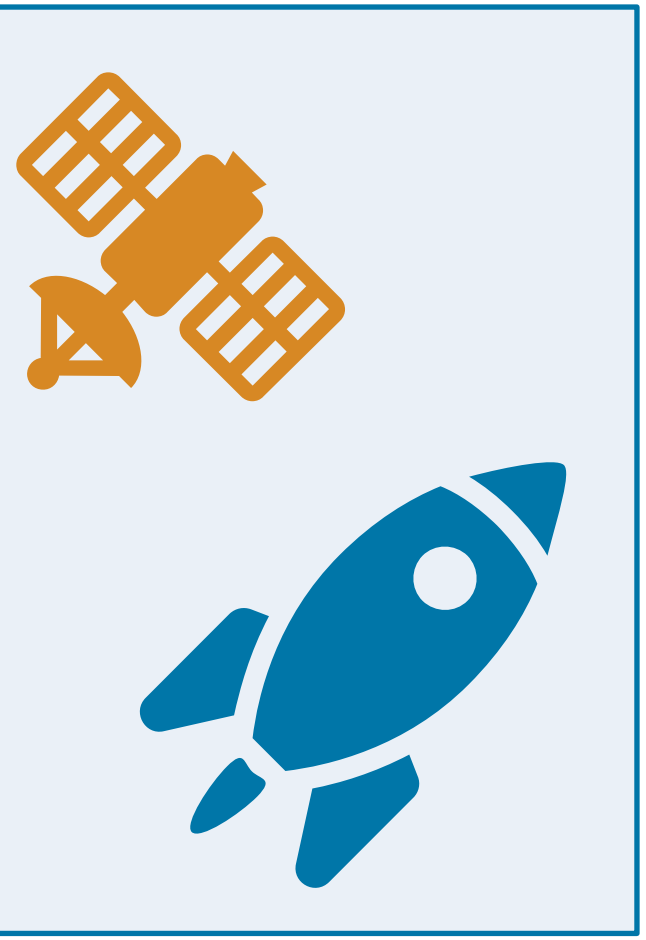
Gapless Verification with the ‘Digital Thread’ – State of the Art and Practical Challenges

María Fernández – *Application Engineer @ MathWorks*

Software Product Assurance Workshop 2023
27 – September – 2023

Common Challenges of Space Systems Development

Strict cost, weight and power budgets

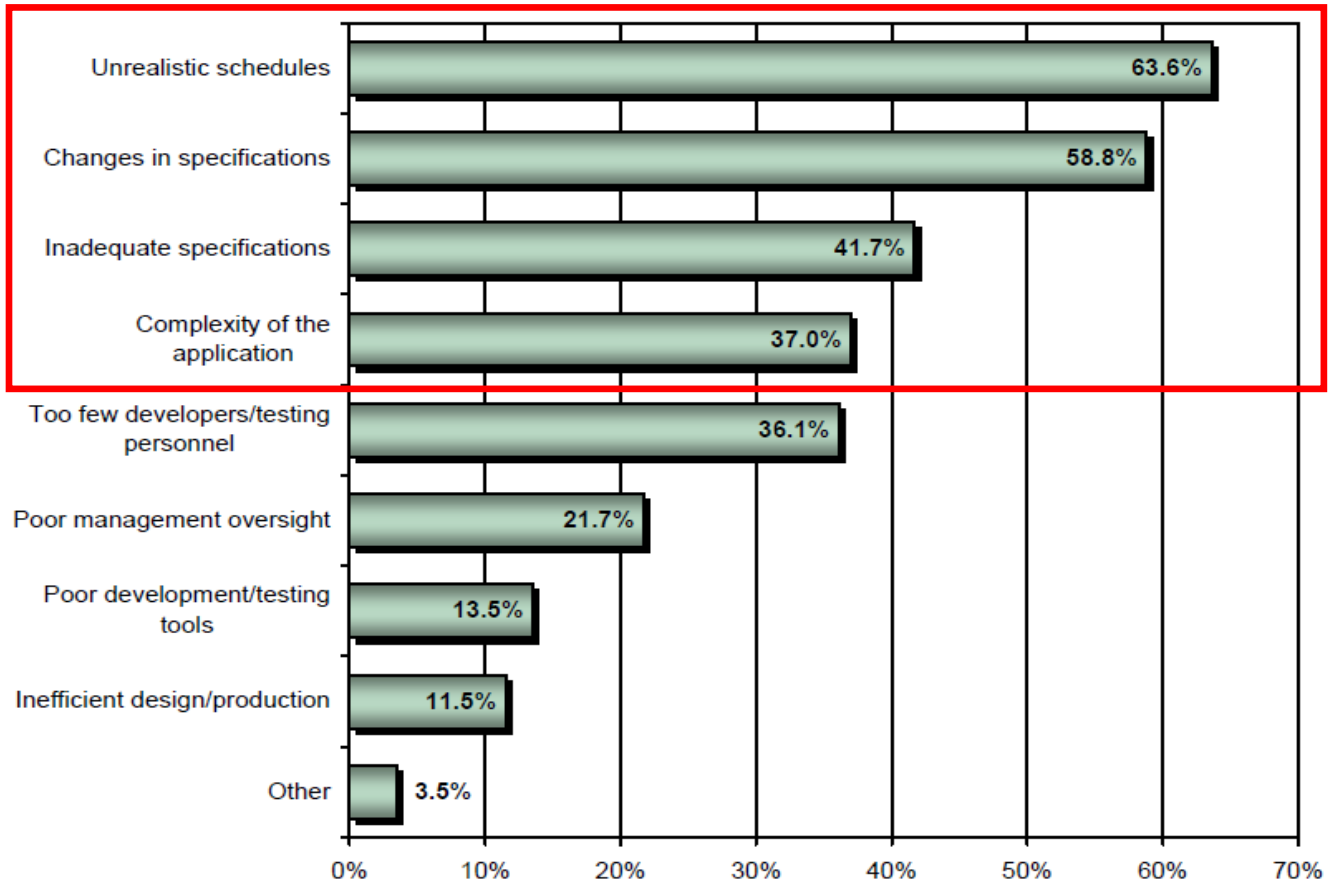


Strict deadlines

Large teams having to coordinate

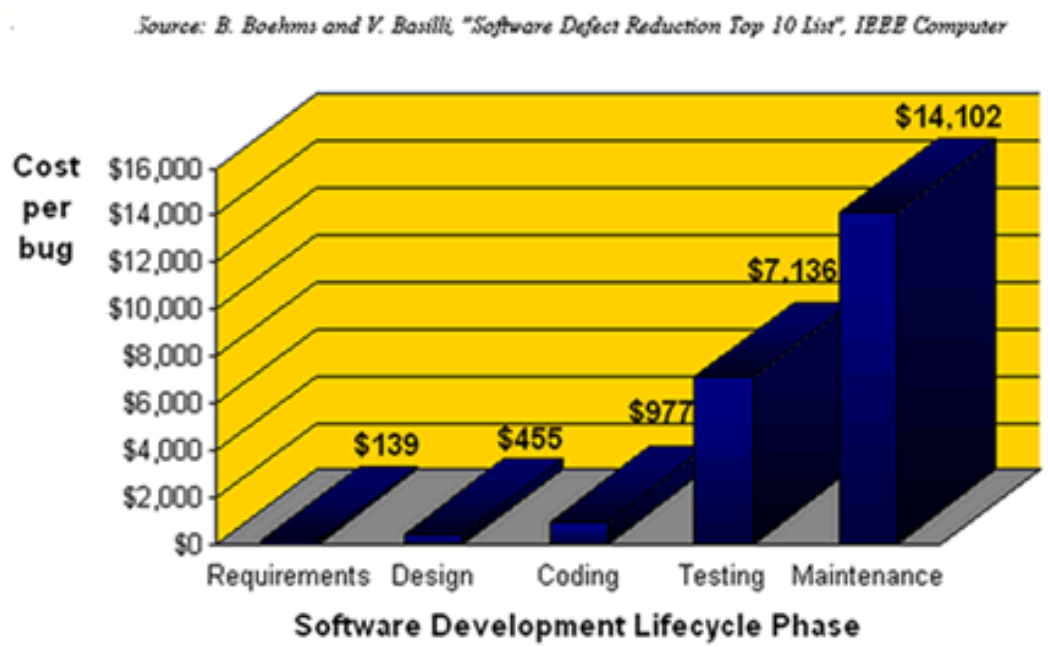
High-integrity software development & testing

Why did we miss our deadline?



Reasons for late projects, as reported by Venture Development Corporation.
Source: Embedded Software Strategic Market Intelligence report, Volume 4, December 2007, VDC.
Note: Percentages sum to over 100% due to multiple responses.

Minimize Costs by Detecting Errors Earlier



"…each delay in the detection and correction of a design problem makes it an order of magnitude more expensive to fix…"

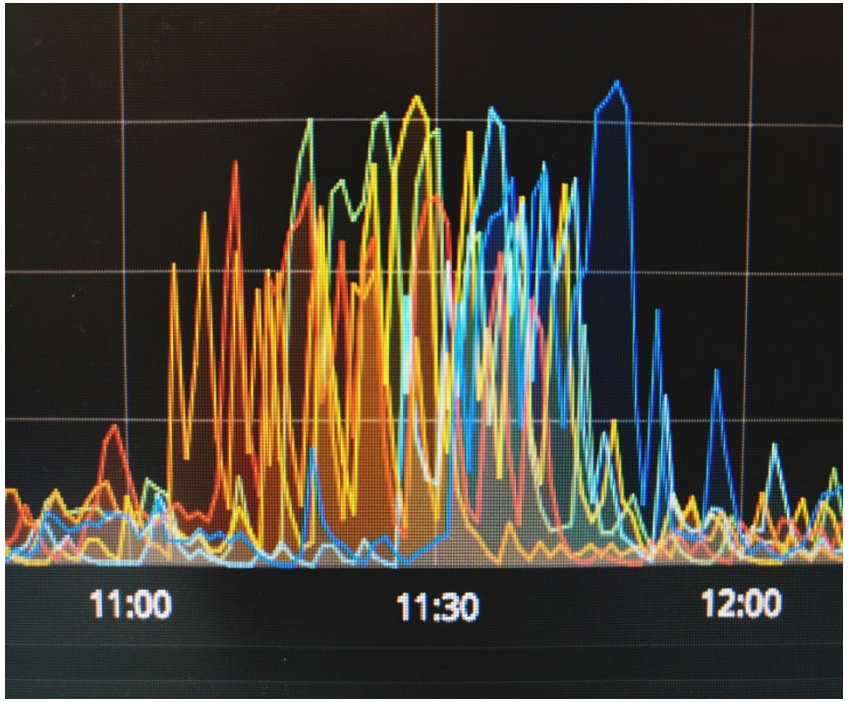
Clive Maxfield and Kuhoo Goyal
"EDA: Where Electronics Begins"
TechBites Interactive, October 1, 2001
ISBN: 0971406308]

Trending Satellite Technologies

- Broadband communications constellations with advanced phased array technology
- Autonomous Rendezvous and Proximity Operations including satellite servicing
- Optical Earth-imaging with very fast revisit times
- Radar sensing (Synthetic Aperture Radar)

Trend: Machine Learning / Deep Learning

- Telemetry outlier detection
  - Call attention to a potential spacecraft anomaly

