

— Verification of Avionics Systems Using Simulink Test and Simulink Real-Time

— MATLAB Expo | May 2023

Agenda

01

Introduction

02

Objectives & Challenges

03

How did Simulink Test and Simulink Real-Time help us?

04

Future plans

– Introduction

Introduction



Maciej Stefaniak

Advanced Lead Engineer, Systems Engineering, Avionics

Maciej has been responsible for development of new framework for system verification of avionics using Simulink Test, Simulink Real-Time and Real-Time platform from Speedgoat. The framework was successfully deployed in several projects, supporting development according to standards like ARP 4754A or DO-178C. His area of interest beyond Model In the Loop, Hardware In the Loop testing is control systems modelling and Model Based Systems Engineering. Maciej holds Master of Science degree in Radiocommunication from Warsaw University of Technology, Faculty of Electronics and Information Technology.

At GE Aerospace,
we **invent the future** of flight,
lift people up
And bring them **home safely**





We see an industry that matters to the world

- History of innovation
- Purpose-driven people
- Technologies to enable net-zero flight



GE Aerospace

Photo courtesy of Boeing featuring Craig Bomben, Boeing's Enterprise Chief Pilot and VP of Flight Operations. Photo taken prior to COVID-19 restrictions.



More than 100 years of innovation
**GE Aerospace has achieved
the following firsts:**

- U.S. jet engine & U.S. turboprop engine
- Mach 2 engine
- Composite fan blade in airline service
- World record for thrust – GE90 & GE9X
- Additive jet engine parts approved by U.S. FAA

'22 GE Aerospace... \$26.1B revenue^{-a)}

Commercial Engines & Services ... \$18.7B



Military Engines & Services ... \$4.4B



Aviation Systems ... \$1.6B



Avio Aero & Turboprops ... \$0.9B



Additive & Other ... \$0.5B



GE Aerospace global footprint

North America

- Canada
- Mexico
- U.S.A.

Asia & Australia

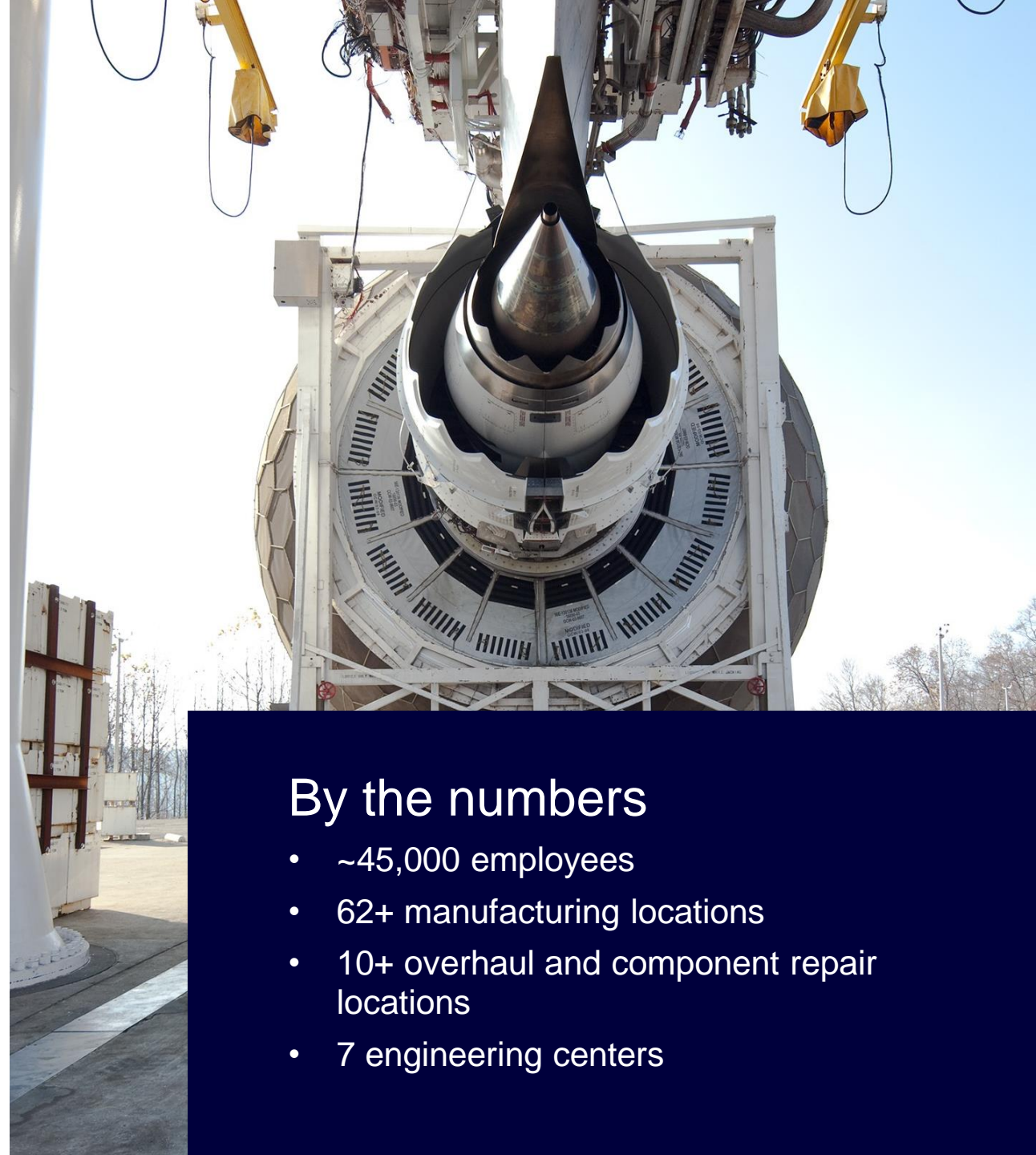
- Australia
- China
- Malaysia
- Korea
- Singapore
- Taiwan
- United Arab Emirates
- Qatar
- India

South America

- Brazil

Europe

- Czech Republic
- France
- Hungary
- Italy
- Poland
- Romania
- Turkey
- United Kingdom
- Germany

