSC_CMP		
RSC_SYS		
1	#1	Runway Sign Classifier System Requirements
1.1	#2	Introduction
1.2	#3	System Description
1.3	#4	System Functional Requirements
1.3.4	#23	Detection latency
1.3.5	#25	Detection precision
1.3.3	#24	Multiple signs detection
1.3.2	#36	Signs Classification
1.3.1	#5	Signs Detection
1.4	#27	System Operational Domain Requirements
1.4.1	#28	Airports
1.4.3	#21	Distance
1.4.5	#22	Horizontal angle of view
1.4.2	#20	Light conditions
1.4.7	#35	Sign rotation
1.4.6	#34	Vertical angle of view
1.4.4	#30	Weather conditions
RCS_DATA		

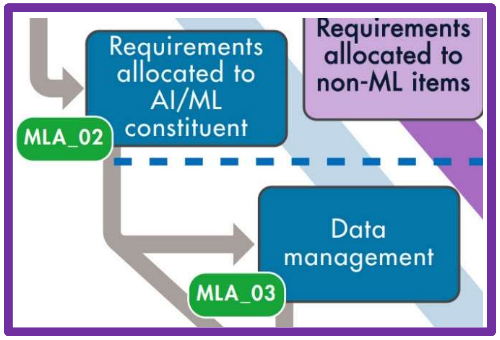
Link system requirements to data requirements





