Verification and Validation Test Framework Using a Model-Based Systems Engineering Approach

Clara Ramirez
Doctoral Student, Mechanical Engineering
Graduate Research Assistant
University of Connecticut
Email: clara.ramirez@uconn.edu

Amy Thompson, Ph.D.
Associate Professor-In-Residence, Systems Engineering
Associate Director, Pratt & Whitney Institute for Advanced Systems Engineering
University of Connecticut
Email: amy.2.thompson@uconn.edu

Motivation

- Automotive, aerospace, defense, and other industries are adopting model-based systems engineering (MBSE) approaches to replace document-centric methods.

- Adoption of MBSE not only for system design but also for verification, validation (V&V), and other lifecycle activities.

- Improve verification and validation (V&V) tests and analyses

- Improve V&V activities and processes

- Digitalization of V&V for earlier and continuous application

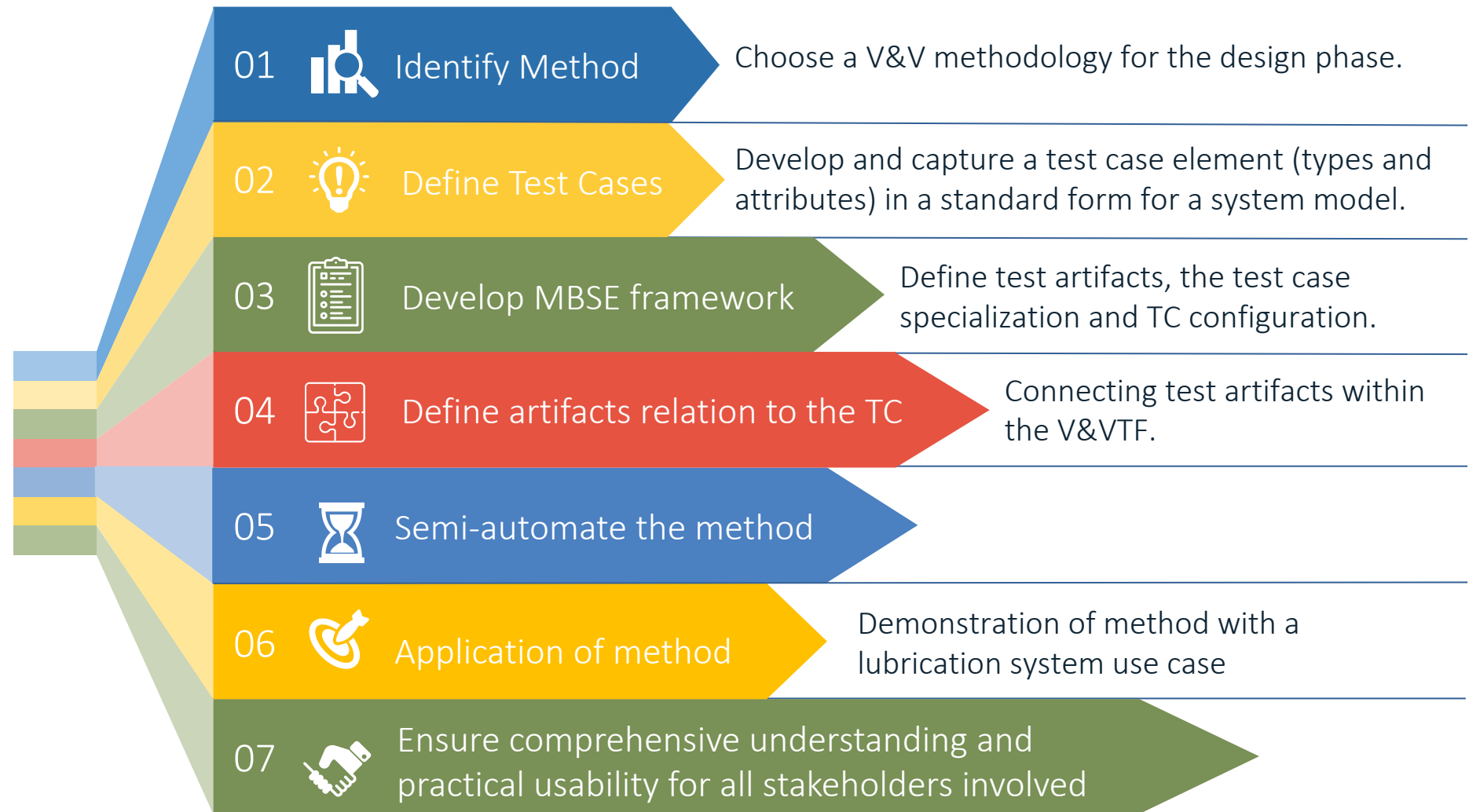- Consistent application of V&V approach for large scale complex system

Industry Application Focus

- Propose a general V&V Test Framework (V&VTF) using SysML and an MBSE tool for connecting systems models with physics-based models on a digital platform.

- Tailor the framework for large-scale engineered systems to effectively manage design changes and their propagation.

- Enable the creation of descriptive systems models across multiple engineering domains, avoiding domain-specific languages for simplicity and faster MBSE adoption.

- Design the framework to accommodate organizations with complex global supply chains, minimizing customizations and incompatible modeling methods among organizations.

What lies behind the developed approach?

Method Development Process

Based on a thorough literature review and identification of state-of-the-art approaches, selected a suitable verification and validation (V&V) methodology for the design phase.

01 Identify Method — Choose a V&V methodology for the design phase.

02 Define Test Cases — Develop and capture a test case element (types and attributes) in a standard form for a system model.

03 Develop MBSE framework — Define test artifacts, the test case specialization and TC configuration.

04 Define artifacts relation to the TC — Connecting test artifacts within the V&VTF.

05 Semi-automate the method

06 Application of method — Demonstration of method with a lubrication system use case

07 Ensure comprehensive understanding and practical usability for all stakeholders involved

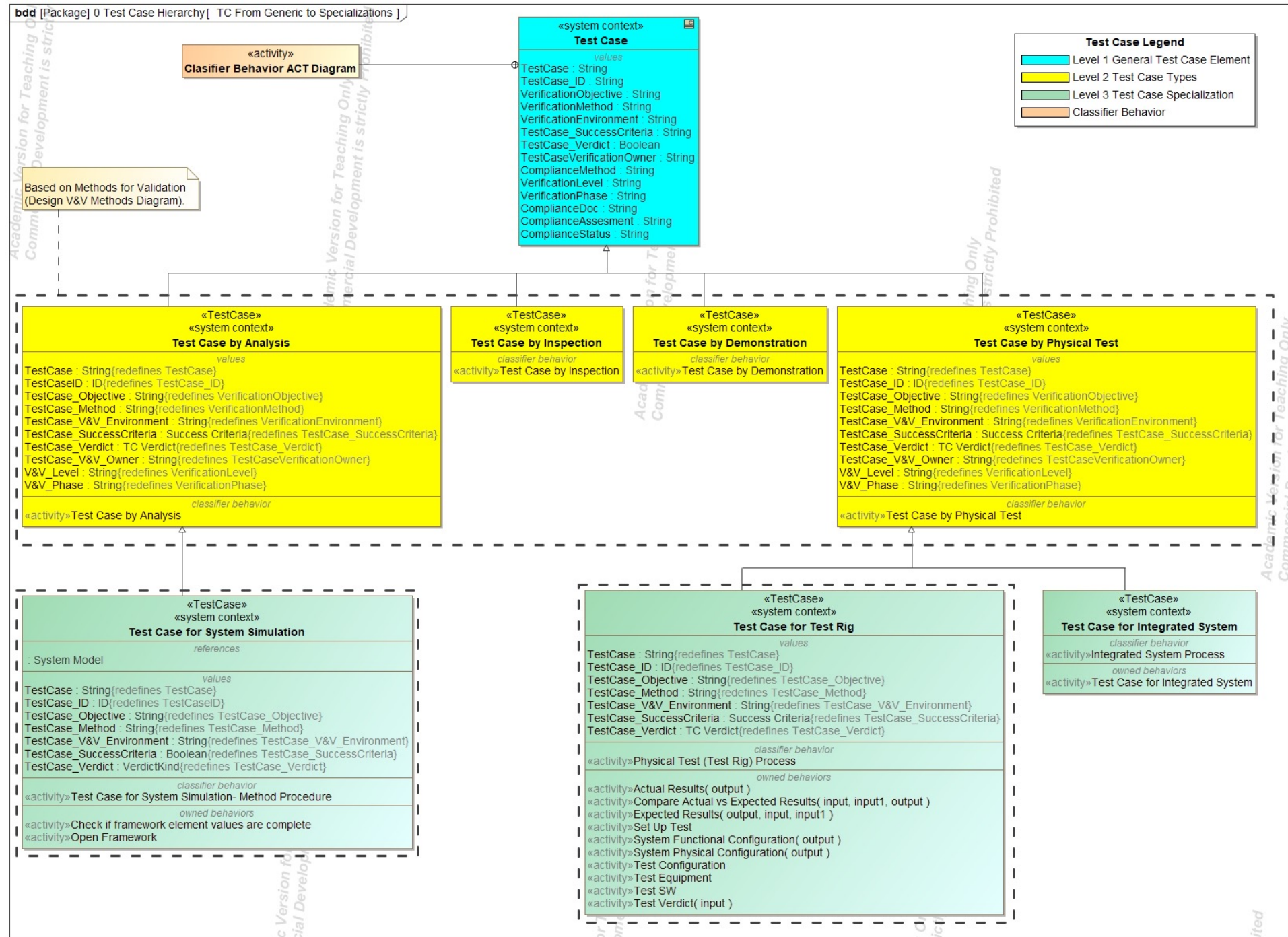This page does not contain any export regulated technical data.