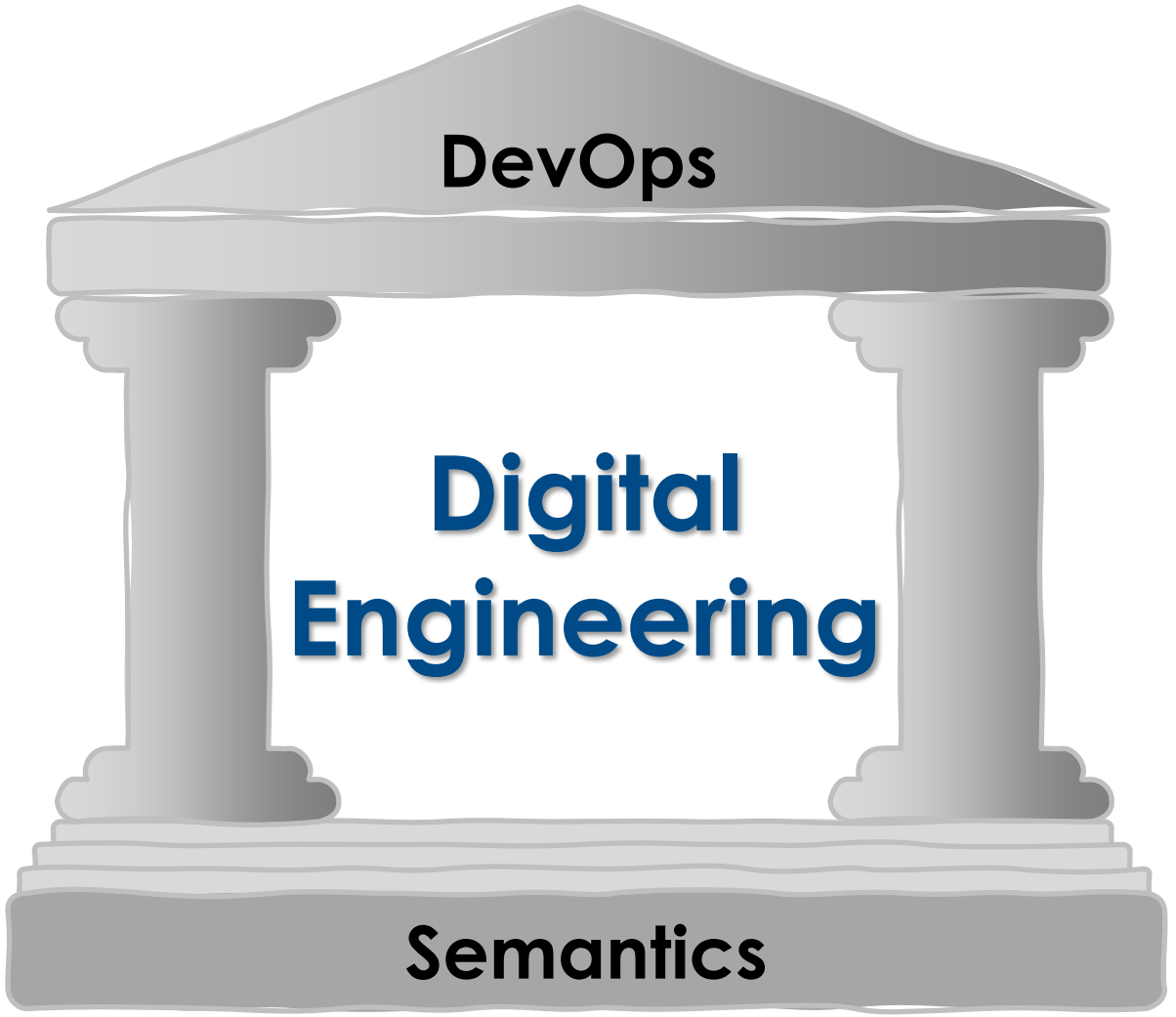


- Geospatial analytics
  - Derive business insights from satellite data

Digital Engineering Main Principles

Enabling Automation

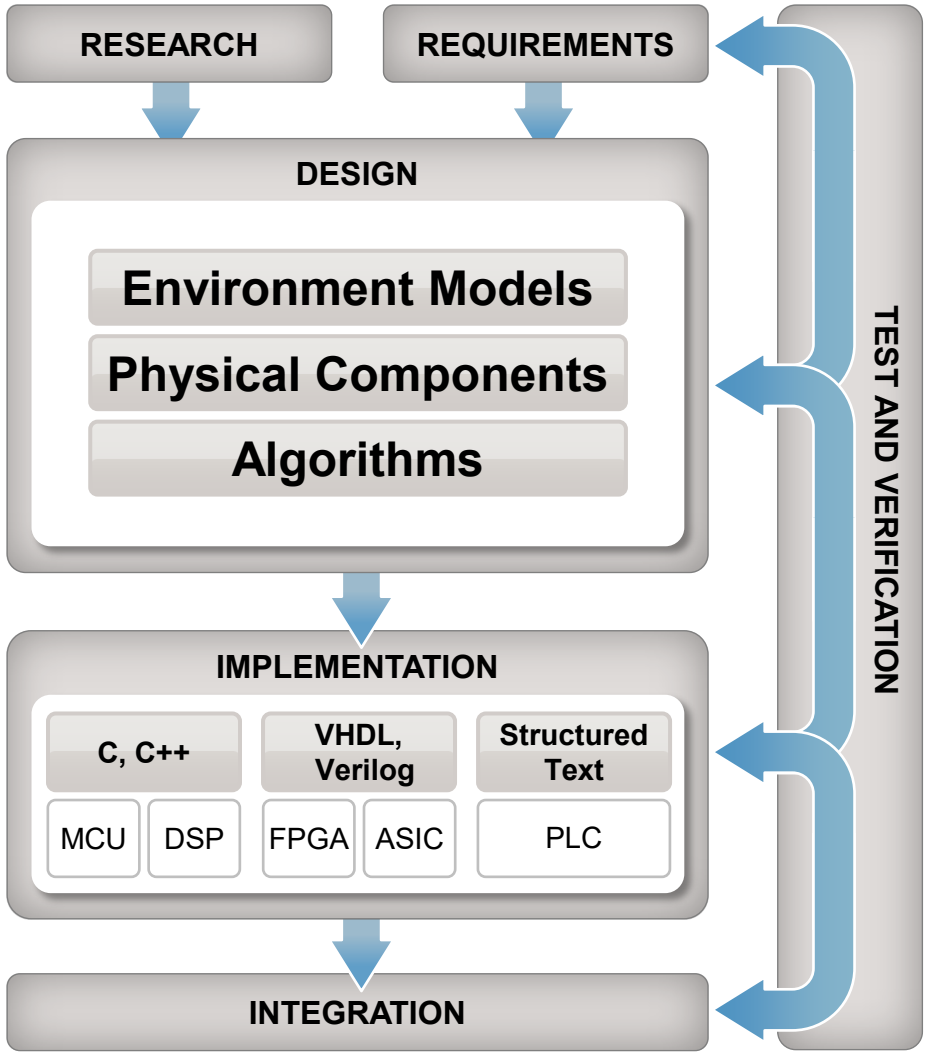
Reducing Human-in-the-loop effort



Maintaining Authoritative Source of Truth

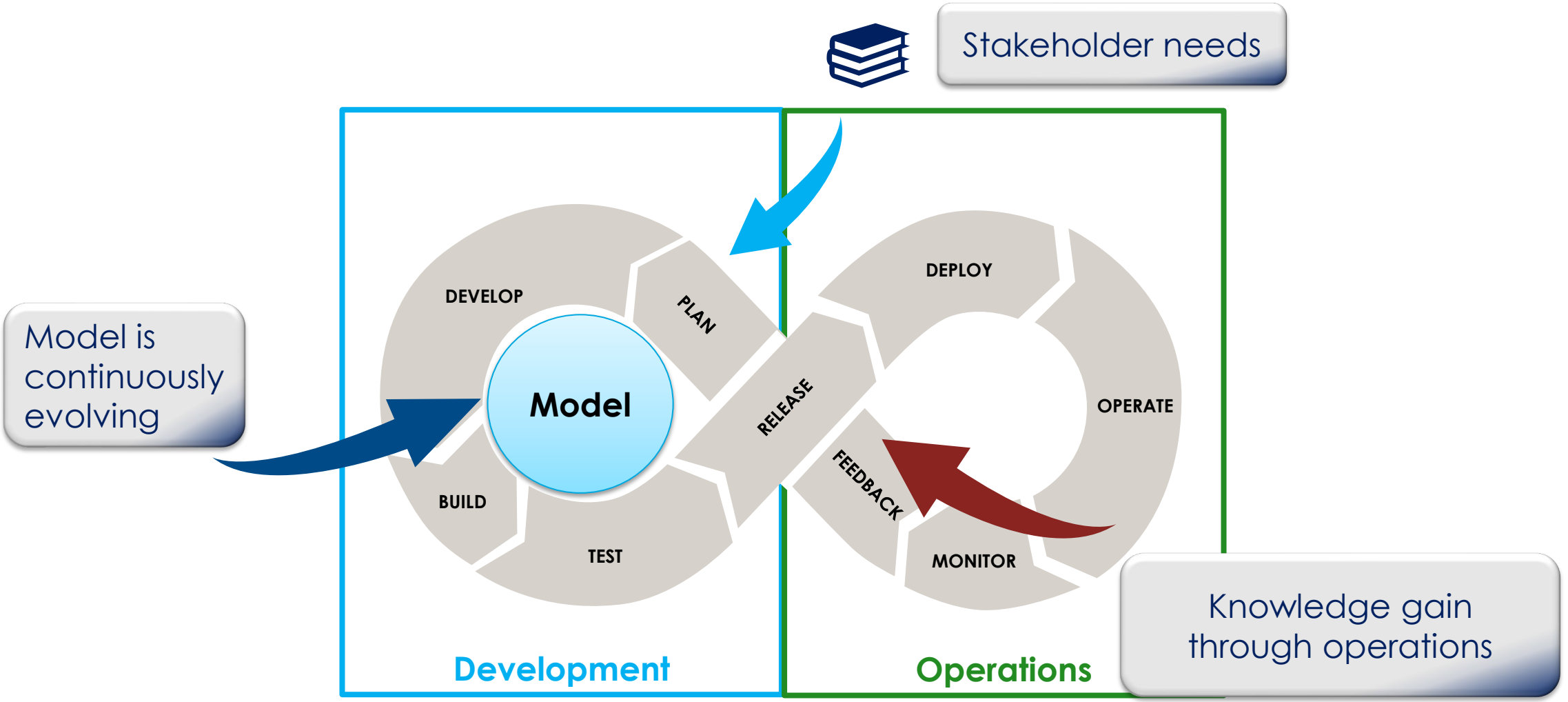
Model-based development

Model-Based Space Vehicle Design Process

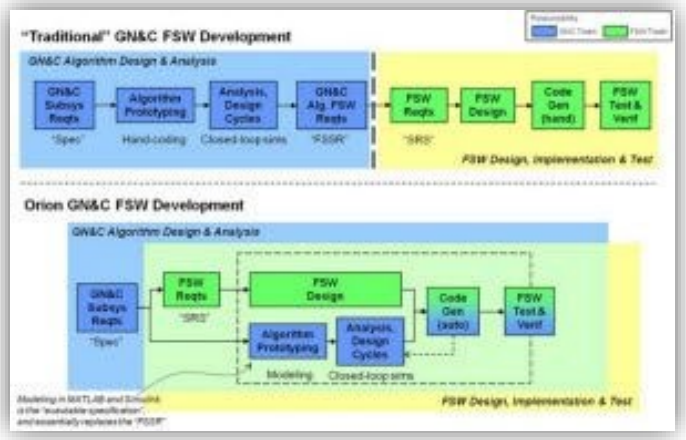


Executable model as the source of "truth"

DevOps – An Agile and Iterative Development Approach



Here some customer reference of Model-Based Design



[NASA - Orion GN&C: MATLAB and Simulink Modeling Standards]

ESA and Airbus Create Upper-Stage Attitude Control Development Framework Using Model-Based Design

Challenge

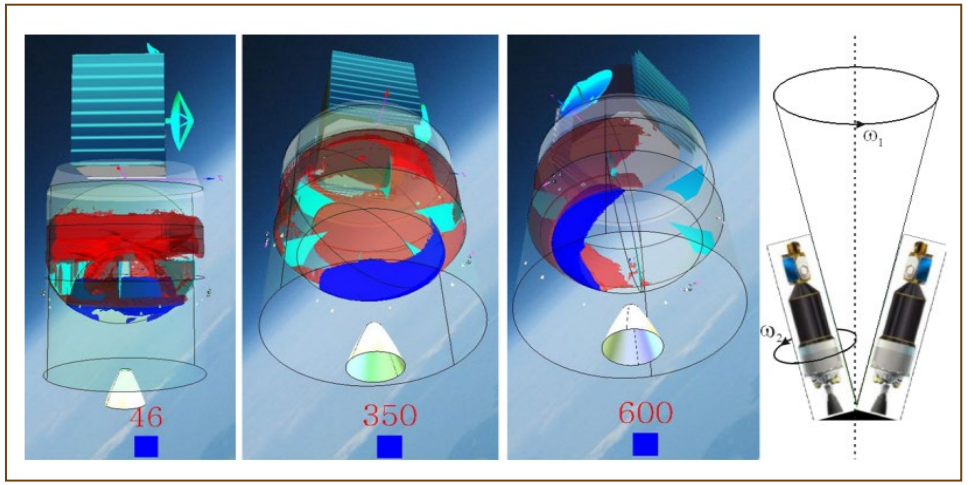
Speed the development of software for controlling complex launcher upper stage missions including the attitude of satellite payloads after they separate from ESA launch vehicles

Solution

Use Model-Based Design to develop controller models and multidomain physical models, run closed-loop simulations, and generate code for PIL testing

Results

- Design iterations reduced from one week to one day
- Failure modes modeled and eliminated
- Comprehensive design framework established



Propellant motion in spinning upper stages at 46, 350, and 600 seconds. Distribution after 350 seconds becomes uneven

"Model-Based Design multiplies the range of capabilities that I have as an engineer. As an individual control engineer I can do what previously took a handful of engineers, because I can create and simulate my own multidomain models. I don't have a wall around me anymore; I am able to better communicate and contribute across disciplines."