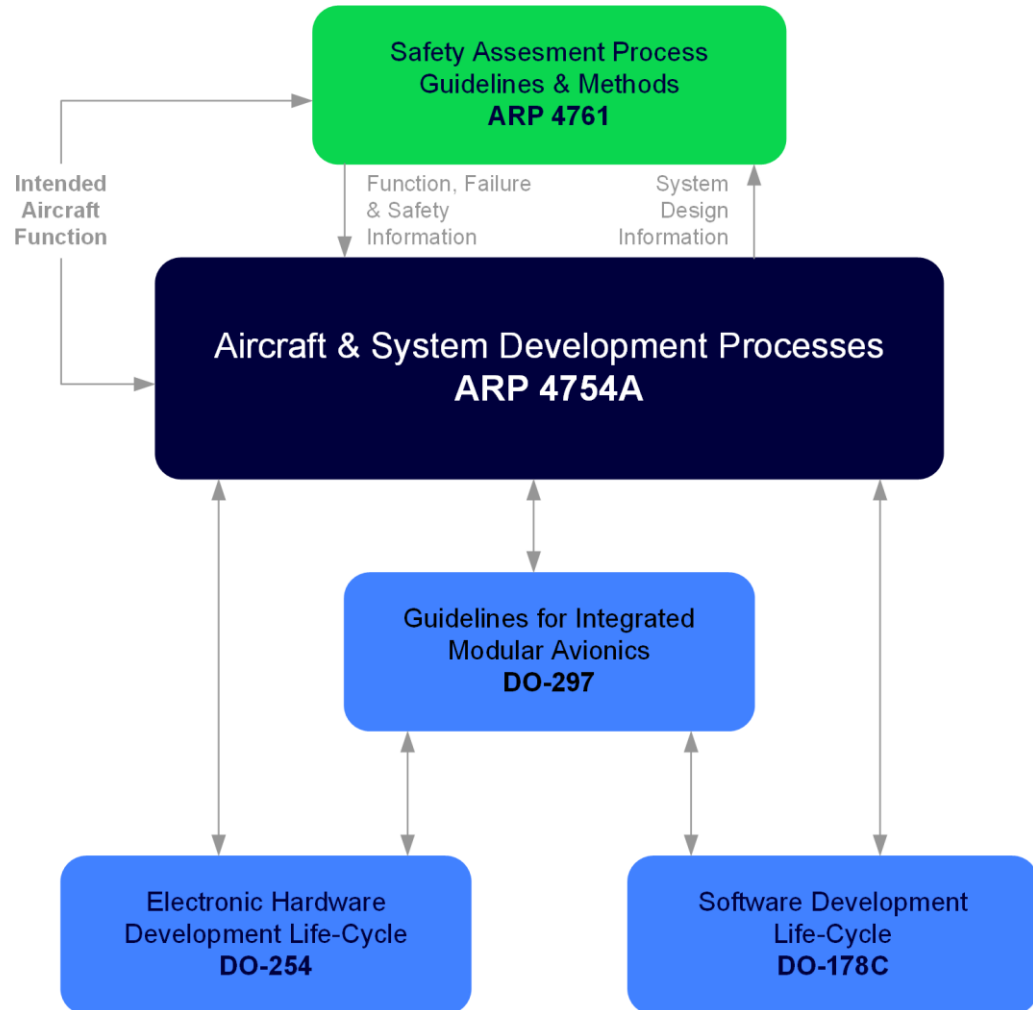


By the numbers

- ~45,000 employees
- 62+ manufacturing locations
- 10+ overhaul and component repair locations
- 7 engineering centers

– Objectives & Challenges

Business context



- Safety on the first place
- Highly regulated industry
- Certification – evidence of compliance
- Requirements Based Testing - every requirement needs to be verified

Real-Time MiL Cost saving
Automation Generic
HiL Reuse Easy to
Unified Certification use

Automation

- Reduce Human factor impact
- Test management
- Test preparation (compile&deploy)
- Test execution
- Test Result generation
- Test Report generation
- Test Result analysis

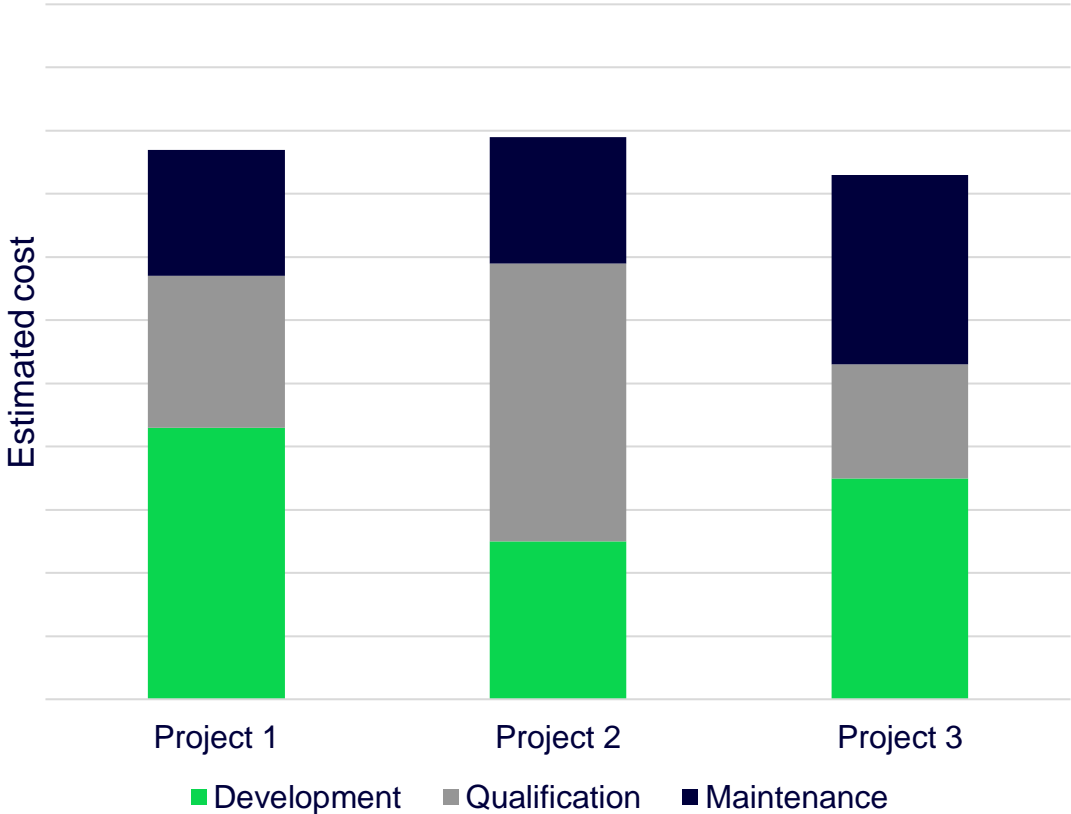
Generic & Reuse

- Common for all projects – projects are standardised
- Applicable to whole group of systems rather single product
- Building qualified test system - reuse of models, code and documentation*