Identifying the V&V Method

- Design Validation (DVD) focuses on ensuring that the implemented design meets the intended purpose and stakeholder expectations.

- DVD can be performed in a virtual or physical environment, with validation in a virtual environment often conducted through modeling and simulation.

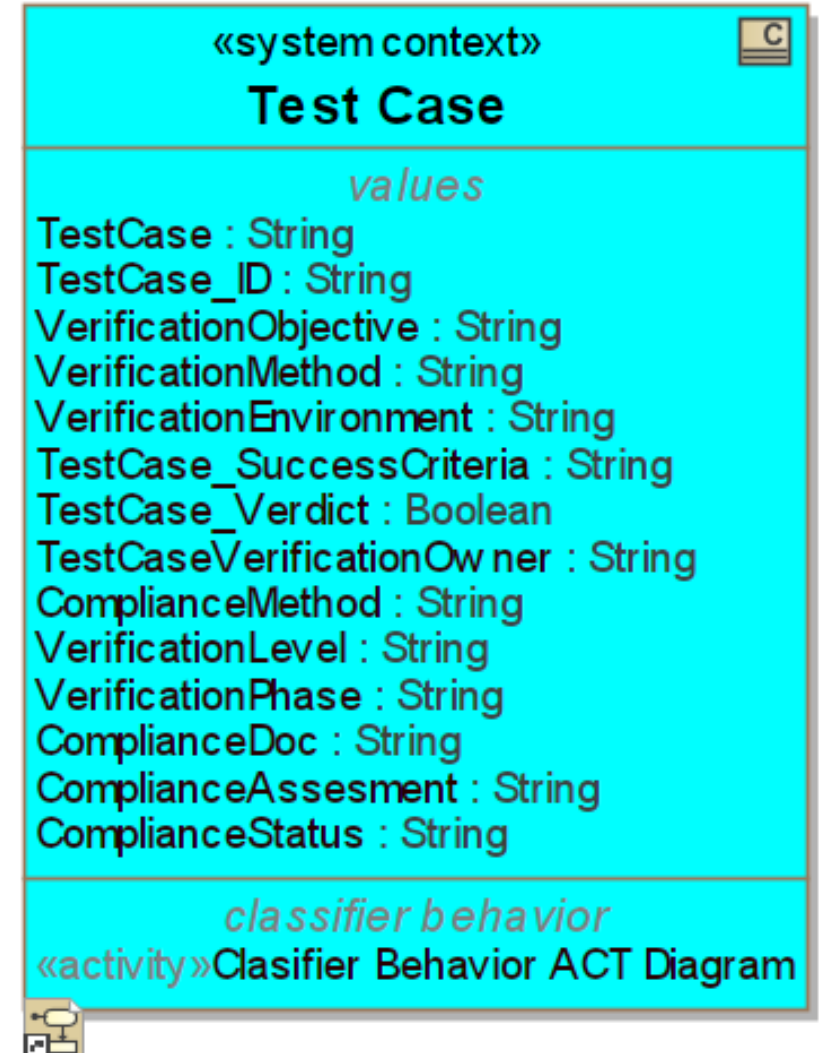| Operational Environment :Virtual & Physical	Methods For Validation
	Analysis (Simulation, Analogy, Sampling, Modeling)
	Inspection (Visual Examination, precision measurement)
	Demonstration (Qualitative determination)
	Test (Direct measurement with instrumentation, Direct measurement with special test equipment)
	Methods for Verification
	Enterprise Guidelines
	Enterprise Process
	Enterprise Rules
	Others

V&V Methods and Operational Environment Defined in the Model. Adapted from Guide to Verification and Validation (INCOSE, 2022)

This page does not contain any export regulated technical data.

Design Validation with Test Cases

bdd [Package] 0 Test Case Hierarchy [TC From Generic to Specializations]

Test Case Legend
- Level 1 General Test Case Element
- Level 2 Test Case Types
- Level 3 Test Case Specialization
- Classifier Behavior

«activity»
Clasifier Behavior ACT Diagram

Based on Methods for Validation
(Design V&V Methods Diagram).

«system context»
Test Case
values
TestCase : String
TestCase_ID : String
VerificationObjective : String
VerificationMethod : String
VerificationEnvironment : String
TestCase_SuccessCriteria : String
TestCase_Verdict : Boolean
TestCaseVerificationOwner : String
ComplianceMethod : String
VerificationLevel : String
VerificationPhase : String
ComplianceDoc : String
ComplianceAssesment : String
ComplianceStatus : String

«TestCase»
«system context»
Test Case by Analysis
values
TestCase : String{redefines TestCase}
TestCaseID : ID{redefines TestCase_ID}
TestCase_Objective : String{redefines VerificationObjective}
TestCase_Method : String{redefines VerificationMethod}
TestCase_V&V_Environment : String{redefines VerificationEnvironment}
TestCase_SuccessCriteria : Success Criteria{redefines TestCase_SuccessCriteria}
TestCase_Verdict : TC Verdict{redefines TestCase_Verdict}
TestCase_V&V_Owner : String{redefines TestCaseVerificationOwner}
V&V_Level : String{redefines VerificationLevel}
V&V_Phase : String{redefines VerificationPhase}
classifier behavior
«activity»Test Case by Analysis

«TestCase»
«system context»
Test Case by Inspection
classifier behavior
«activity»Test Case by Inspection

«TestCase»
«system context»
Test Case by Demonstration
classifier behavior
«activity»Test Case by Demonstration

«TestCase»
«system context»
Test Case by Physical Test
values
TestCase : String{redefines TestCase}
TestCase_ID : ID{redefines TestCase_ID}
TestCase_Objective : String{redefines VerificationObjective}
TestCase_Method : String{redefines VerificationMethod}
TestCase_V&V_Environment : String{redefines VerificationEnvironment}
TestCase_SuccessCriteria : Success Criteria{redefines TestCase_SuccessCriteria}
TestCase_Verdict : TC Verdict{redefines TestCase_Verdict}
TestCase_V&V_Owner : String{redefines TestCaseVerificationOwner}
V&V_Level : String{redefines VerificationLevel}
V&V_Phase : String{redefines VerificationPhase}
classifier behavior
«activity»Test Case by Physical Test