- Samir Bennani, ESA

Link to user story

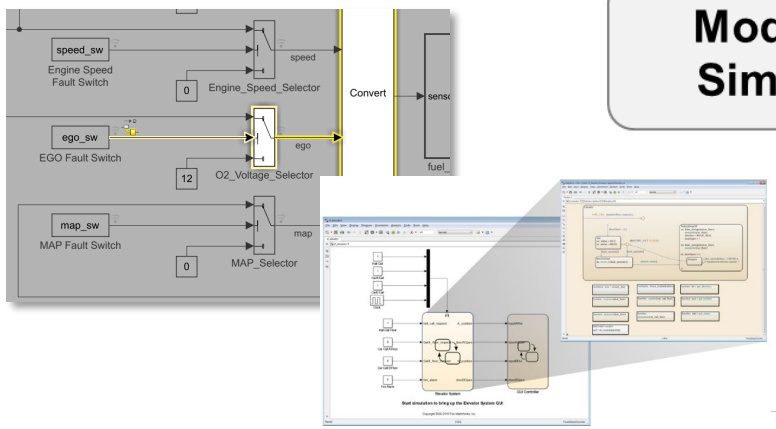
Model-Based Design: From Concept to Code

SIMULINK®

Model and Simulate Your System

Test Early and Often

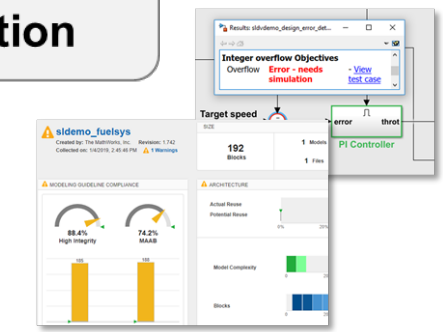
Automatically Generate Code

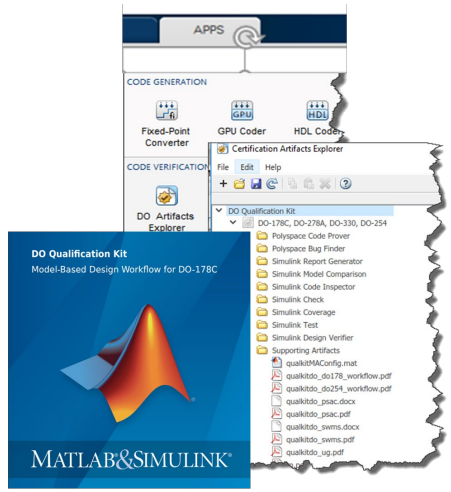


Modeling & Simulation

Test & Verification

Code Generation



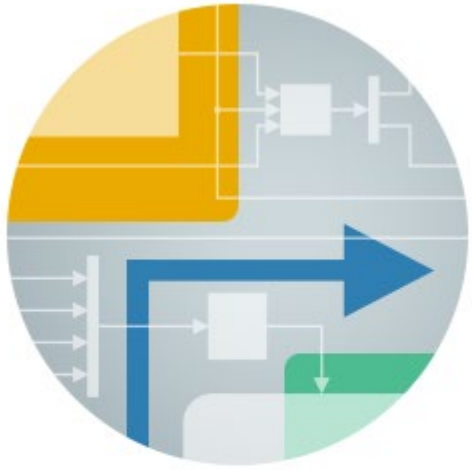


Meet Standards

MathWorks Products Overview

MATLAB

The Language of Technical Computing



SIMULINK

Simulation and Model-Based Design

MathWorks Products Overview

MATLAB

The Language of Technical Computing

SIMULINK

Simulation and Model-Based Design

Data Analytics

Wireless Communications

Deep Learning

Computer Vision

Signal Processing

Robotics

Control Systems

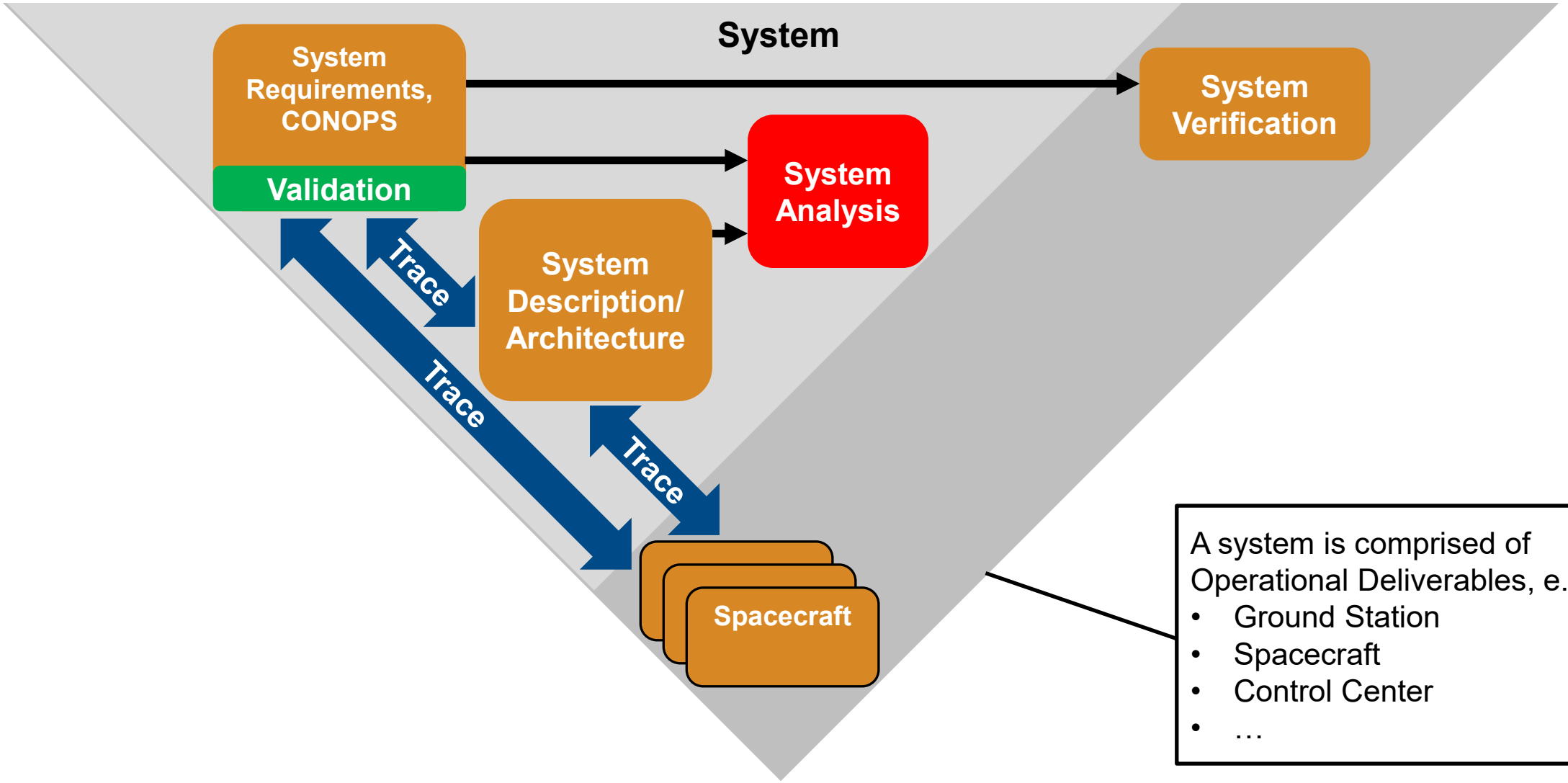
MathWorks Products Overview

MATLAB
The Language of Technical Computing

SIMULINK
Simulation and Model-Based Design

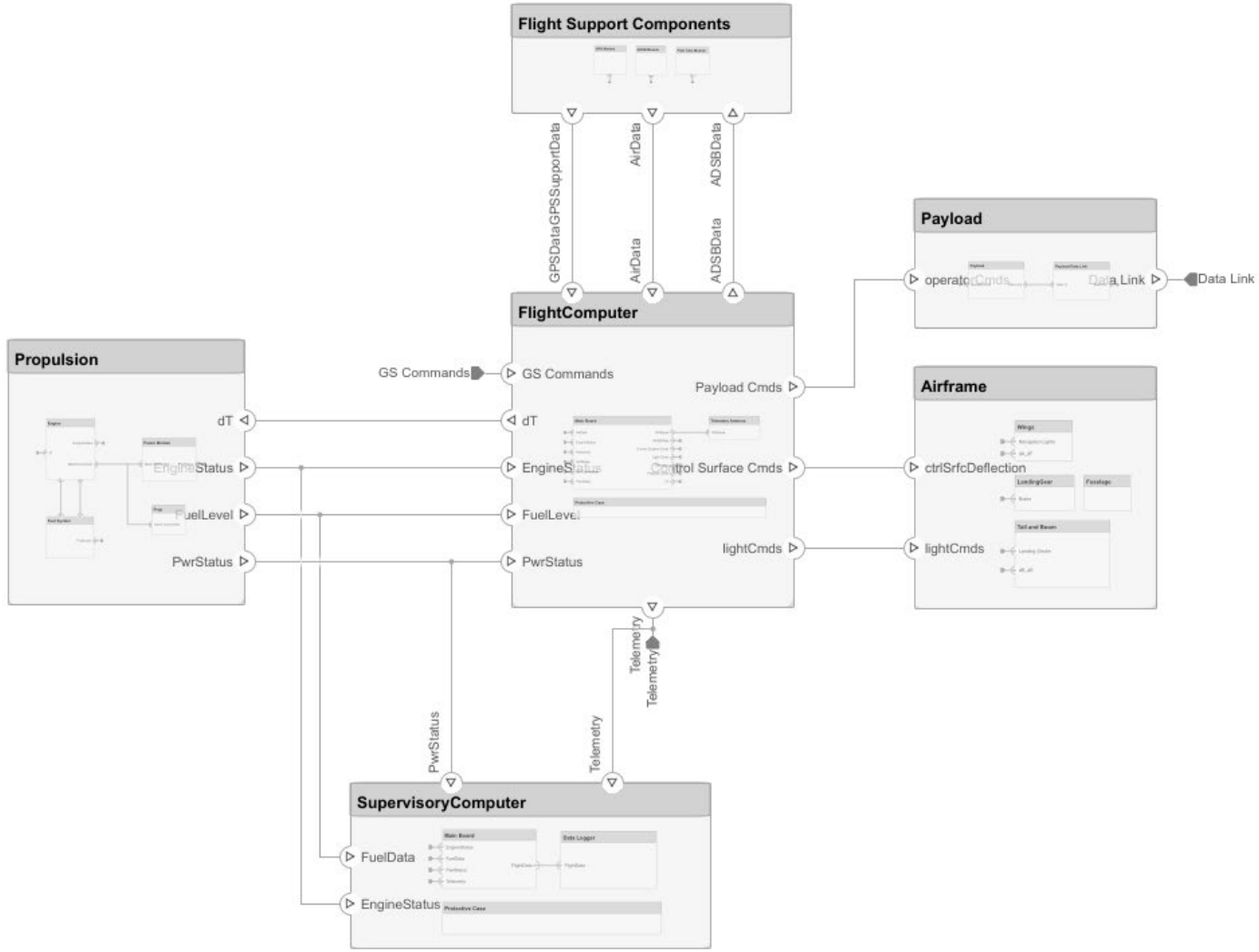
Event-Based Modeling	Physical Modeling	Control Systems	Signal Processing and Communications
Code Generation	Real-Time Simulation and Testing	Verification, Validation, and Test	Simulation Graphics and Reporting