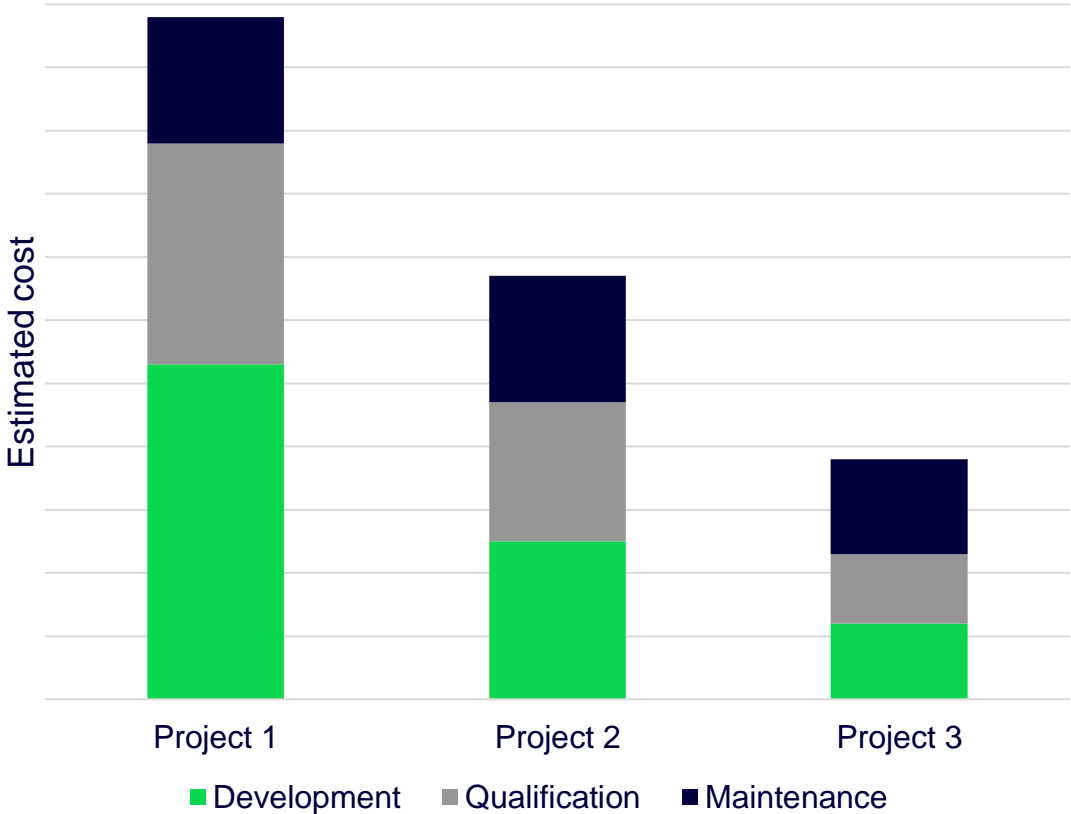
* RTCA DO-330 - Software Tool Qualification Considerations

Generic & Reuse

Each project builds its own Test Rig



Each project reuses Test Rig Framework



Real-Time

- Accurate and precise sample time representation
- Preferred 10 times faster than Unit Under Test (UUT)
- Both model & test executed in Real-Time on Target computer

Hardware in the Loop (HiL) & Model in the Loop (MiL)

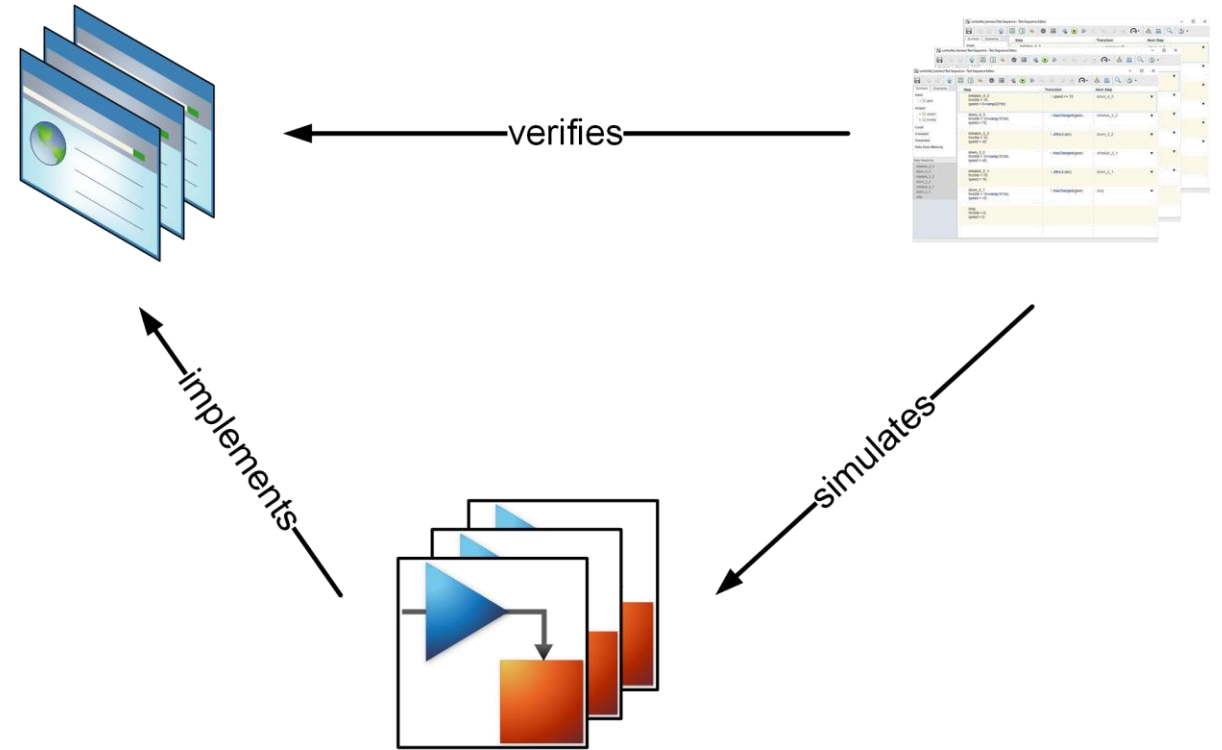
- Verification in open and closed control loop
- Different variety of interfaces: analogue and digital
- High range of communication protocols supported:
 - ARINC 429 (Serial)
 - ARINC 664 (Ethernet)
 - ARINC 825 (CAN)
- Integration with 3rd party Test&Measurement equipment
- MiL & HiL shall share common test
- Reduce Test Rig availability bottleneck