«TestCase»
«system context»
Test Case for System Simulation
references
: System Model
values
TestCase : String{redefines TestCase}
TestCase_ID : ID{redefines TestCaseID}
TestCase_Objective : String{redefines TestCase_Objective}
TestCase_Method : String{redefines TestCase_Method}
TestCase_V&V_Environment : String{redefines TestCase_V&V_Environment}
TestCase_SuccessCriteria : Boolean{redefines TestCase_SuccessCriteria}
TestCase_Verdict : VerdictKind{redefines TestCase_Verdict}
classifier behavior
«activity»Test Case for System Simulation- Method Procedure
owned behaviors
«activity»Check if framework element values are complete
«activity»Open Framework

«TestCase»
«system context»
Test Case for Test Rig
values
TestCase : String{redefines TestCase}
TestCase_ID : ID{redefines TestCase_ID}
TestCase_Objective : String{redefines TestCase_Objective}
TestCase_Method : String{redefines TestCase_Method}
TestCase_V&V_Environment : String{redefines TestCase_V&V_Environment}
TestCase_SuccessCriteria : Success Criteria{redefines TestCase_SuccessCriteria}
TestCase_Verdict : TC Verdict{redefines TestCase_Verdict}
classifier behavior
«activity»Physical Test (Test Rig) Process
owned behaviors
«activity»Actual Results(output)
«activity»Compare Actual vs Expected Results(input, input1, output)
«activity»Expected Results(output, input, input1)
«activity»Set Up Test
«activity»System Functional Configuration(output)
«activity»System Physical Configuration(output)
«activity»Test Configuration
«activity»Test Equipment
«activity»Test SW
«activity»Test Verdict(input)

«TestCase»
«system context»
Test Case for Integrated System
classifier behavior
«activity»Integrated System Process
owned behaviors
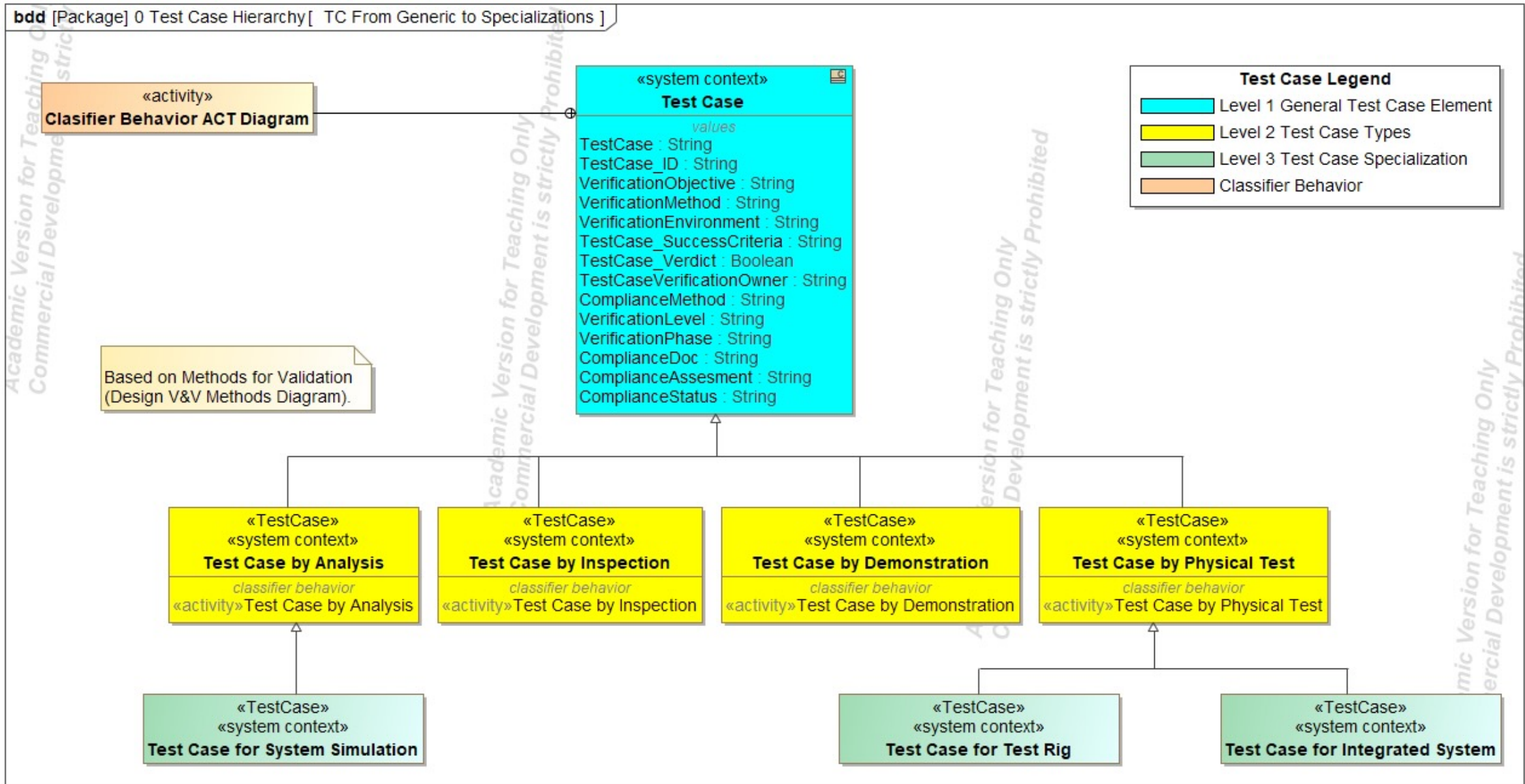«activity»Test Case for Integrated System

7/20/2023

Design Validation with Test Cases

- SysML Test Case (TC) Element, (Adapted from Selvey et al.'s Verification Element (2018) and Scukanec (2011))

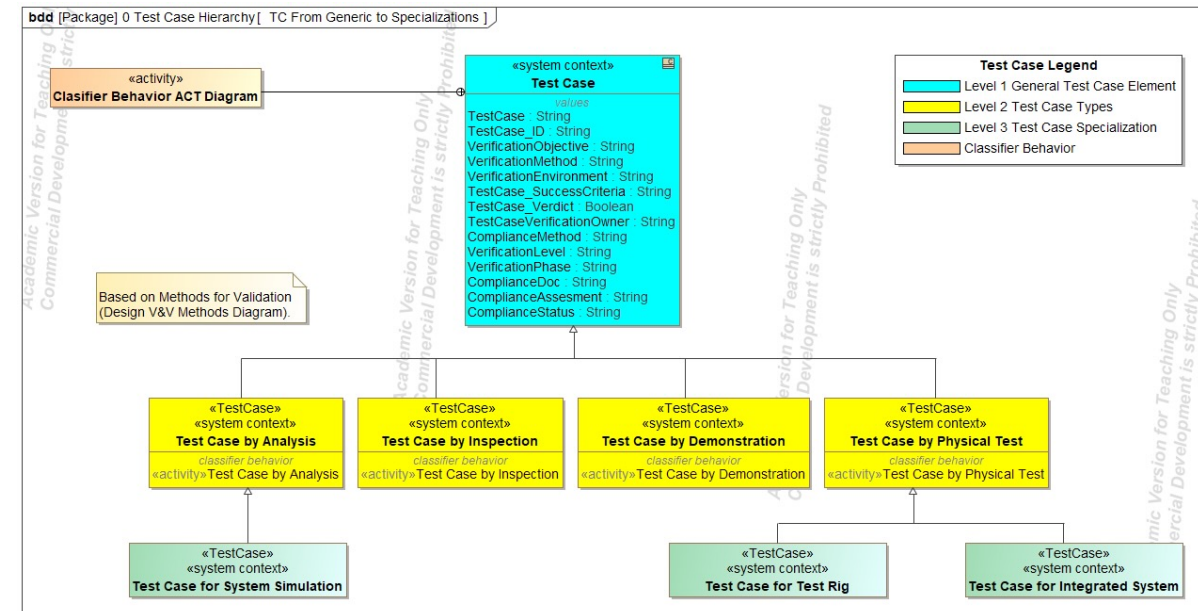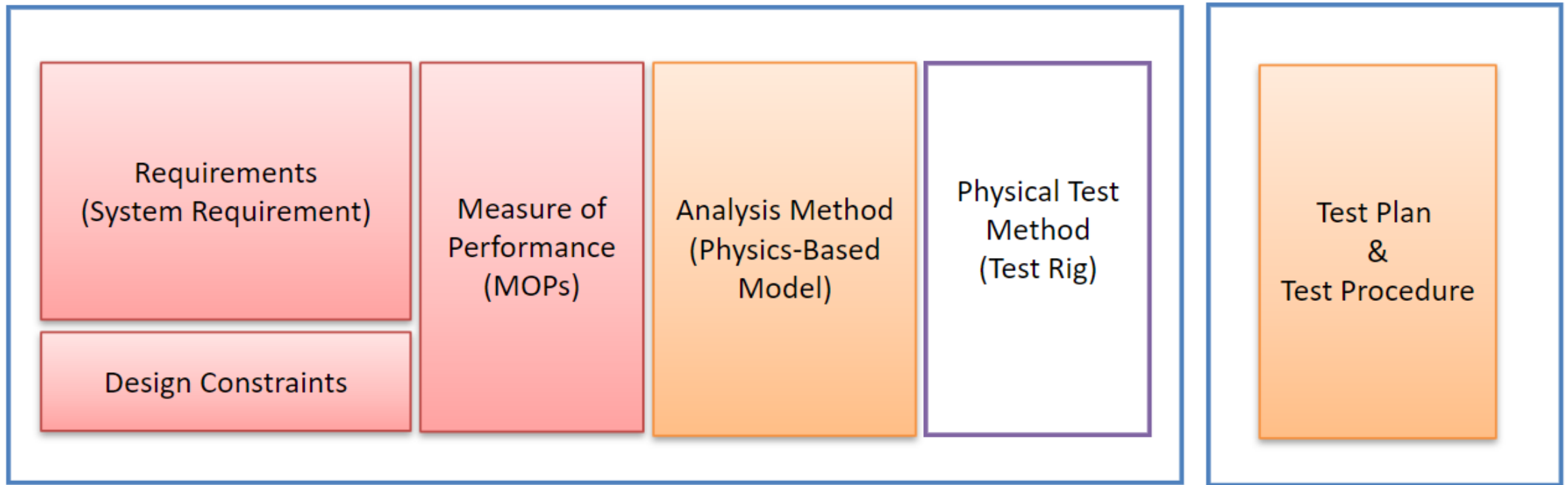- TC block contains a classifier behavior