Space System Engineering Workflow

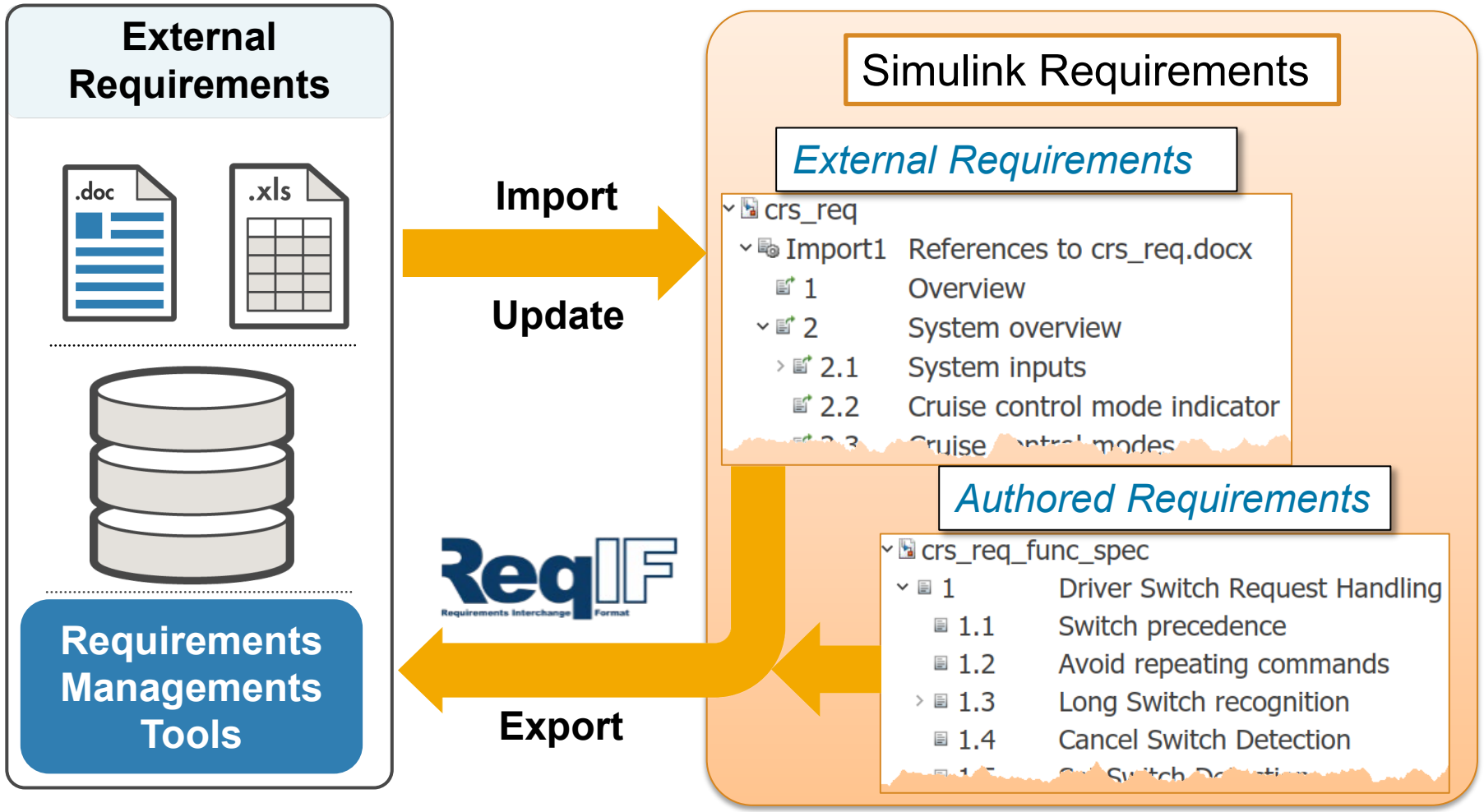


Space System Architecture Design

- Develop architectures
- Allocate requirements
- Perform trade studies and analyses

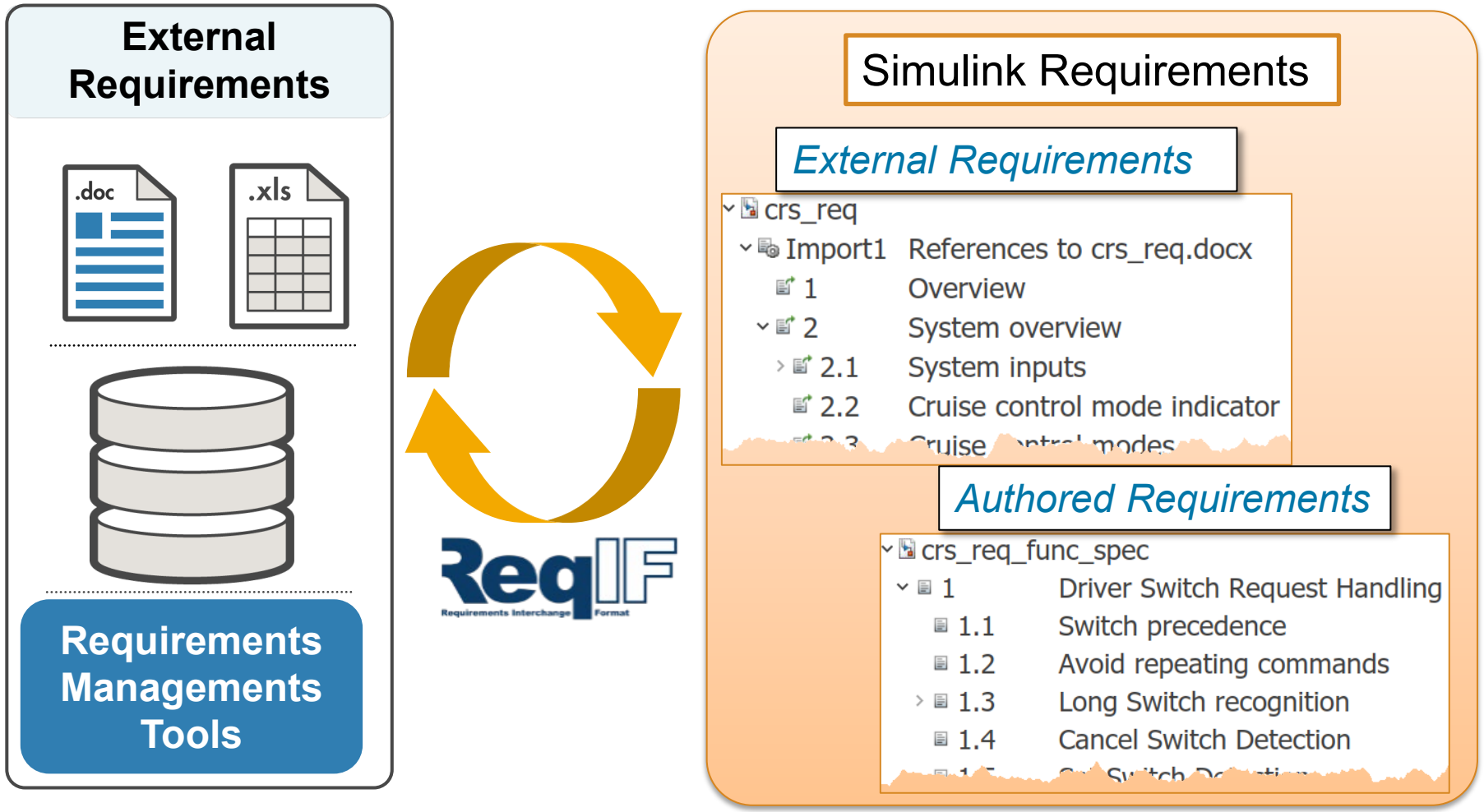


Requirements Management
Integrate with requirements tools and author requirements



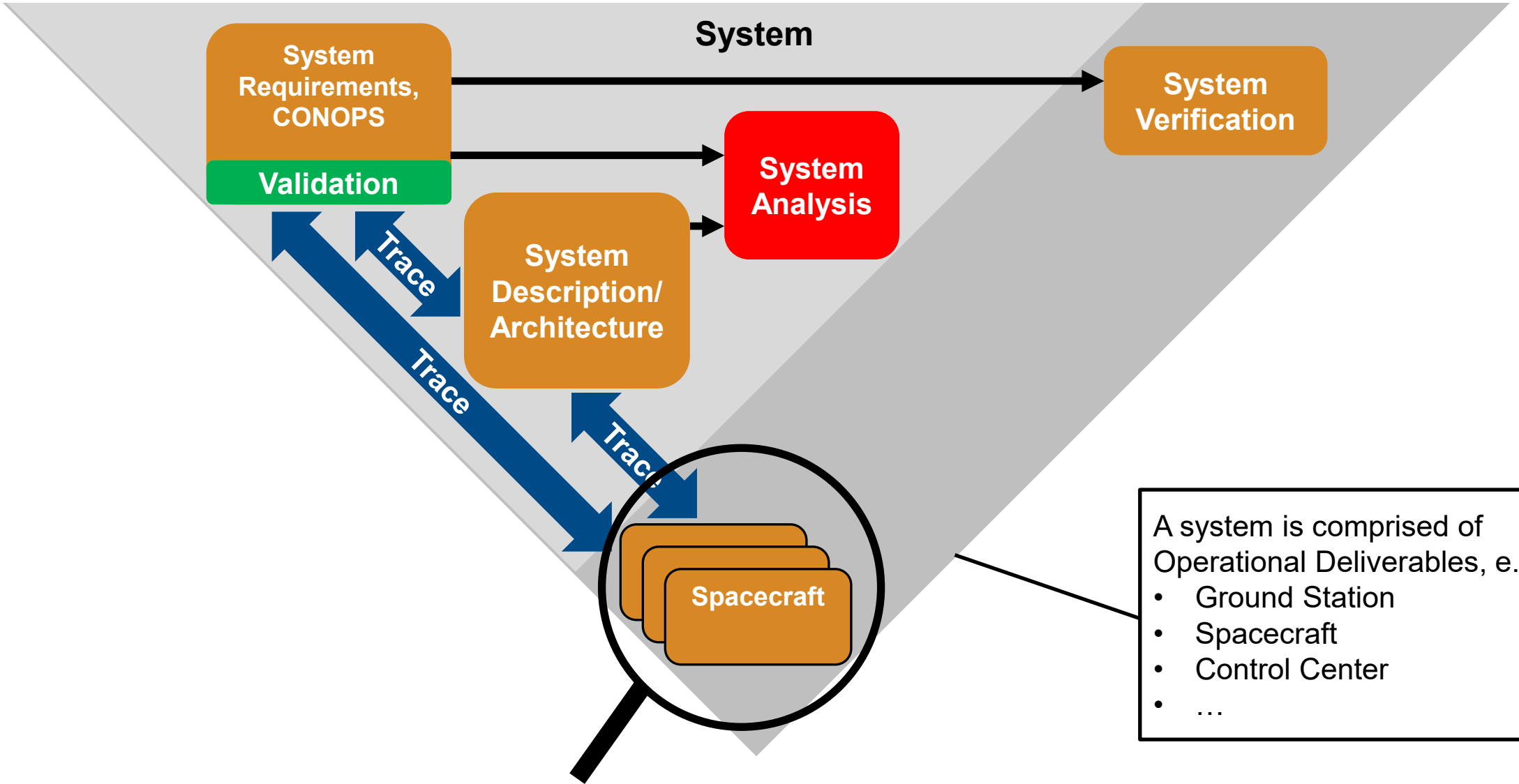
- Import from:
  - Word / Excel
  - IBM® Rational® DOORS®
  - ReqIF™ standard
- Update synchronizes changes from source
- Edit and add further details to import
- Author requirements
- Export ReqIF
  - Enables roundtrip with external tools