Unified & Easy to use

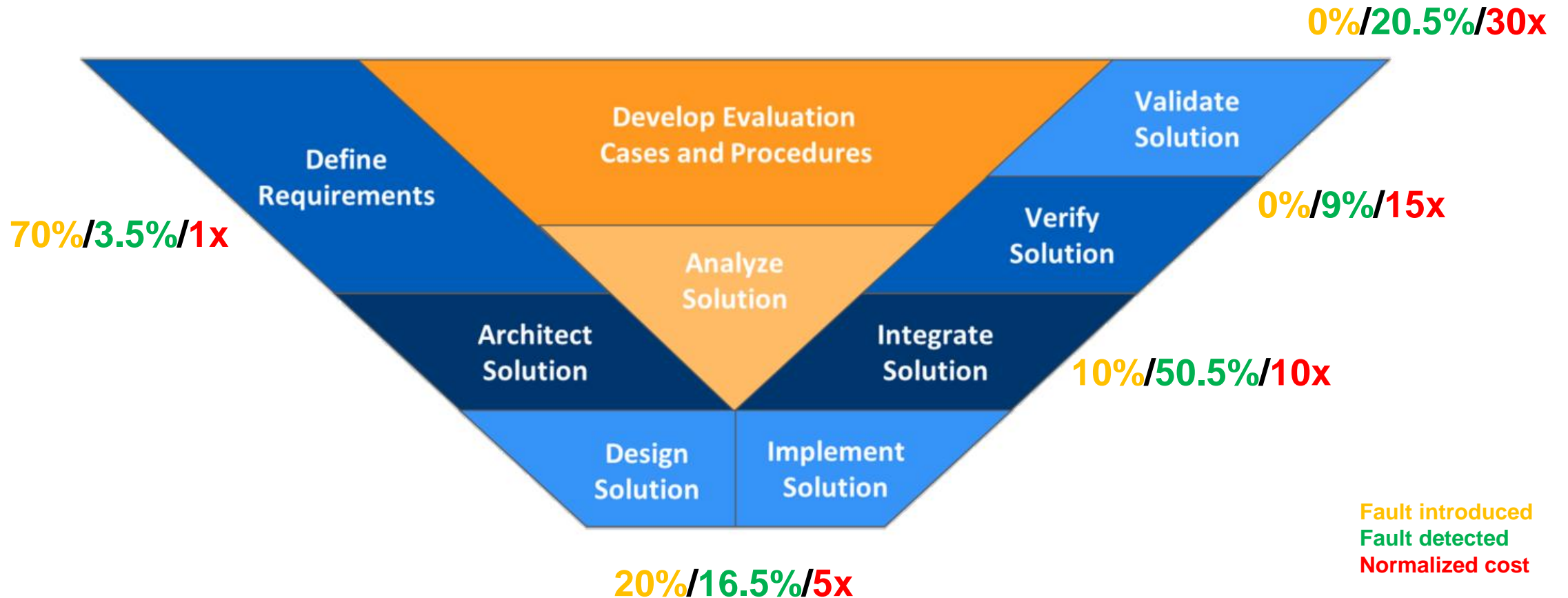
- Verification Team should focus on the UUT not tool itself
- Graphical User Interface (GUI)
- User Experience
- Fast to learn
- Predefined list of Input and Output signals (drop-down lists)
- Support for manual results analysis/debugging

Certification

- Deliver evidence to show compliance
- Traceability – „The recorded relationship established between two or more elements of the development process”
- Scale and complexity is challenging
- Coverage matrix generation
- Requirements verification
- Requirements validation



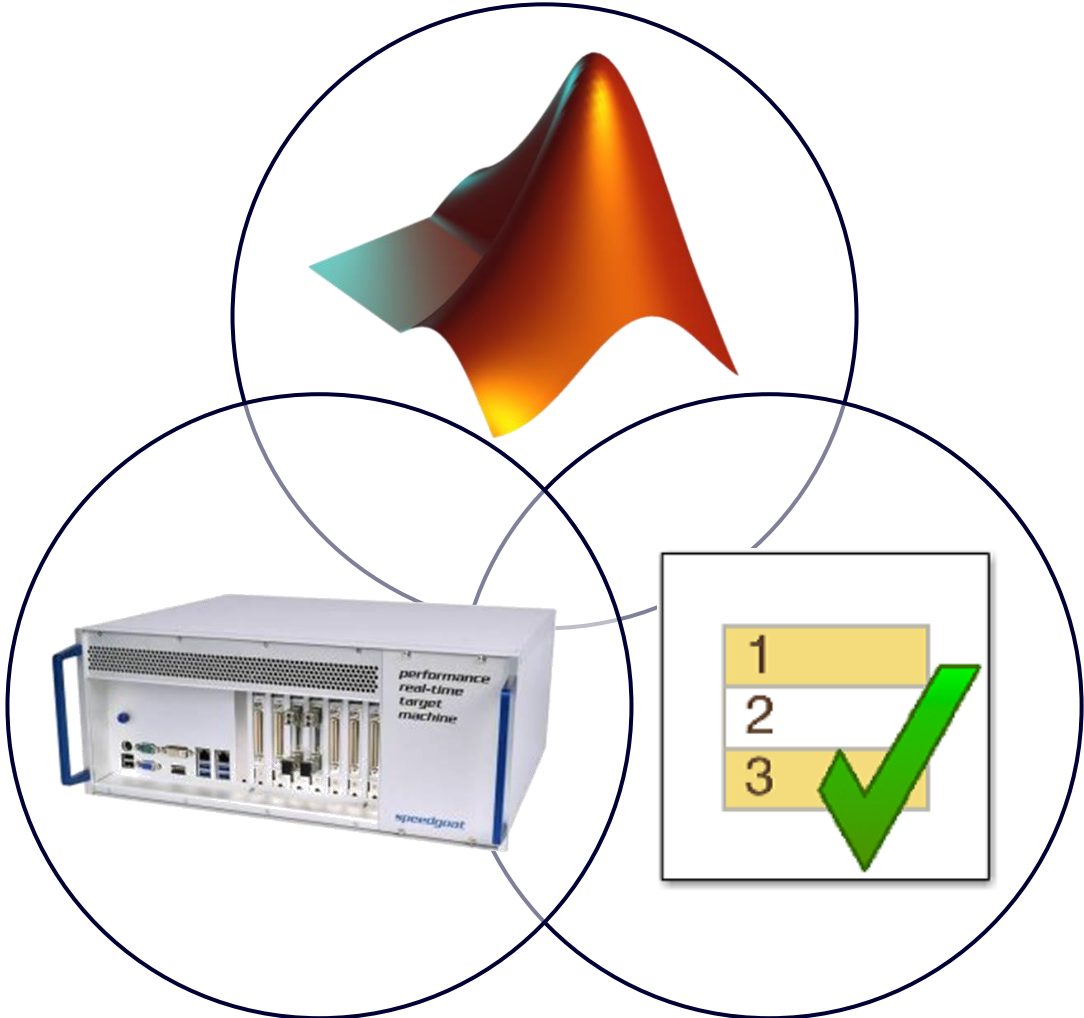
Requirements validation



Source: NIST, Planning Report 02-3 The Economic Impacts of Inadequate Infrastructure for Software Testing

- How did Simulink Test and Simulink Real-Time help us?