«system context»
Test Case

values

TestCase : String
TestCase_ID : String
VerificationObjective : String
VerificationMethod : String
VerificationEnvironment : String
TestCase_SuccessCriteria : String
TestCase_Verdict : Boolean
TestCaseVerificationOwner : String
ComplianceMethod : String
VerificationLevel : String
VerificationPhase : String
ComplianceDoc : String
ComplianceAssesment : String
ComplianceStatus : String

classifier behavior

«activity»Clasifier Behavior ACT Diagram

This page does not contain any export regulated technical data.

Design Validation with Test Cases

bdd [Package] 0 Test Case Hierarchy [TC From Generic to Specializations]

«activity»
Clasifier Behavior ACT Diagram

«system context»
Test Case

values
TestCase : String
TestCase_ID : String
VerificationObjective : String
VerificationMethod : String
VerificationEnvironment : String
TestCase_SuccessCriteria : String
TestCase_Verdict : Boolean
TestCaseVerificationOwner : String
ComplianceMethod : String
VerificationLevel : String
VerificationPhase : String
ComplianceDoc : String
ComplianceAssesment : String
ComplianceStatus : String

Test Case Legend
Level 1 General Test Case Element
Level 2 Test Case Types
Level 3 Test Case Specialization
Classifier Behavior

Based on Methods for Validation
(Design V&V Methods Diagram).

«TestCase»
«system context»
Test Case by Analysis
classifier behavior
«activity»Test Case by Analysis

«TestCase»
«system context»
Test Case by Inspection
classifier behavior
«activity»Test Case by Inspection

«TestCase»
«system context»
Test Case by Demonstration
classifier behavior
«activity»Test Case by Demonstration

«TestCase»
«system context»
Test Case by Physical Test
classifier behavior
«activity»Test Case by Physical Test

«TestCase»
«system context»
Test Case for System Simulation

«TestCase»
«system context»
Test Case for Test Rig

«TestCase»
«system context»
Test Case for Integrated System

Design Validation with Test Cases

- Efficient modeling process for test case generation including definition of test case and activities

- Reusability by different types and context

- User can customize test cases using standard approach or add elements according to test type

This page does not contain any export regulated technical data.

Current Practices

Current Document Based Approach for Test Requests

Requirements (System Requirement)	Measure of Performance (MOPs)	Analysis Method (Physics-Based Model)	Physical Test Method (Test Rig)	Test Plan & Test Procedure
Design Constraints				

Verification and Validation Test Framework (TF) and Elements

MBSE Methodology

Requirements — *verifies* ← Test Case — *employs* → Test Artifacts

Test Case — *validates* → System Under Test

Test Case → Test Configuration — *composed by* → Test Model Configuration/Test Physical Configuration

This page does not contain any export regulated technical data.
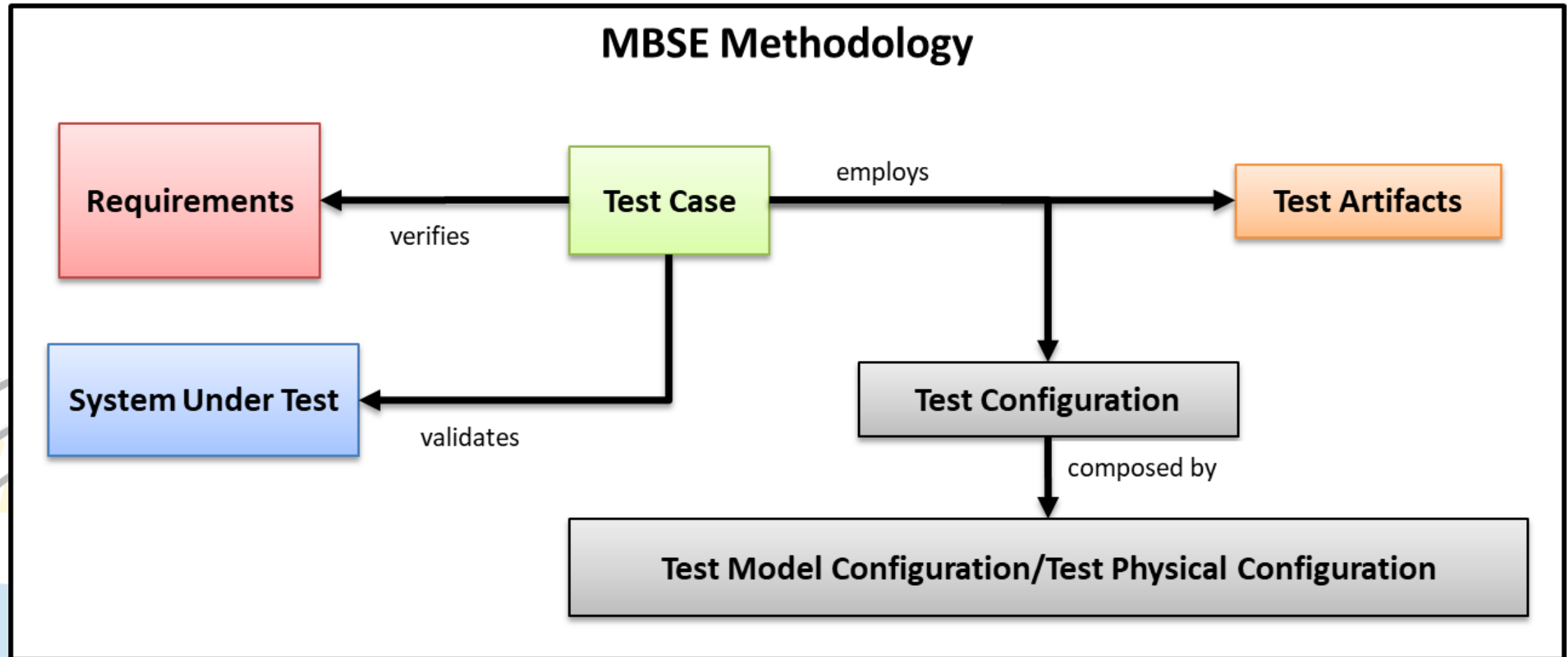
Proposed Verification and Validation Test Framework (TF) Artifacts

- Develop a V&V library of test frameworks that captures method

- Customized SysML elements such as stereotypes

- Defined model and physical configuration

- Captured test plan, test procedure, and test run.