Requirements Management
Roundtrip workflow with external tools thru ReqIF

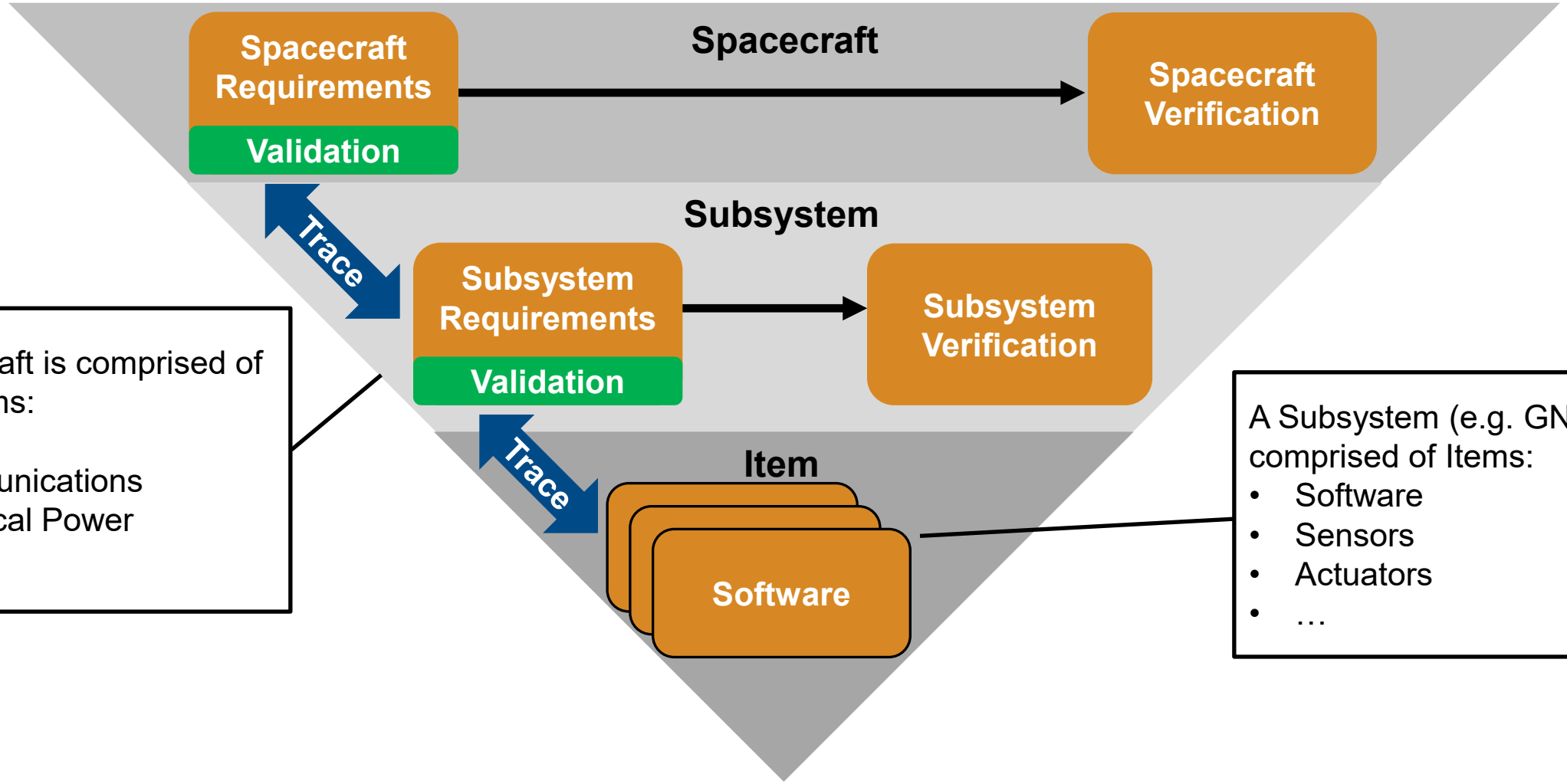


- Import from:
  - Word / Excel
  - IBM® Rational® DOORS®
  - ReqIF™ standard
- Update synchronizes changes from source
- Edit and add further details to import
- Author requirements
- Export ReqIF
  - Enables roundtrip with external tools

Space System Engineering Workflow



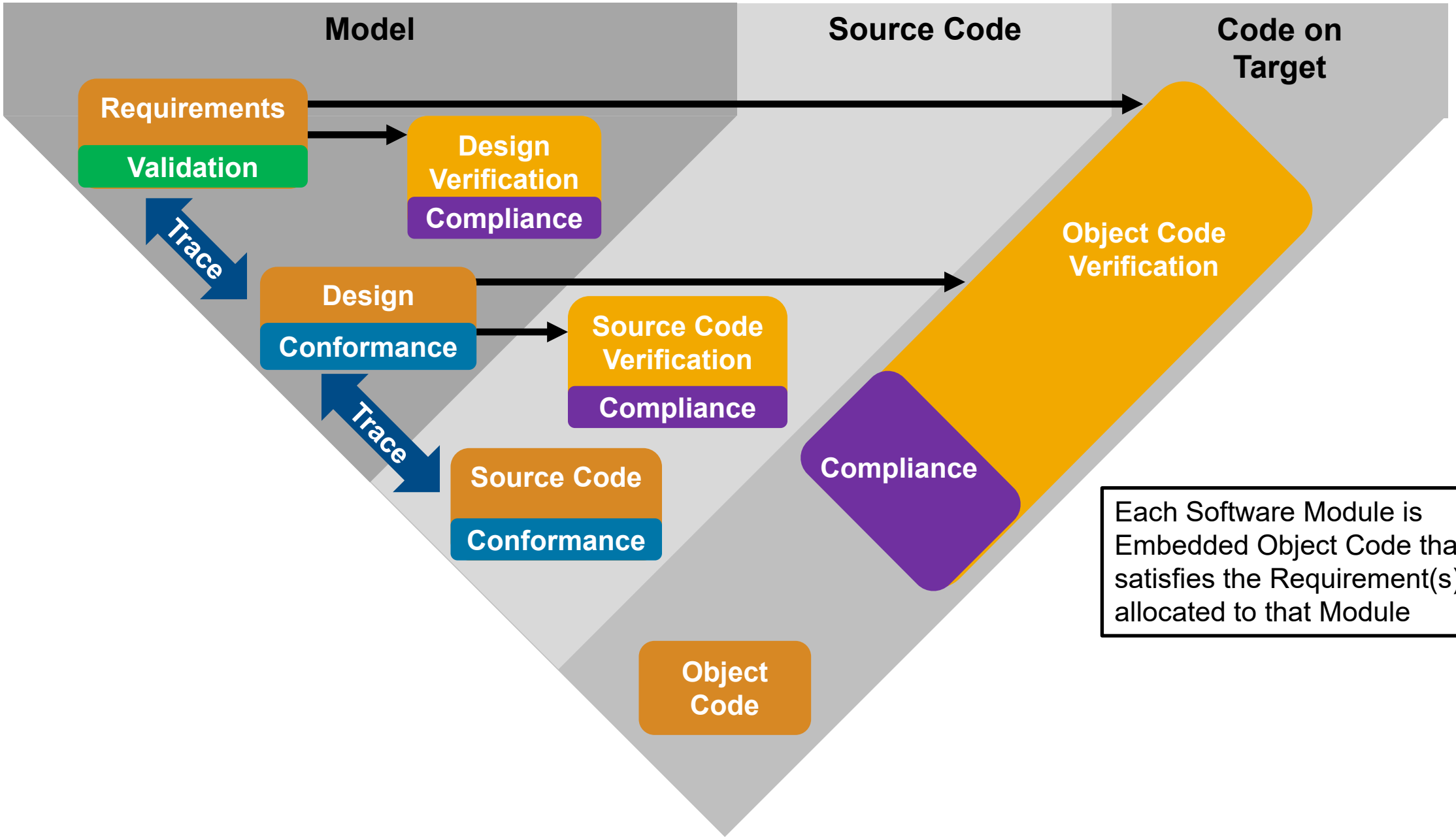
Spacecraft Subsystems Workflow



A spacecraft is comprised of subsystems:
- GNC
- Communications
- Electrical Power
- …

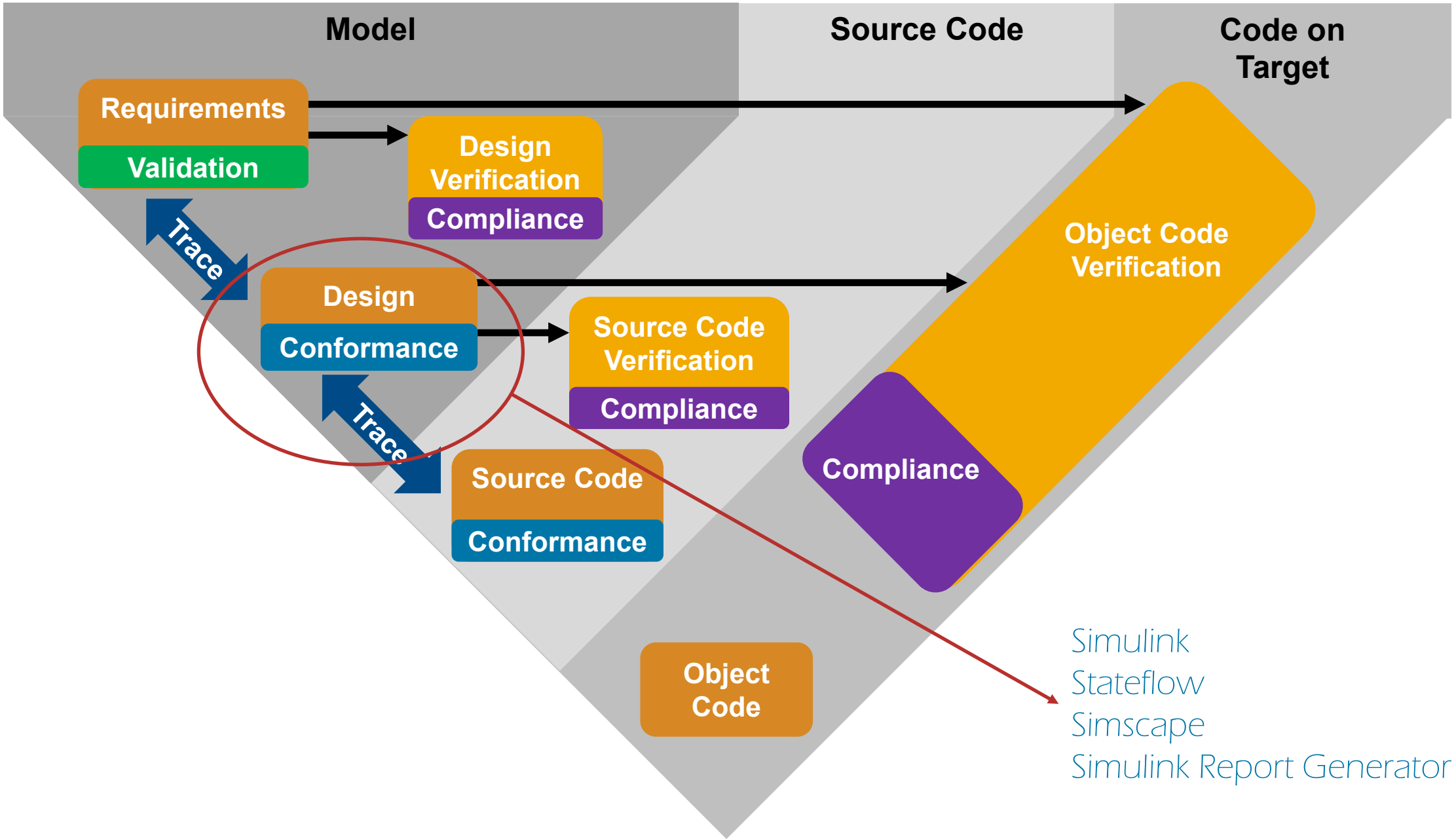
A Subsystem (e.g. GNC) is comprised of Items:
- Software
- Sensors
- Actuators
- …