Three pillars of modern Test System



Speedgoat as HW platform



- Accelerate the design and testing of control designs and embedded controllers from MATLAB & Simulink
- Achieve most demanding sample rates and compute highly complex applications
- Vast range of supported I/O and large I/O expansion flexibility
- I/O modules are pre-installed at the time the target machine is purchased. Install additional I/O modules on your own at any time.
- Connect the target machine with your hardware, right from Simulink
- Flexible installation
- Maintained for the years to come

Roles in Test System

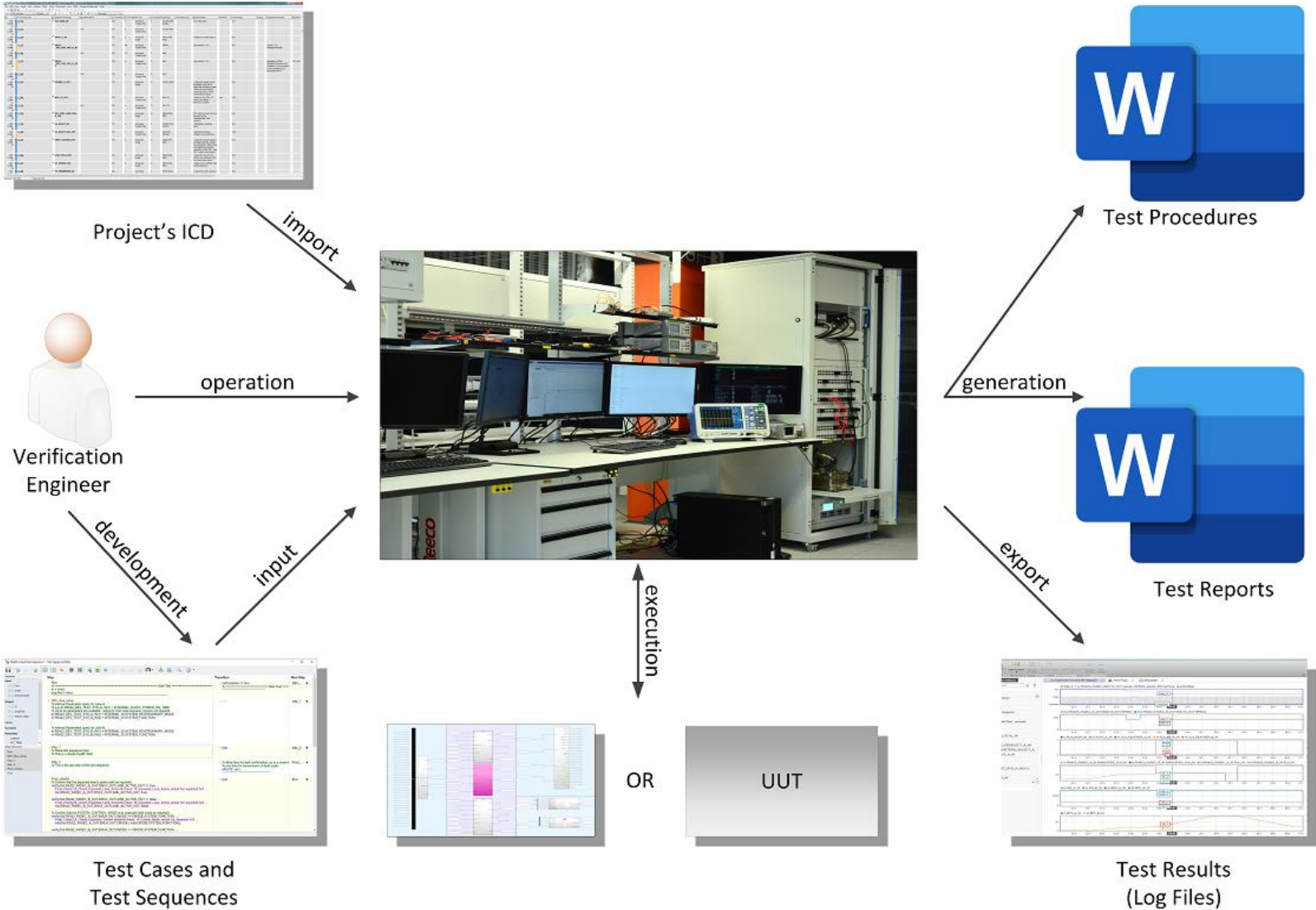
Verification Engineer

- Main Test System User
- Good understanding of requirements (UUT)
- Limited understanding of MATLAB/Simulink
- Uses predefined templates