- Captured the requirements related to the TC (for the SCS) and variables from the physics-based model using a functional mockup unit (FMU) block (a FMU file contains a simulation model that adheres to the FMI standard).

This page does not contain any export regulated technical data.
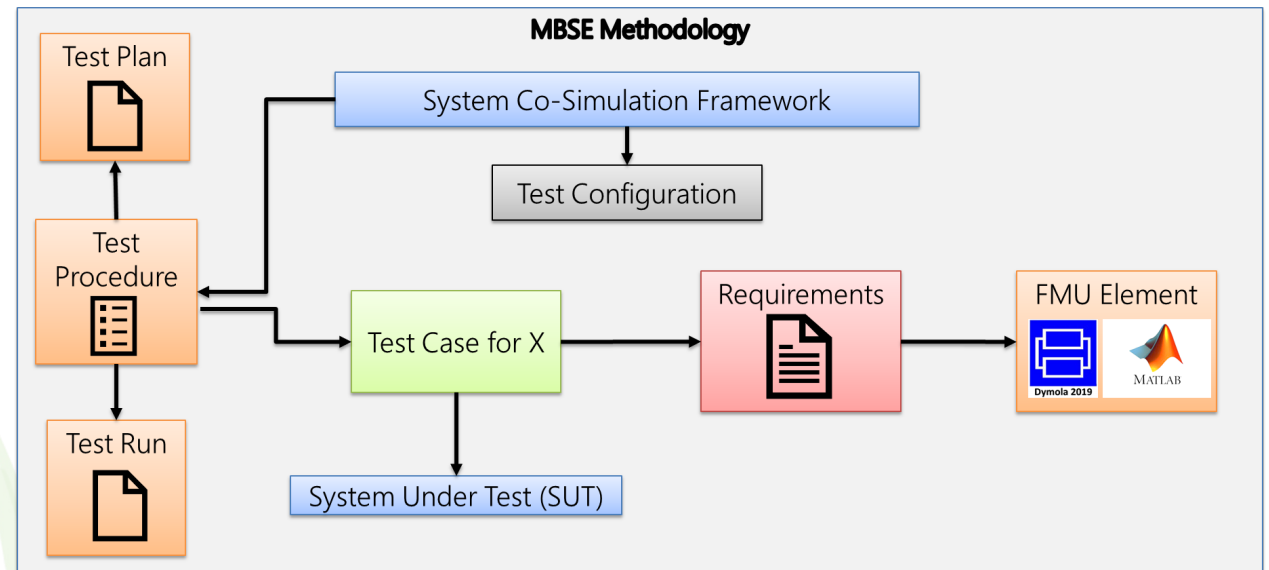
Proposed Verification and Validation Test Framework (TF) Artifacts

- Capturing Artifacts relation to Test Case
  - *Test Plan,* a detailed document that describes the test strategy, objectives, schedule, estimation, deliverables, and resources required to perform testing for a system.
  - *Test Procedure*, created to confirm that a system will meet or not meet requirements. In a typical scenario, each set of test procedures linked to requirements specifications and contains a set of steps that will be performed during the test.
  - *Test Run*, a unique occurrence of the running of a test procedure. There may be multiple test runs of a single procedure.

This page does not contain any export regulated technical data.
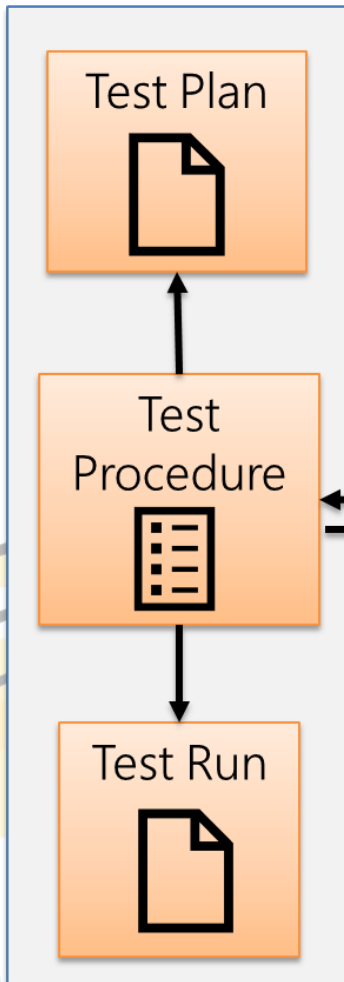
Design Validation by System Co-Simulation