Software Module Development Workflow

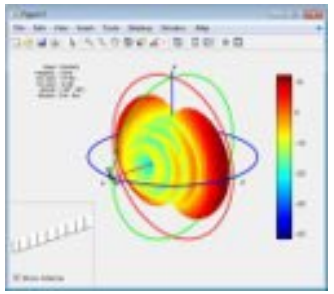


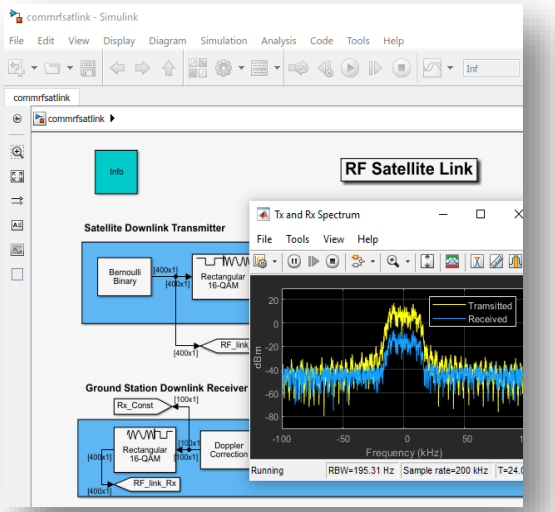
Each Software Module is Embedded Object Code that satisfies the Requirement(s) allocated to that Module

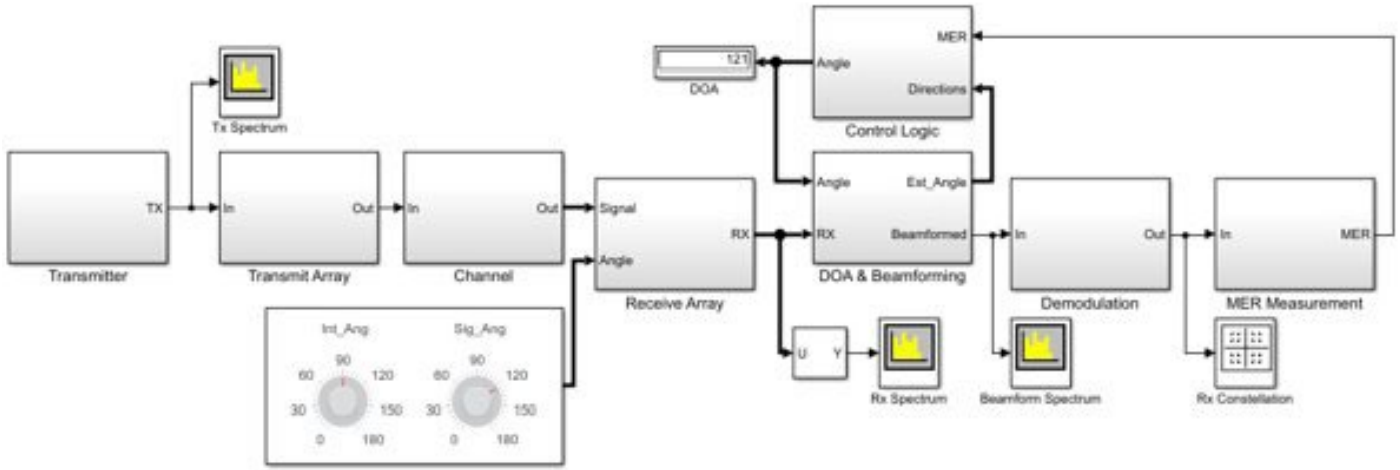
Software Module Development Workflow

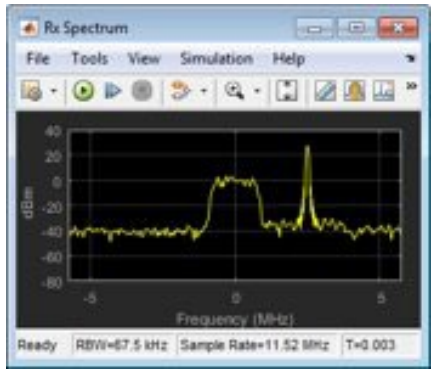


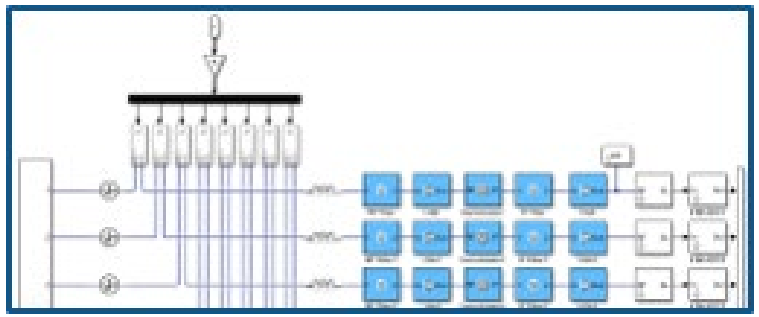
Communications system design in Simulink