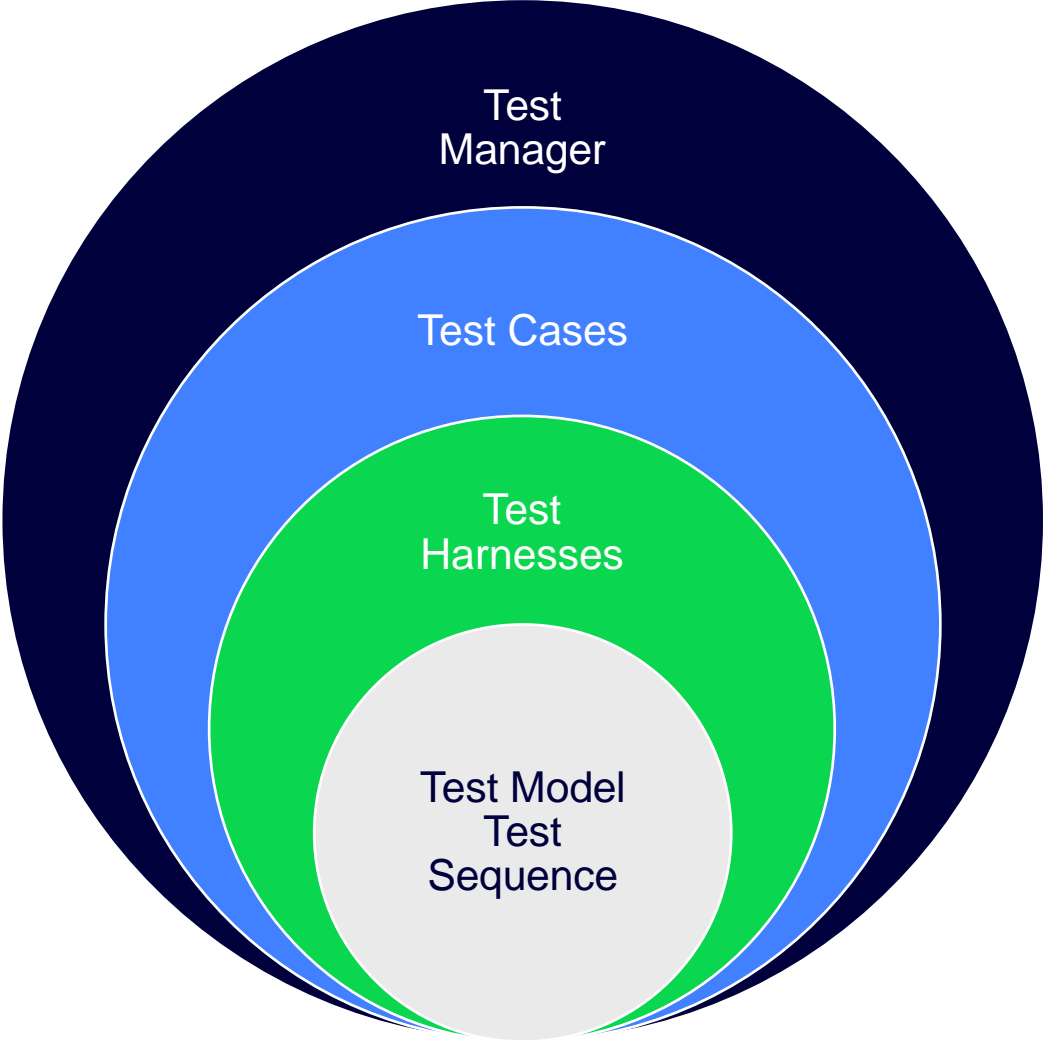
Test System Engineer

- Good understanding of MATLAB/Simulink
- Good understanding of Interface Control Document
- Good understanding of communication protocols
- Good understanding of Speedgoat HW
- Limited understanding of UUT

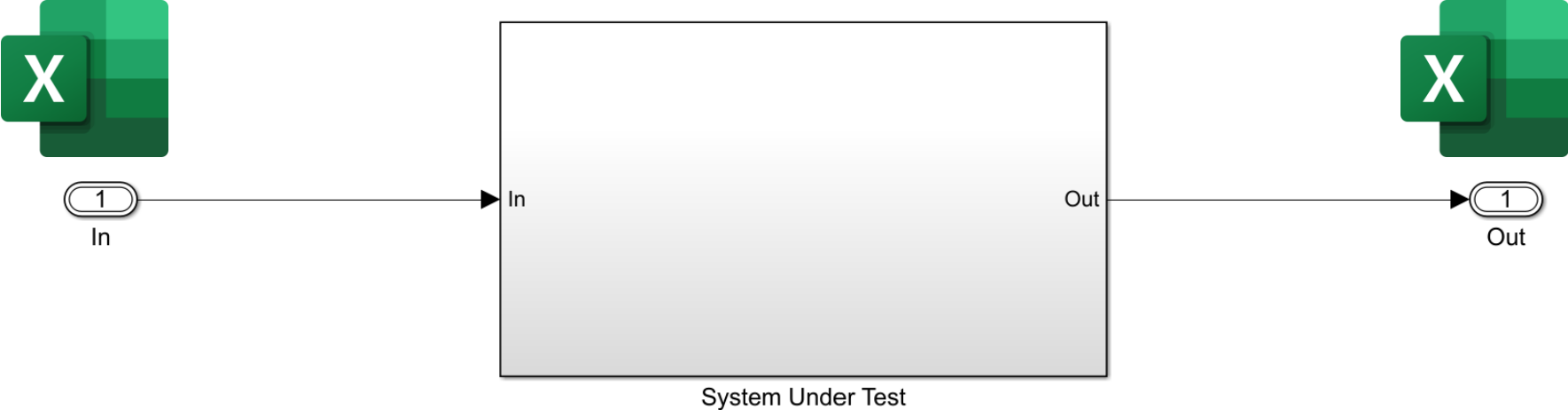
Test System context of use



Simulink Test hierarchy



Test Model



Test Sequence

calculator_Harness1/Test Sequence - Test Sequence Editor

Step	Transition	Next Step	Description
Run %% Initialize data outputs. In.powerSupply = false; In.arguments(1).arg1 = 0; In.arguments(1).arg2 = 0; In.arguments(1).operation = operationEnum.Add; In.arguments = repmat(In.arguments(1), 3, 1); % bus array	1. after(1, sec)	step_1 ▼	
step_1 % turn on calc In.powerSupply = true;	1. after(1, sec)	step_2 ▼	
step_2 % set arguments In.arguments(idx).arg1 = input1; In.arguments(idx).arg2 = input2; In.arguments(idx).operation = operationEnum.Multiply;	1. after(1, sec)	step_3 ▼	
step_3 % check results verify(abs(Out.results(idx) - expectedResult) < TOLERANCE, 'test:result_check', 'Addition failed: result = %f, expected = %f', Out.results(idx), expectedResult);	1. true	End ▼	
End			

Symbols


- Input**
 - 1. Out
- Output**
 - 1. In
- Local**
- Constant**
 - TOLERANCE
- Parameter**
 - expectedResult
 - idx
 - input1
 - input2
- Data Store Memory**

Step Hierarchy

- Run
- step_1
- step_2
- step_3
- End

Manual testing mode

```
mtm_TestModel_Harness/Test Sequence - Test Sequence Editor
```