Test Case by Analysis example, using system co-simulation framework & customized "validate" stereotype
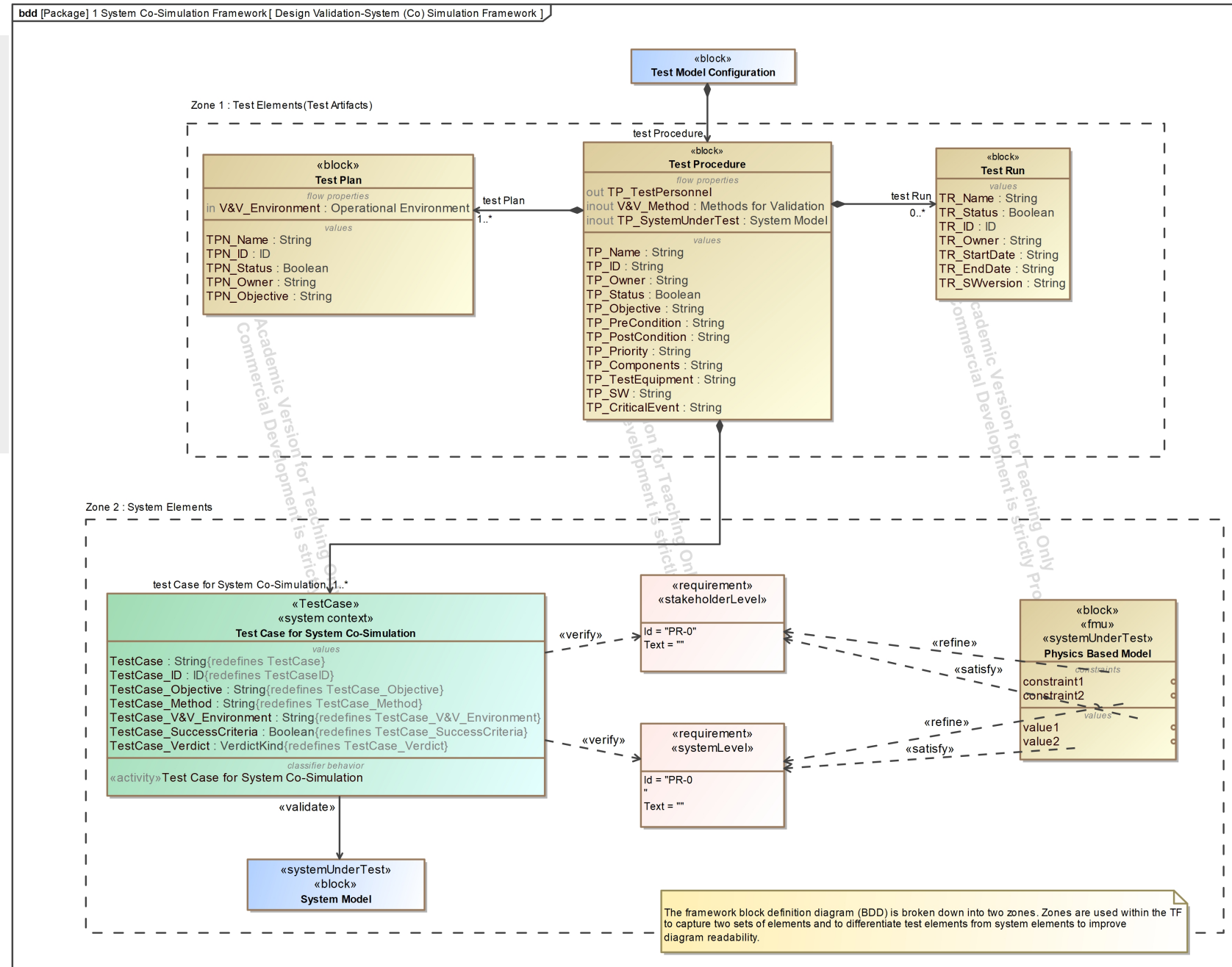
Design Validation by System Co-Simulation
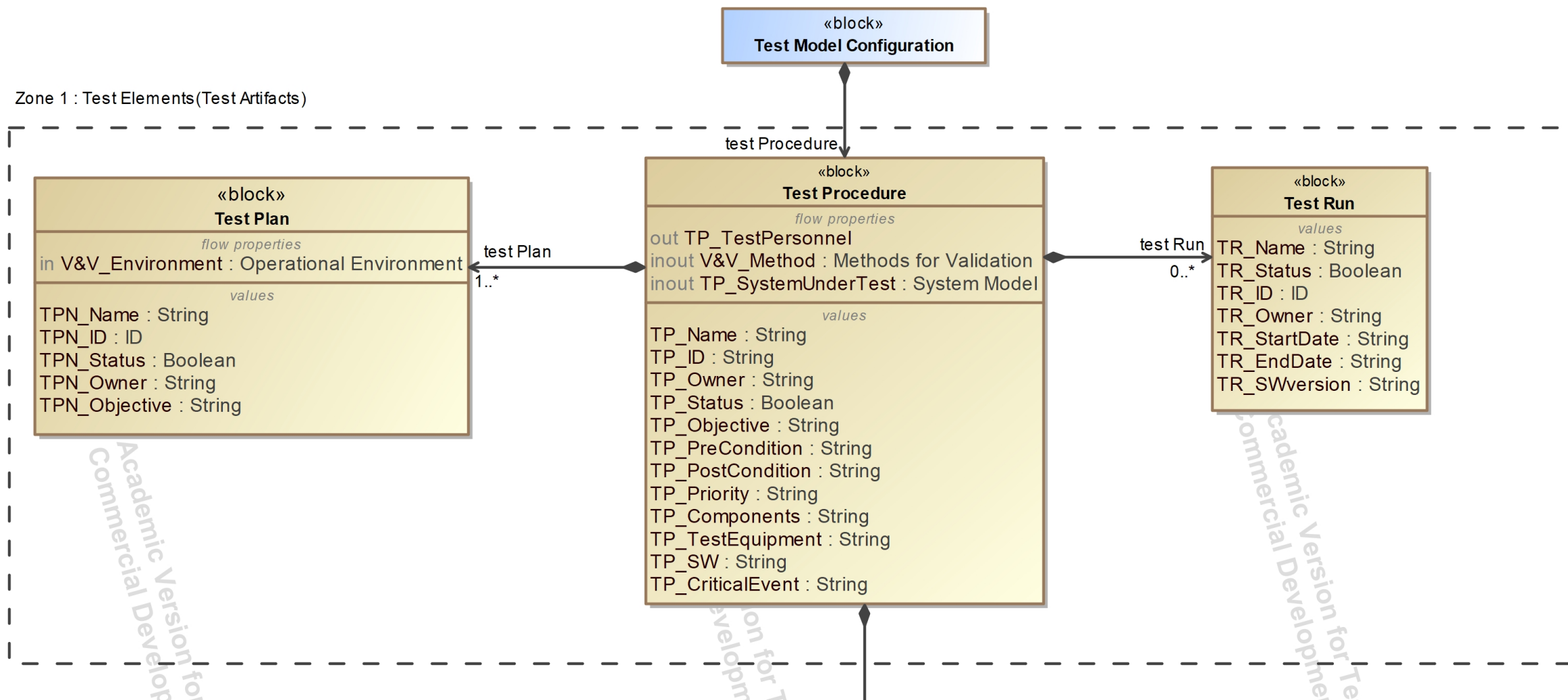
Design Validation by System Co-Simulation
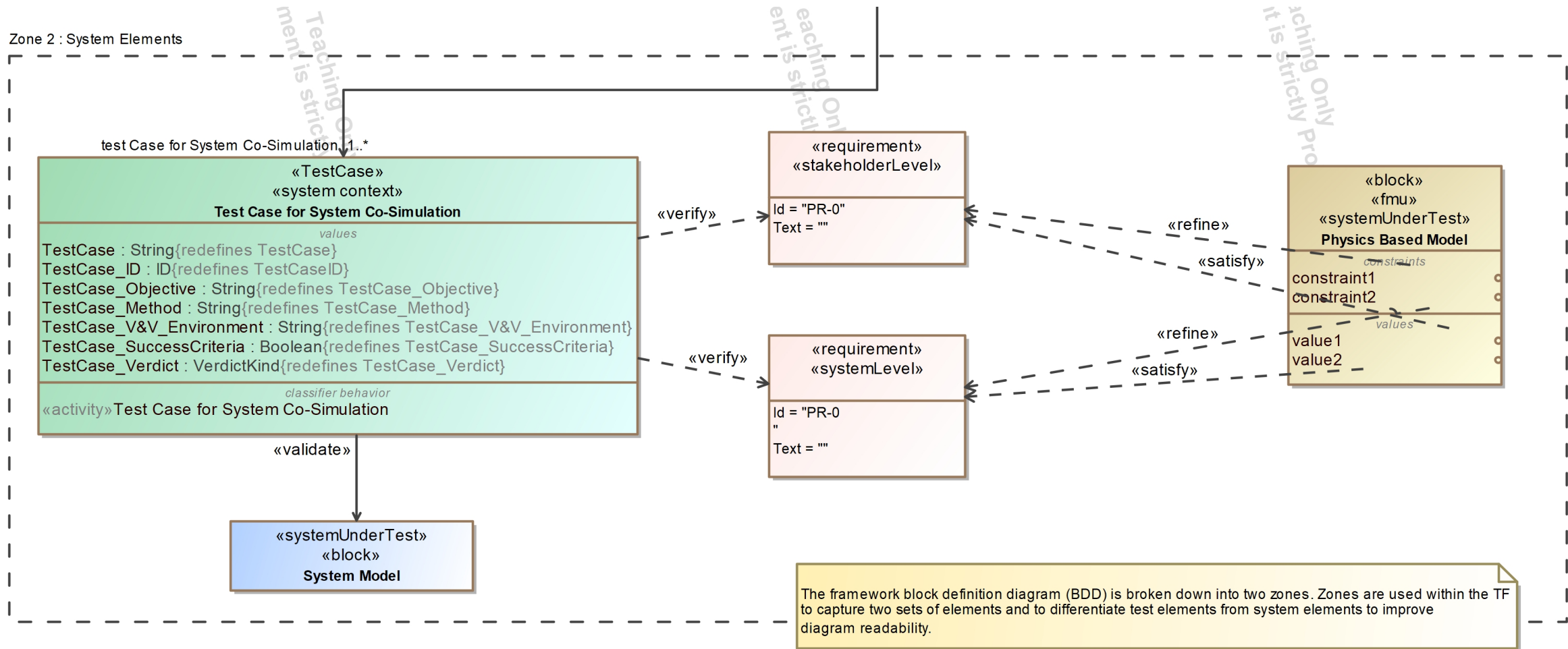
Design Validation by System Co-Simulation