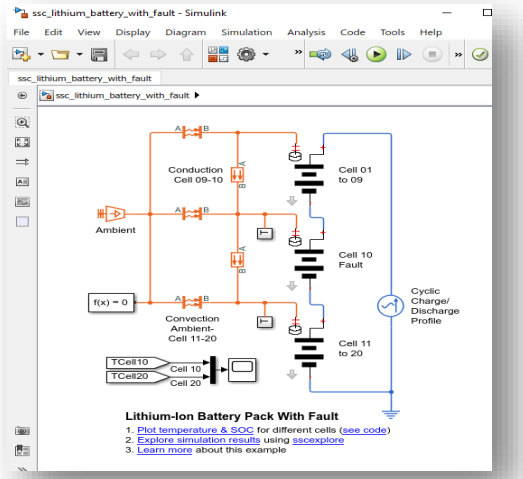


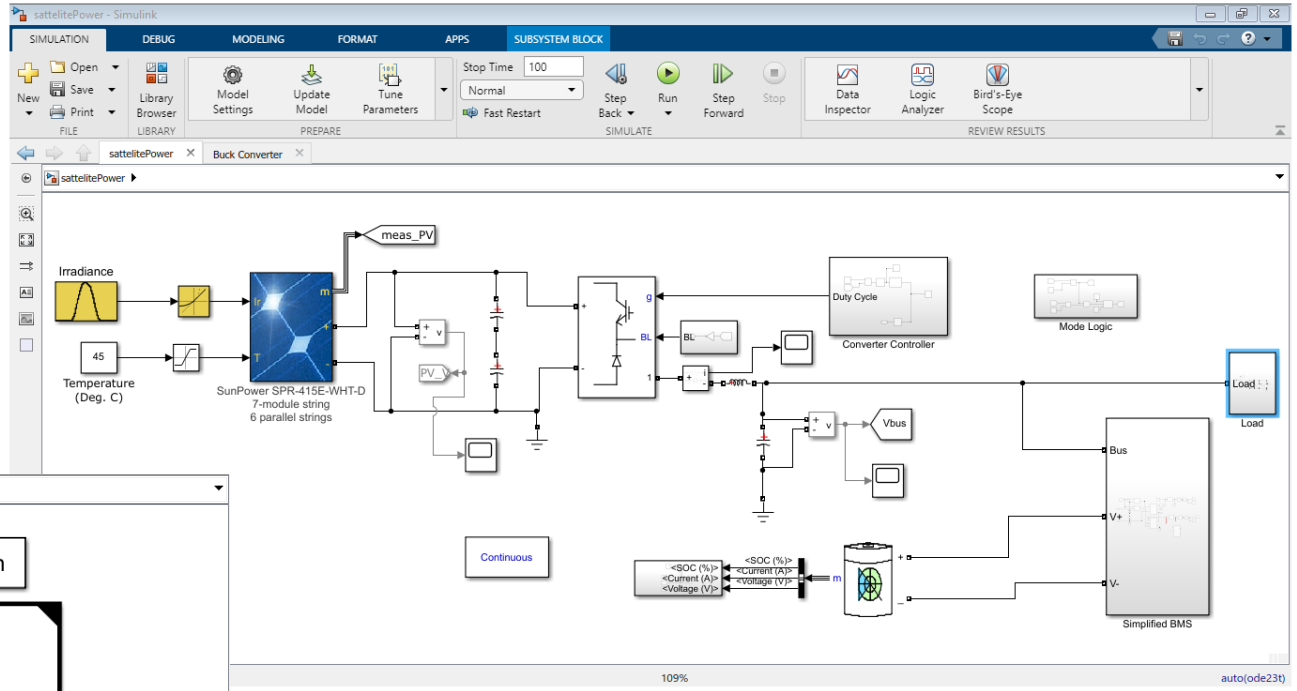


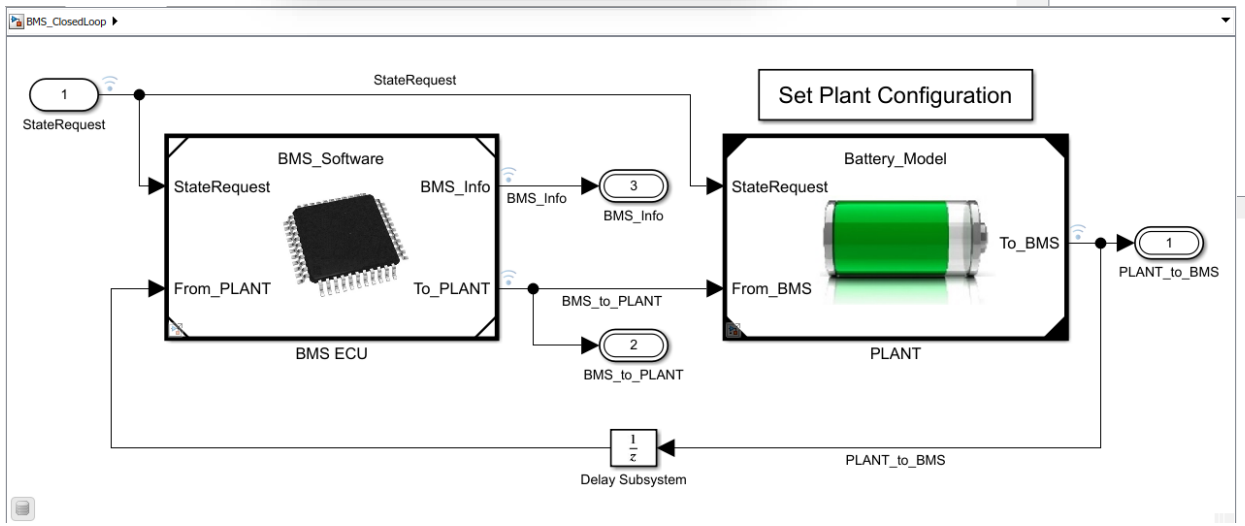


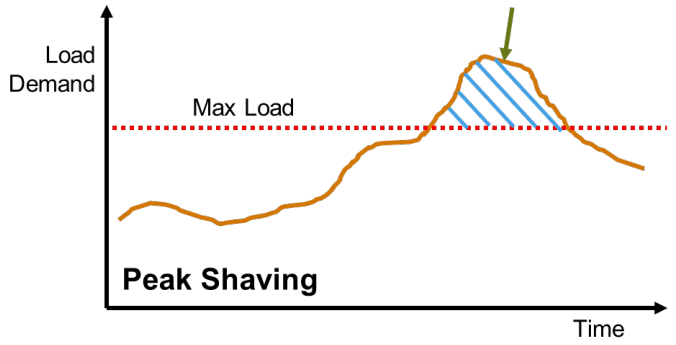


Power system design in Simulink

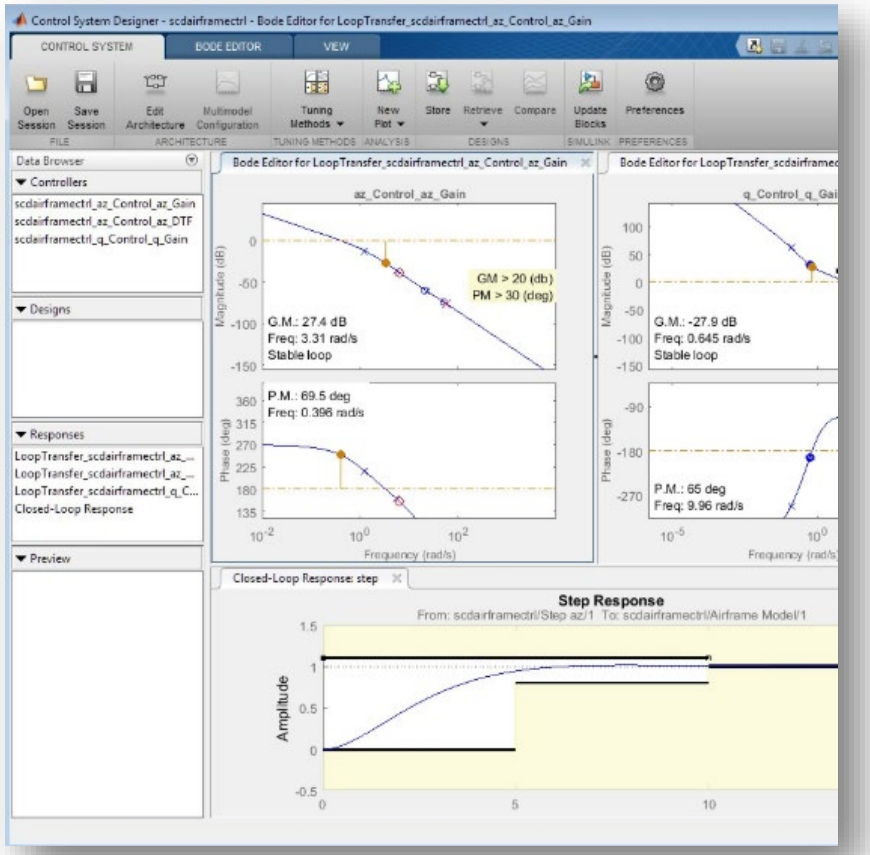




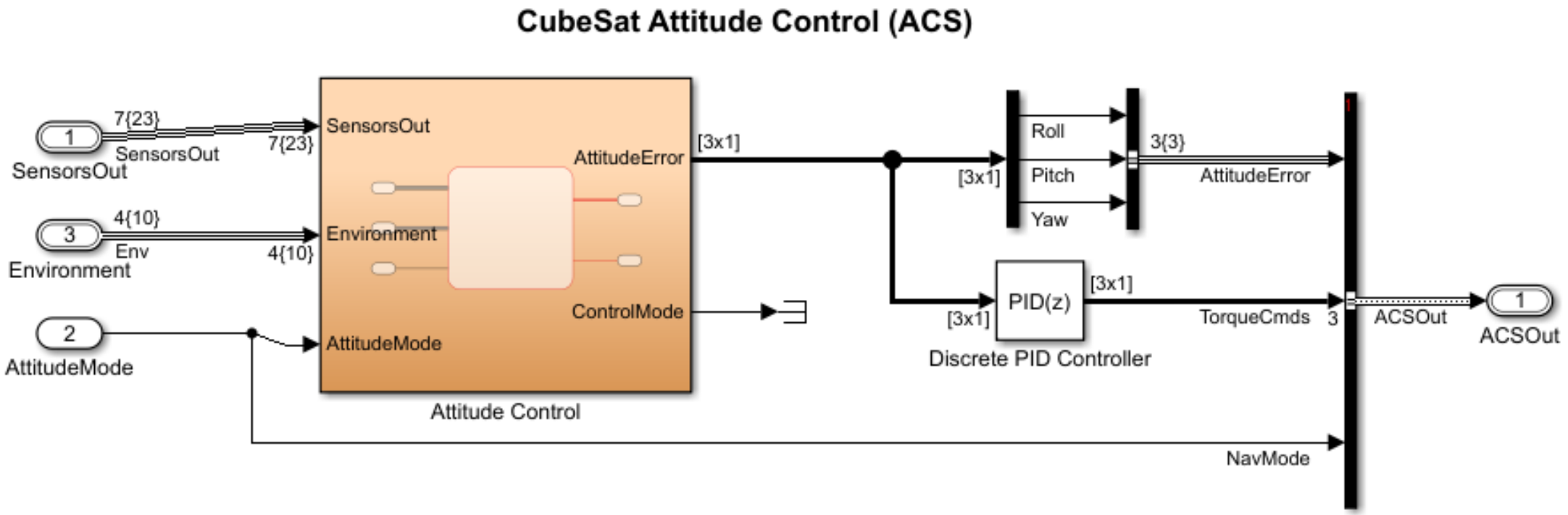




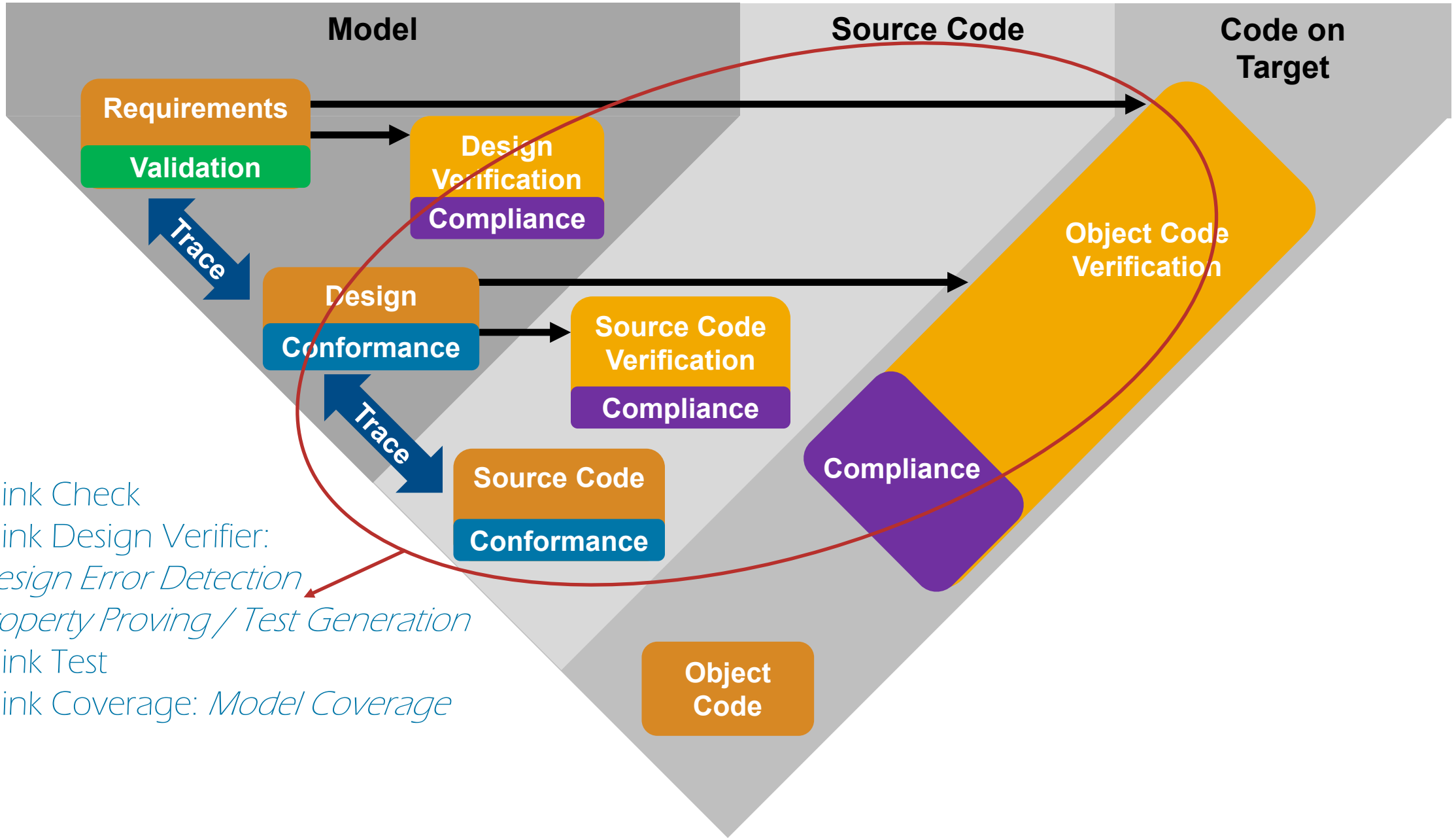
GNC Algorithms



Attitude control system design in Simulink



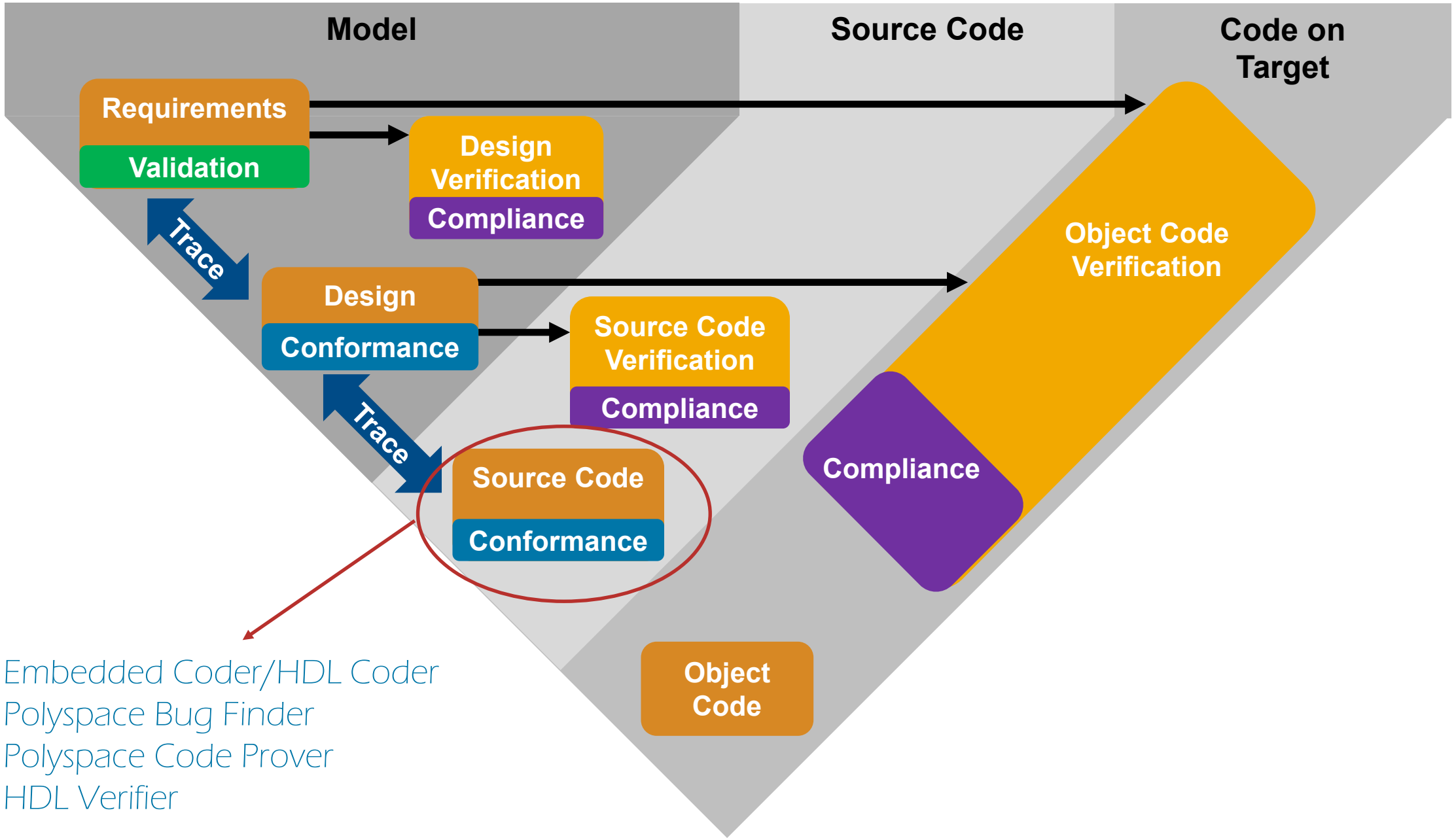
Software Module Development Workflow



Simulink Check
Simulink Design Verifier:
- *Design Error Detection*
- *Property Proving / Test Generation*