Step

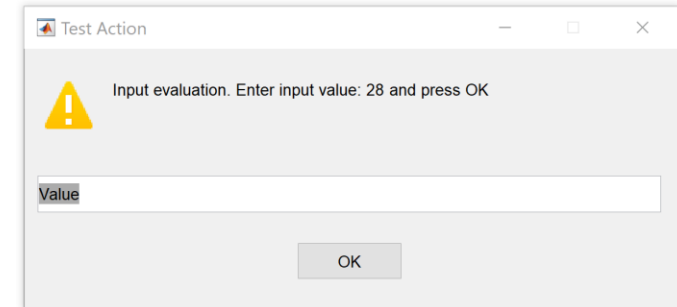
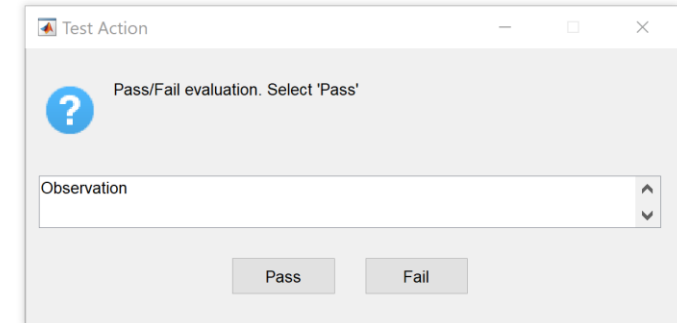
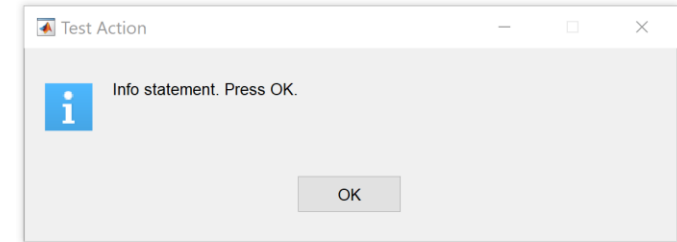
```
Run
In = InInit;
stopTest = false;
promptMode = winType.NoWindow;
promptMessage = sendmsg("");

OK_dialog
% OK dialog check
promptMode = winType.OK;
promptMessage = sendmsg("Info statement. Press OK.");

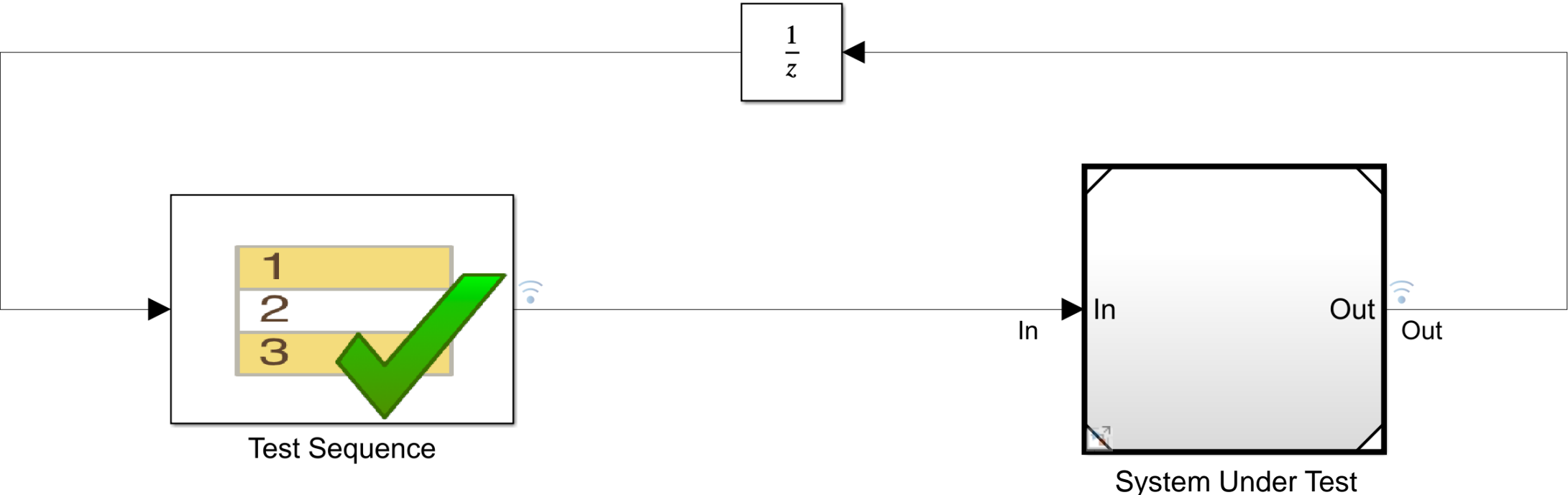
OK_dialog_post
promptMode = winType.NoWindow;
verify(testContinue == true, 'test:OK', 'Transition took too long');

Pass_Fail_dialog_pass
% PassFail dialog check (Pass)
promptMode = winType.PassFail;
promptMessage = sendmsg("Pass/Fail evaluation. Select \"Pass\"");

: Pass_Fail_dialog_pass_post
promptMode = winType.NoWindow;
verify(logical(userInput) == true, 'test:PassFail_Pass', 'Pass/Fail test failed');
```



Test Harness



Test Manager: Test Browser

The screenshot displays the Test Manager Test Browser interface. The top toolbar includes options for File (New, Open, Save), Edit (Cut, Copy, Paste, Delete, Test Spec Report), Run (Run, Run with Stepper, Stop, Parallel), Results (Report, Visualize, Highlight in Model), Environment (Import, Export, Preferences), and Resources (Help). The left sidebar shows a tree view with a filter 'Filter tests by name or tags, e.g. tags: test' and a folder structure: 'tf_example' > 'New Test Suite 1' > 'New Test Case 1'. The main area is titled 'New Test Case 1' and is marked as 'Enabled'. It contains the following configuration sections:

- Simulation Test**: Select releases for simulation: Current
- Create Test Case from External File
- TAGS**
- DESCRIPTION**
- REQUIREMENTS**
- SYSTEM UNDER TEST***:
 - Model: empty_model
 - TEST HARNESS***: Harness: empty_model_Harness1
 - SIMULATION SETTINGS OVERRIDES**
- PARAMETER OVERRIDES***
- CALLBACKS***
- INPUTS**
- SIMULATION OUTPUTS**
- CONFIGURATION SETTINGS OVERRIDES**
- ITERATIONS***
- LOGICAL AND TEMPORAL ASSESSMENTS**
- CUSTOM CRITERIA**

At the bottom left, a table lists the properties of the selected test case:

PROPERTY	VALUE
Name	New Test Case 1
Type	Simulation Test
Model	empty_model
Harness Name	empty_model_Harness1
Simulation Mode	[Model Settings]
Location	C:\Users\212679850\Docu...
Enabled	<input checked="" type="checkbox"/>
Hierarchy	tf_example » New Test Suit...
Type	Type, name, or tags, e.g. tags: test

Test Manager: Results and Artifacts

The screenshot displays the Test Manager interface with the following components:

- TESTS** tab selected.
- DATA INSPECTOR** toolbar with icons for Subplots, Legend, Transfer Signal View, Highlight in Model, Update Baseline, Previous Failure, and Next Failure.
- Results and Artifacts** panel showing a tree view of test results. The 'Verify Statements' folder is expanded, showing 'test:result_check' with a 'Pass' status.
- Properties Table** for the selected artifact 'Sim Output (calculator : ...)':

PROPERTY	VALUE
Name	Sim Output (calculator : ...)
Model	calculator
HarnessName	calculator_Harness
Simulation Mode	normal
Override SIL or PIL Mode	false
Configuration Set	Configuration1
Start Time	0
Stop Time	10
Checksum	3298R14187_14177R664_23
- Visualize** panel showing four time-series plots:
 - test:result_check**: A plot with a 'Pass' status at time 3.0.
 - In.powerSupply**: A plot showing a value of 1.00 at time 3.0.
 - In.arguments(1,1).arg1**: A plot showing a value of 1.00 at time 3.0.
 - Out.failureDetected**: A plot showing a value of 0 at time 3.0.