Developed a V&V library of test frameworks that captures method
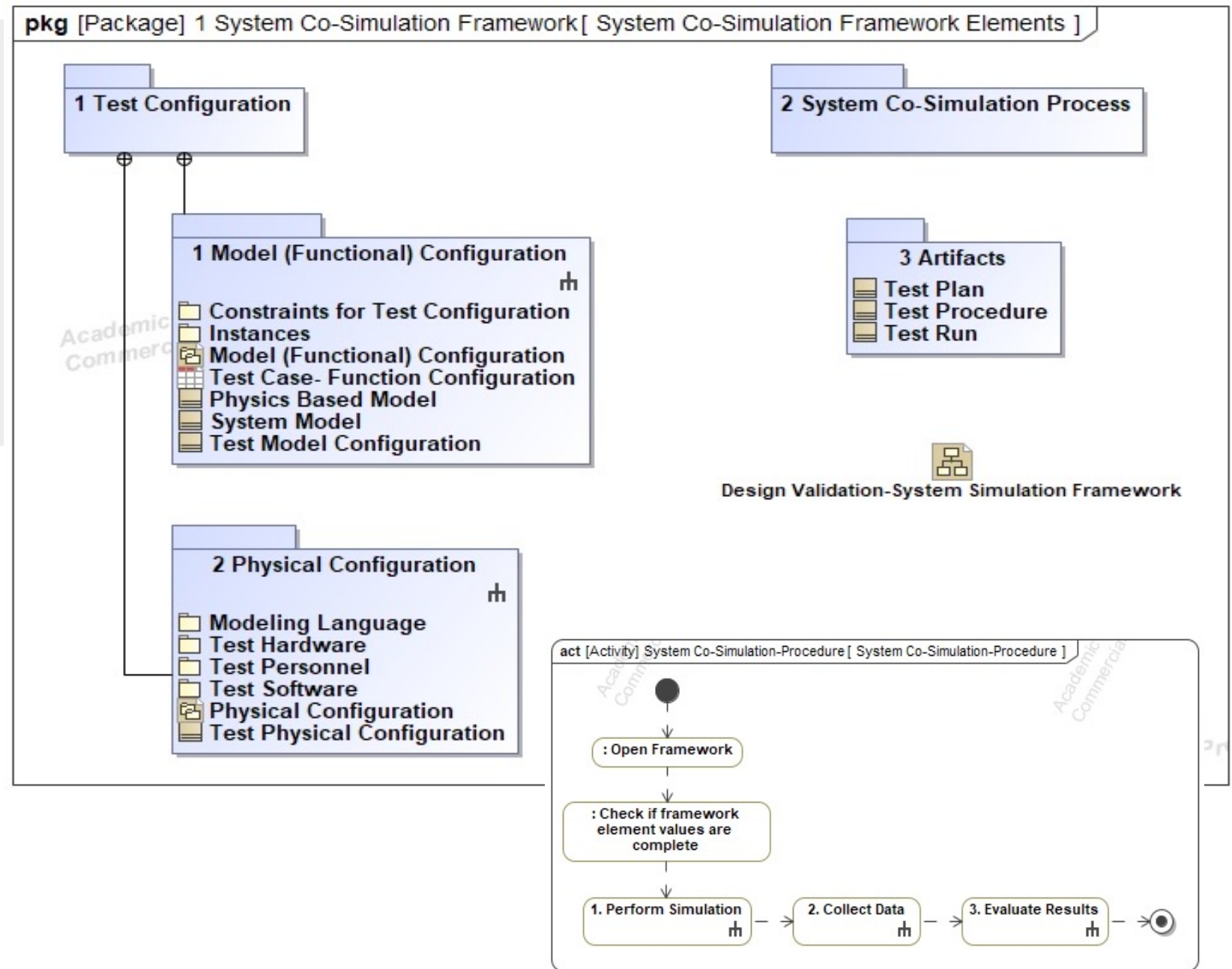
Proposed Verification & Validation Test Framework

MBSE Methodology for Design Validation by System Co-Simulation (DVD-SCS)

Test Plan

Test Configuration

System Co-Simulation Framework

Test Procedure

Test Case for X

Requirements

FMU Element

Test Run

System Under Test (SUT)

MBSE Methodology for Design Validation by Physical Test (Test Rig)

Test Plan

Test Configuration

Physical Test Framework

Test Procedure

Test Case for X

Requirements

Test Run

Modified System or SubSytem SysML Model <<SUT>>

Test Rig SysML Model <<SUT>>

7/20/2023

18
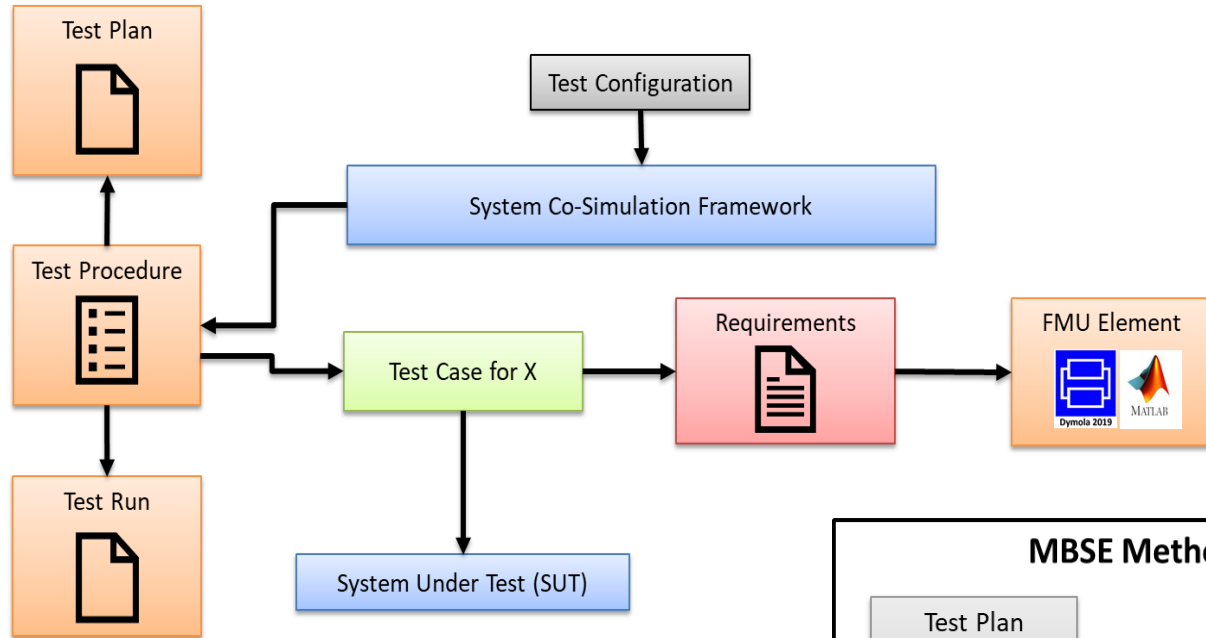
Current Approach & Proposed V&V TF

Current Document Based Approach for Test Requests

| Requirements (System Requirement) | Measure of Performance (MOPs) | Analysis Method (Physics-Based Model) | Physical Test Method (Test Rig) | Test Plan & Test Procedure |

Design Constraints

MBSE Methodology for Design Validation by System Co-Simulation (DVD-SCS)

MBSE Methodology for Design Validation by Physical Test (Test Rig)

Semi- Automation of Test Case Results

Block Definition Diagram (BBD) for Test Configuration

Parametric Diagram for Test Configuration

Block Definition Diagram (BBD) for Instance Configuration and Relation to Requirement