Simulink Test
Simulink Coverage: *Model Coverage*

Software Module Development Workflow



Embedded Coder/HDL Coder
Polyspace Bug Finder
Polyspace Code Prover
HDL Verifier

What's Next: From Design To Production with Model-Based Design

- Link requirements directly to designs
- Integrate testing with design
- Automatically generate standards compliant C/C++ code for embedded processors
- Automatically generate Verilog and VHDL code for FPGA programming or ASIC prototyping and design

Polyspace product family for C/C++

- Polyspace Code Prover
  - Proves code to be safe and dependable
  - Deep verification of software components
  - Perform QA signoff for production ready code

- Polyspace Bug Finder
  - Quickly find bugs in embedded software
  - Check code compliance for MISRA and JSF
  - Intended for every day use by software engineers

Ada language also supported for proving code

Some of the Global Software Standards

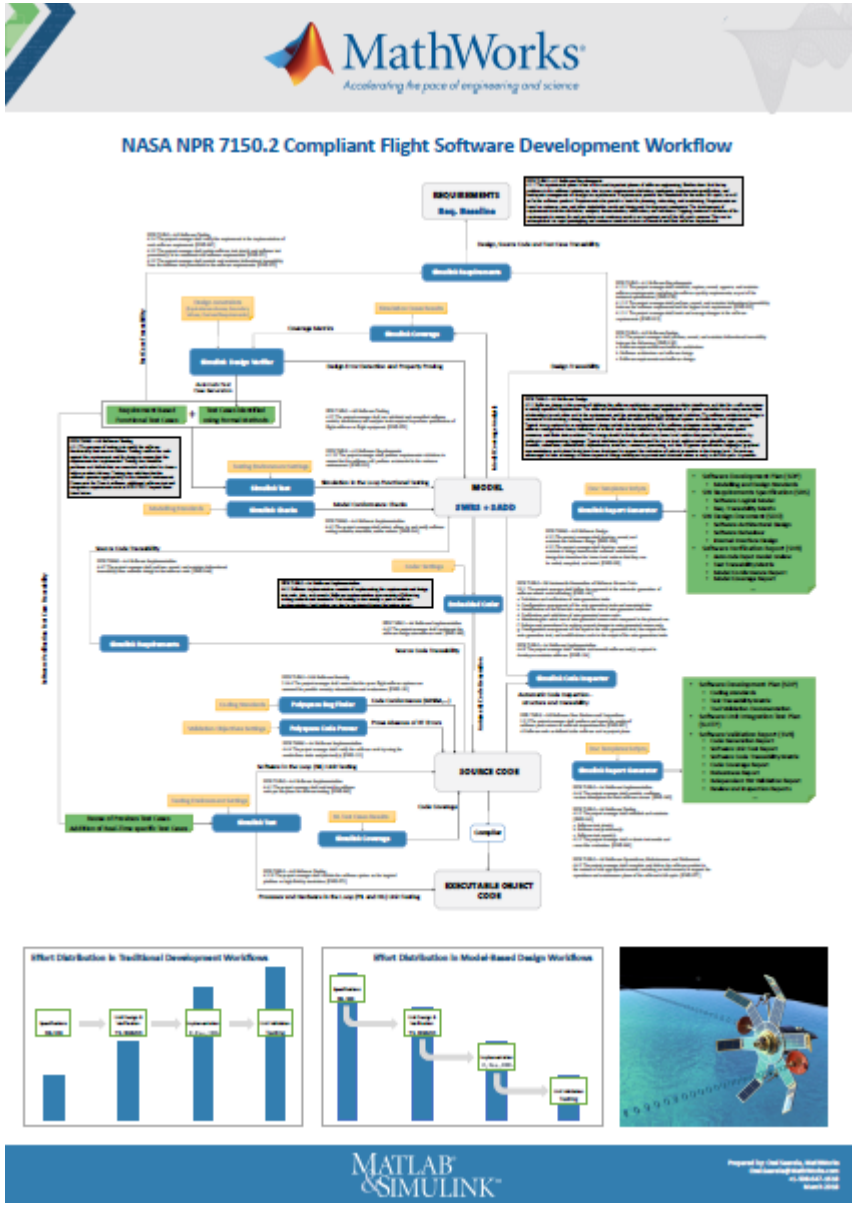
- There are many government/agency standards that can apply to space systems software development
  - NASA: NPR 7150.2 NASA Software Engineering Requirements
  - European Space Agency: ECSS-E-ST-40C – Software general requirements
  - *and others…*
- Companies also have internal standards
  - These standards can be applied to internal projects, or commercial customer programs that do not require a government standard

NASA Software Engineering Standard

- MATLAB and Simulink High Integrity Development Workflow mapped to NASA standard

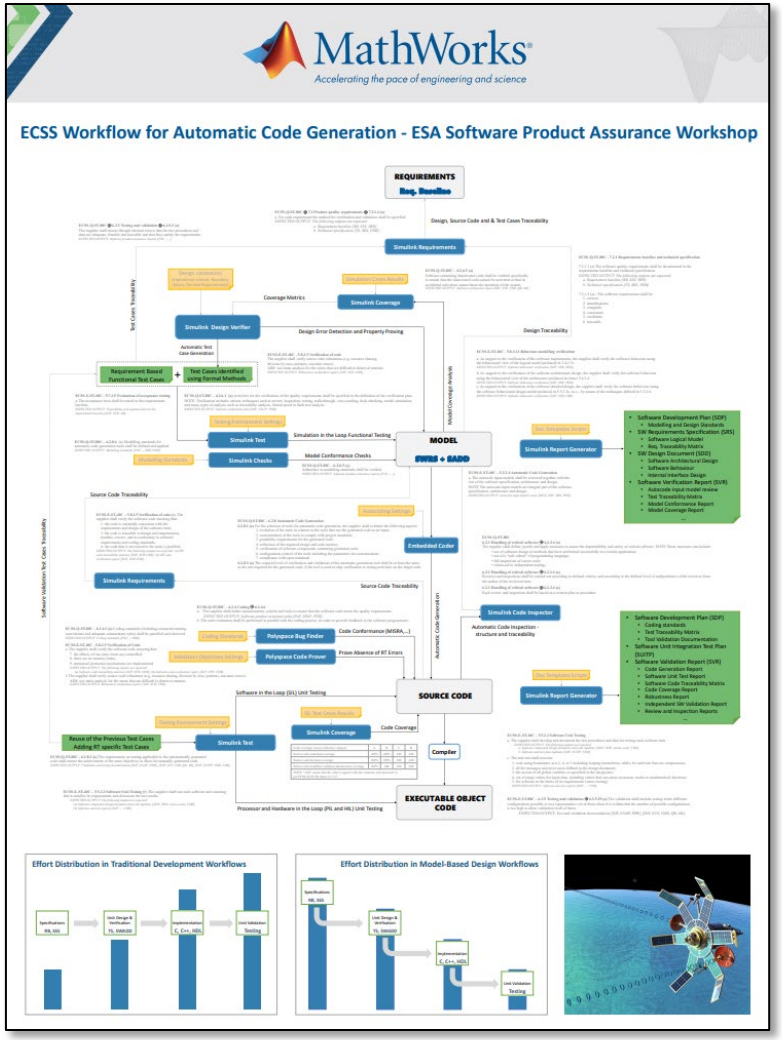
Orion Modeling Standards

- Orion MATLAB/Simulink Standards authored by NASA

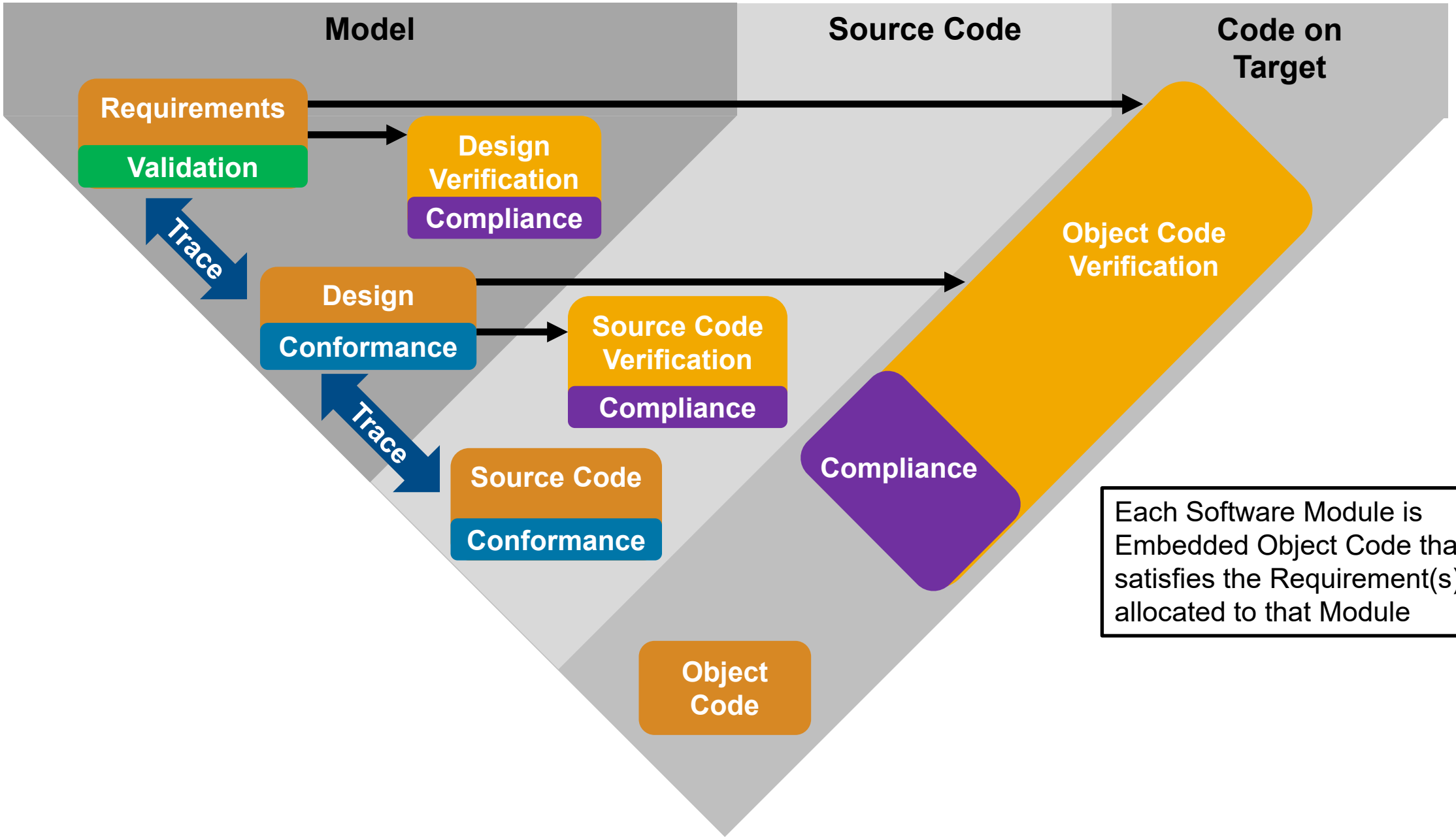


ECSS Software Standards

- ECSS E-40 and ECSS Q-80 workflows developed
- ESA Autocoding Working Group developing "Guidelines for the Automatic Code Generation for AOCS/GNC Flight SW Handbook"
  - Provides detailed modeling and coding guidelines to European space industry



Software Module Development Workflow



Each Software Module is Embedded Object Code that satisfies the Requirement(s) allocated to that Module

Thanks!

mariaf@mathworks.com