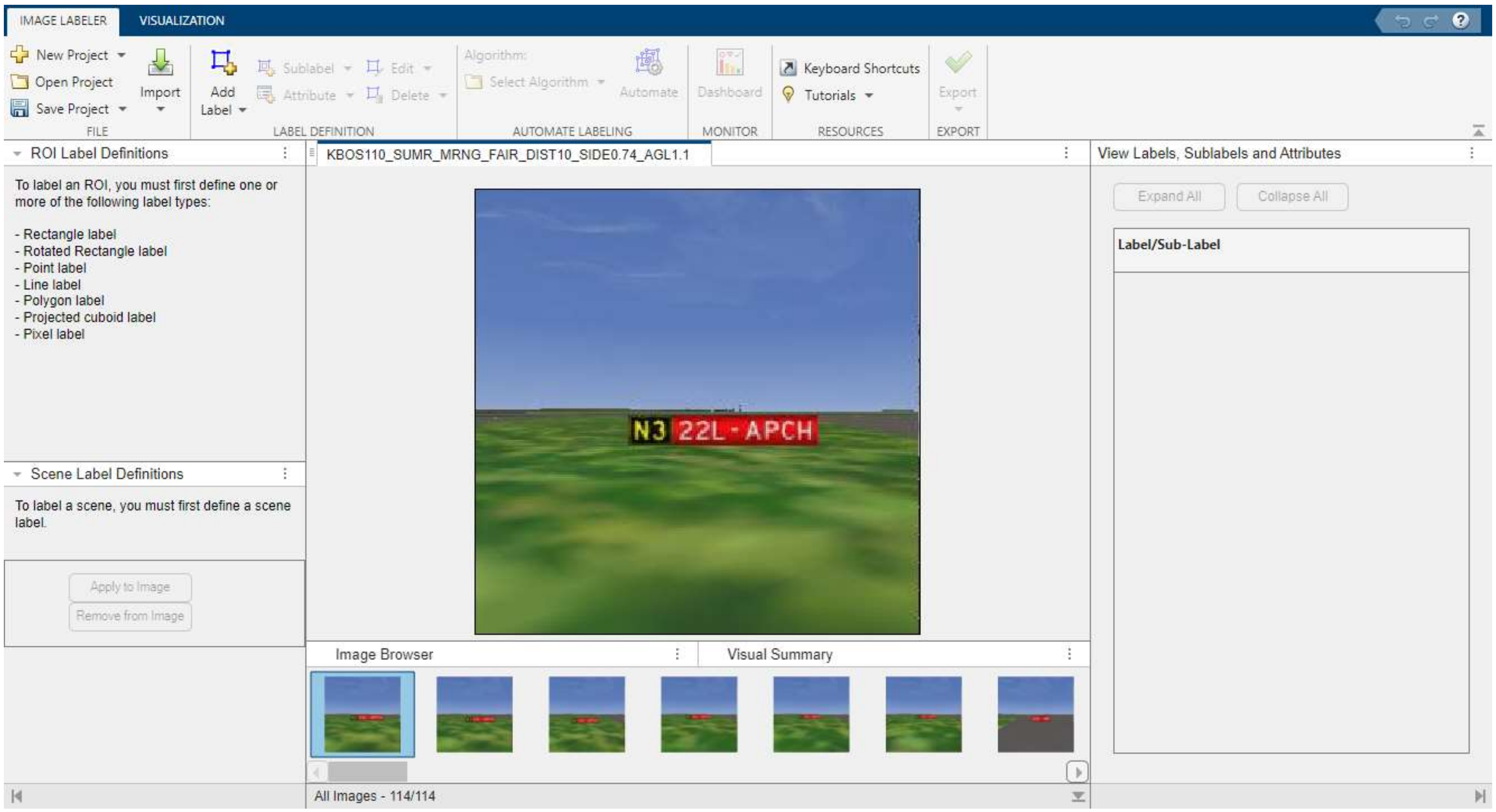


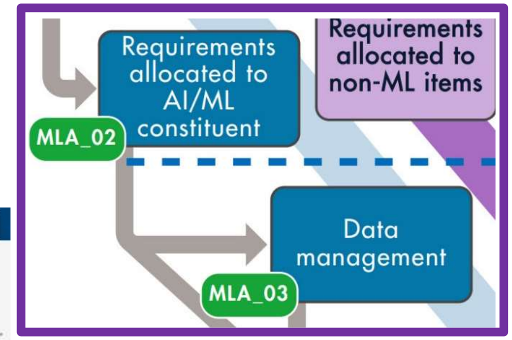
Map data requirements to the data



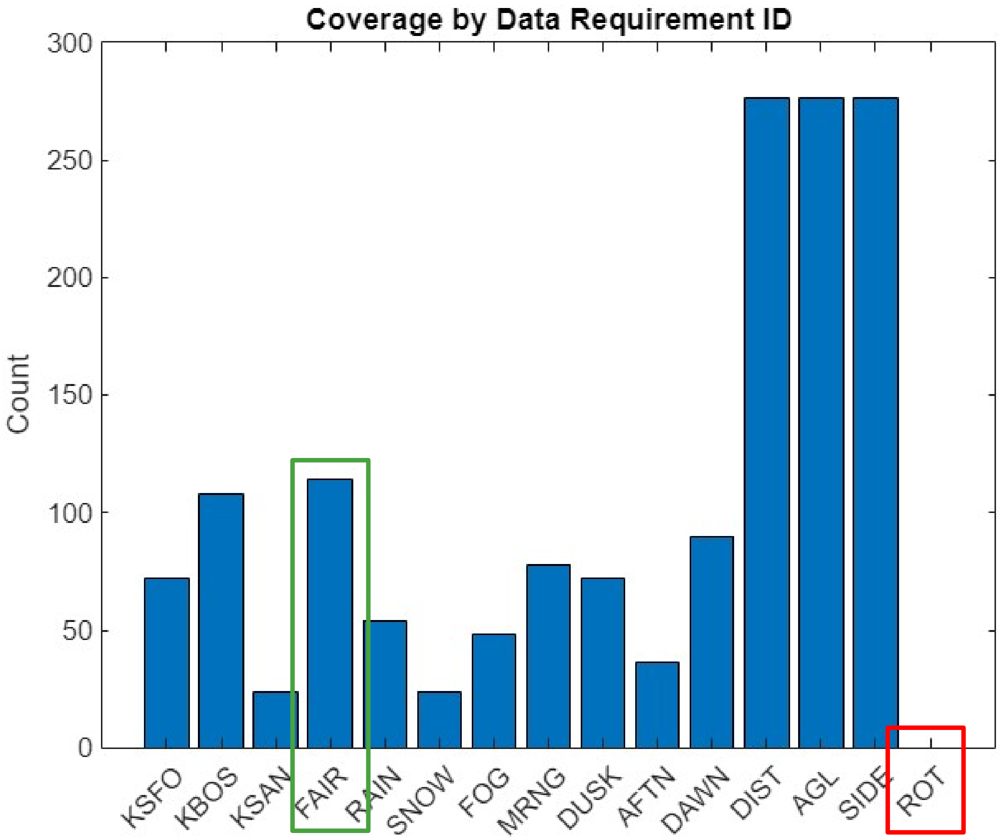
Data Requirement ID	Datastore	Datastore Size	Datastore Link
{'KSFO'}	{1×1 matlab.io.datastore.ImageDatastore}	72	{["Open Datastore"]}
{'KBOS'}	{1×1 matlab.io.datastore.ImageDatastore}	108	{["Open Datastore"]}
{'KSAN'}	{1×1 matlab.io.datastore.ImageDatastore}	24	{["Open Datastore"]}
{'FAIR'}	{1×1 matlab.io.datastore.ImageDatastore}	114	{["Open Datastore"]}
{'RAIN'}	{1×1 matlab.io.datastore.ImageDatastore}	54	{["Open Datastore"]}
{'SNOW'}	{1×1 matlab.io.datastore.ImageDatastore}	24	{["Open Datastore"]}
{'FOG' }	{1×1 matlab.io.datastore.ImageDatastore}	48	{["Open Datastore"]}
{'MRNG'}	{1×1 matlab.io.datastore.ImageDatastore}	78	{["Open Datastore"]}
{'DUSK'}	{1×1 matlab.io.datastore.ImageDatastore}	72	{["Open Datastore"]}
{'AFTN'}	{1×1 matlab.io.datastore.ImageDatastore}	36	{["Open Datastore"]}
{'DAWN'}	{1×1 matlab.io.datastore.ImageDatastore}	90	{["Open Datastore"]}
{'DIST'}	{1×1 matlab.io.datastore.ImageDatastore}	276	{["Open Datastore"]}
{'AGL' }	{1×1 matlab.io.datastore.ImageDatastore}	276	{["Open Datastore"]}
{'SIDE'}	{1×1 matlab.io.datastore.ImageDatastore}	276	{["Open Datastore"]}
{'ROT' }	{0×0 double }	0	{0×0 double }

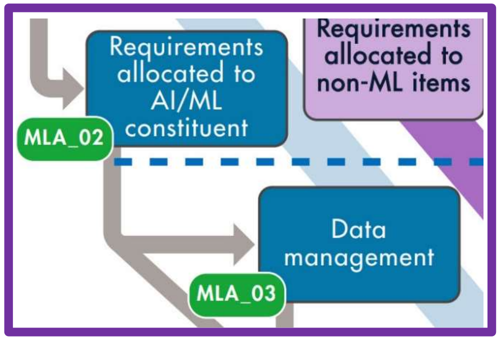
Review data Manually





Compute data coverage per data requirement





Examine data coverage in each of the operational conditions

Missing requirement on sign rotation

Thank you