Simulink Requirements

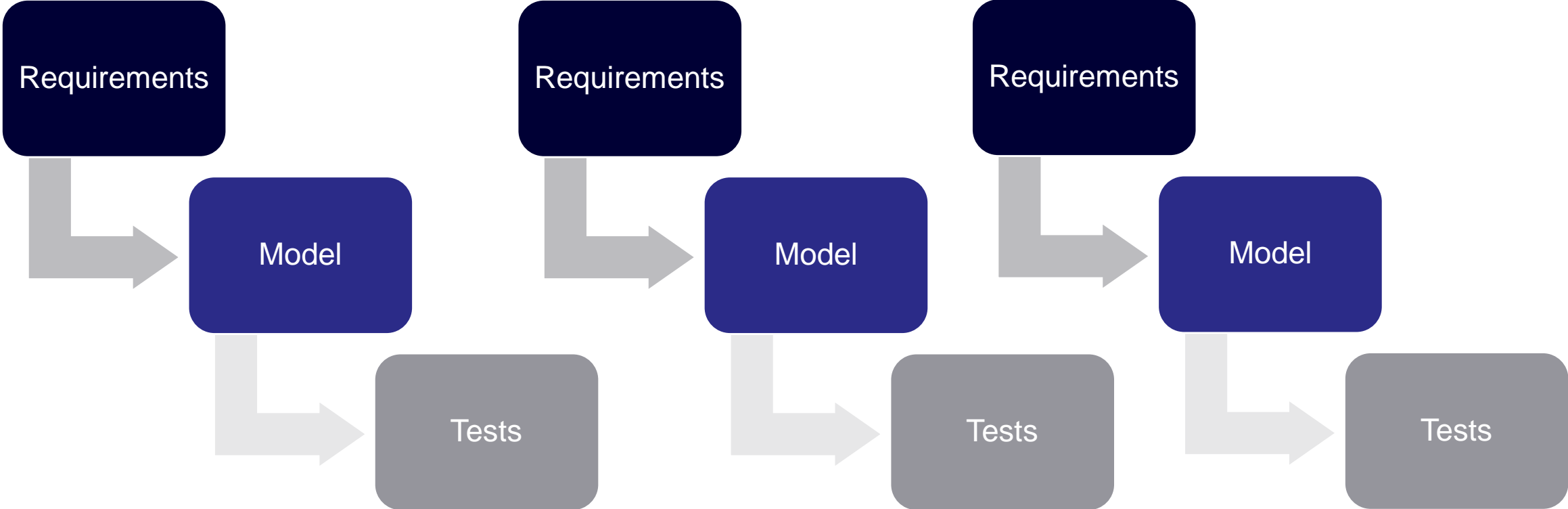
Export to Simulink Requirements

- Export requirements
- Manage requirements within Simulink Requirements
- Trace to Test Cases, Test Steps and Test Model
- Generate Traceability Matrix and others

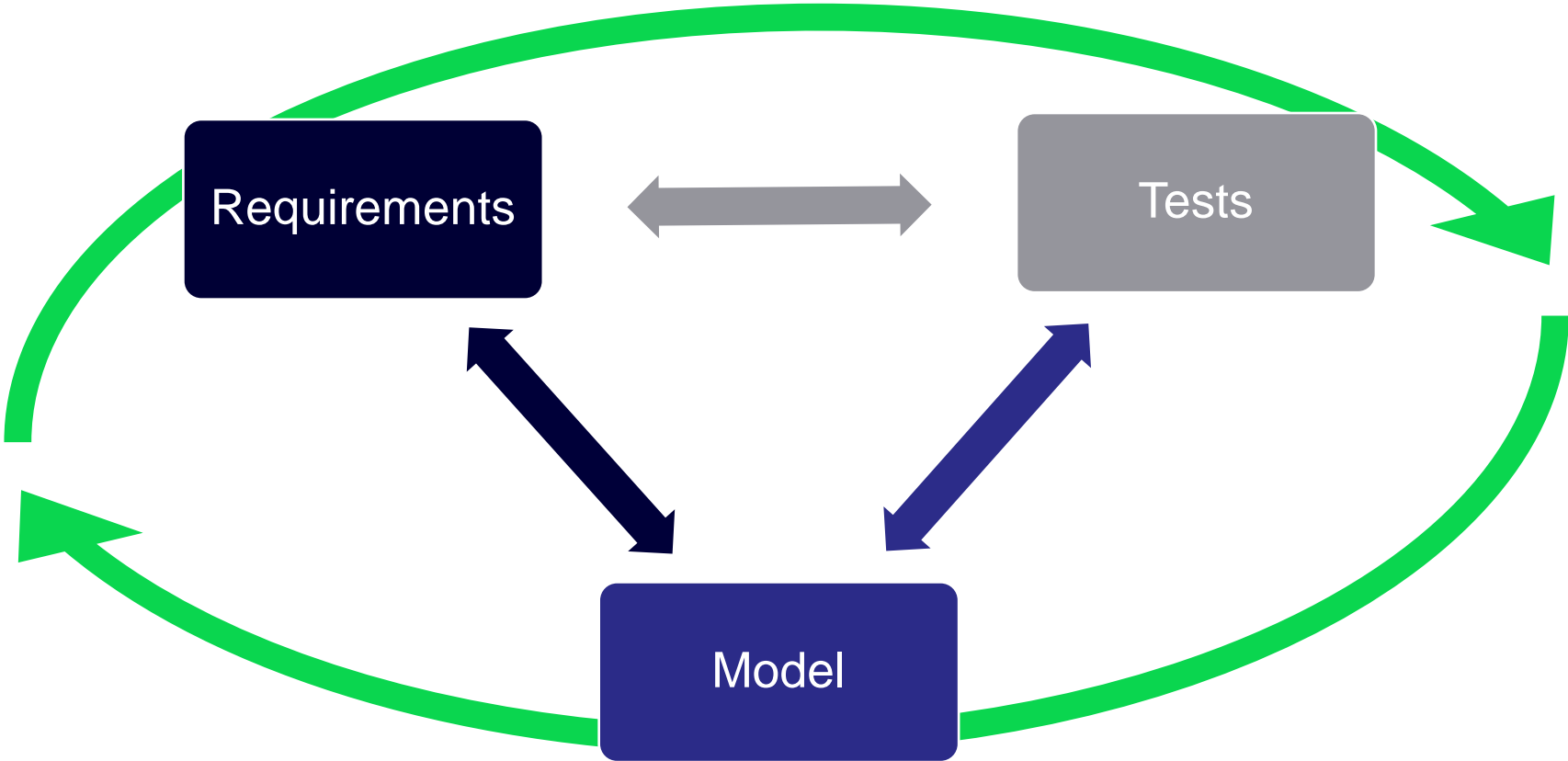
Integrate with Quality Management System

- Traceability done on QMS level – broader picture
- Test Manager needs to be integrated (execute tests & provide results)
- Better analytical capabilities
- More work and knowledge required

Requirements validation - before



Requirements validation - after



– Future plans

Future plans

- Transformation into CI/CD workflow
- Requirements based testing – coverage matrix
- Better integration with QMS
- Integration with 3rd party T&M equipment
- Migration to R2020b and beyond (in progress)
- Fixing bugs and improvements



GE Aerospace