System Model Configuration (SMC) using Test ArtifactTable

Semi- Automation of Test Case Results

Block Definition Diagram (BBD) for Test Configuration

21

Semi- Automation of Test Case Results

Parametric Diagram for Test Configuration

7/20/2023
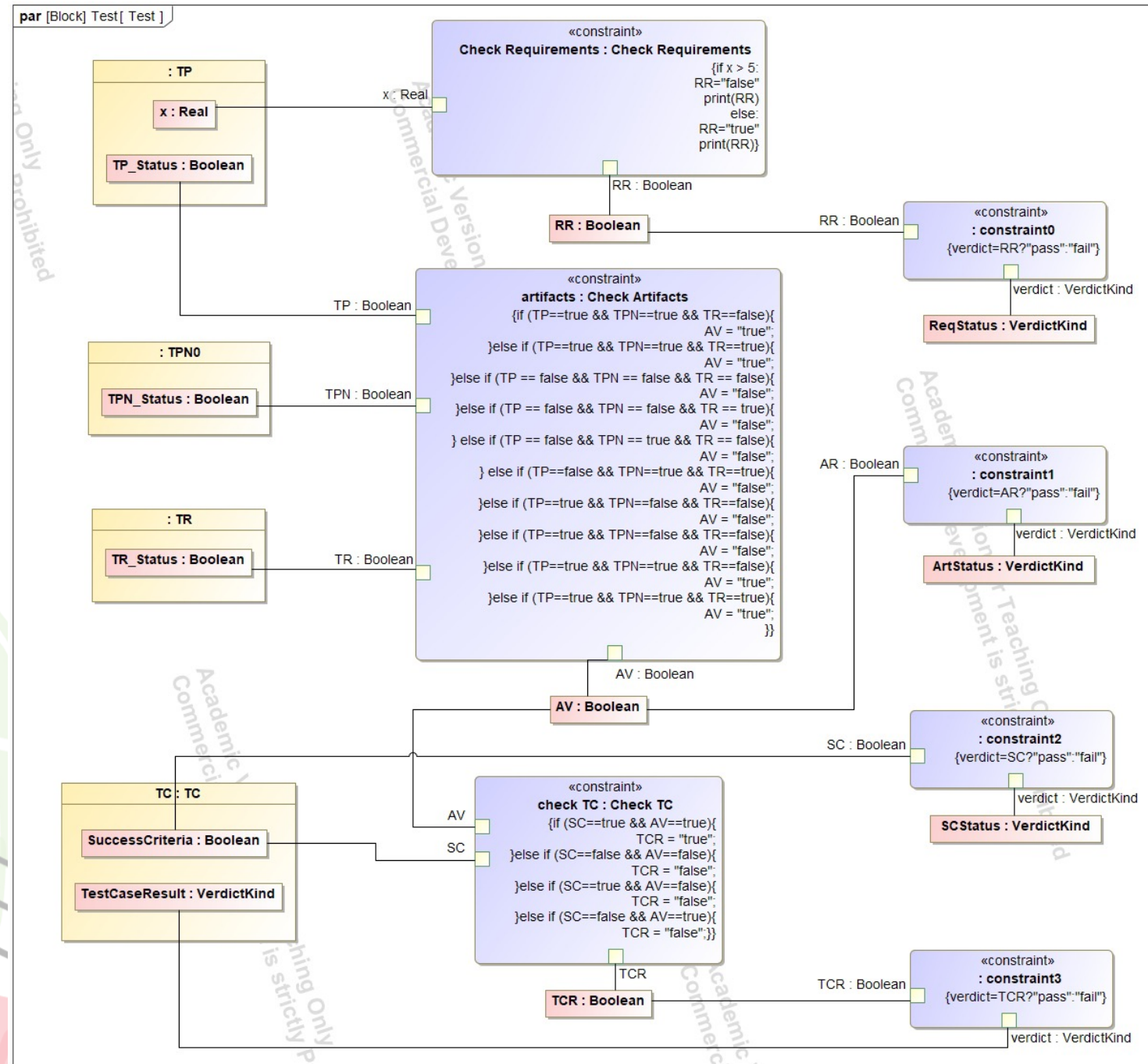
Semi- Automation of Test Case Results

Block Definition Diagram (BBD) shows four instances of one test type
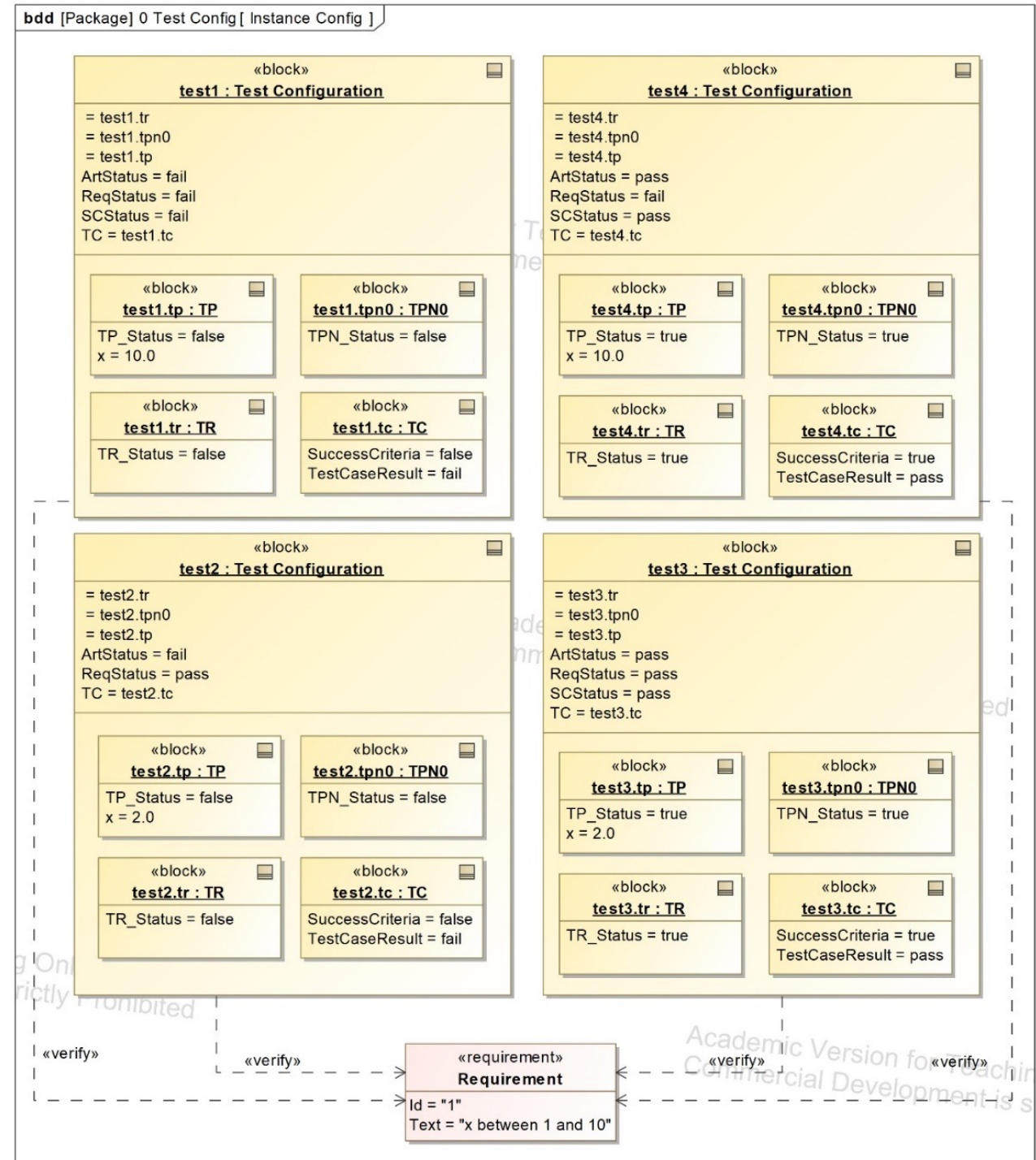
System Model Configuration (SMC) using a Test Artifact (TA) Validation Table

Criteria

Classifier: Test Configuration ... Scope (optional): 0 Test Config {x} ... Filter: ▽

Verification Status: ☐ Pass ☐ Fail ...

#	△ Name	.TPN_Status : Boolean	.TR_Status : Boolean	.TP_Status : Boolean	ArtStatus : VerdictKind	.x : Real	ReqStatus : VerdictKind	TC.SuccessCriteria : Boolean	TC.TestCaseResult : VerdictKind	Verifies
1	test1	☐ false	☐ false	☐ false	fail	10	fail	☐ false	fail	R 1 Requirement
2	test2	☐ false	☐ false	☐ false	fail	2	pass	☐ false	fail	R 1 Requirement
3	test3	☑ true	☑ true	☑ true	pass	2	pass	☑ true	pass	R 1 Requirement
4	test4	☑ true	☑ true	☑ true	pass	10	fail	☑ true	pass	R 1 Requirement

Requirement and Test Artifact (TA) Validation Table

- System Model Configuration or System Function Configuration
  - Table captures all configurations of model as separate instances.
  - Save run of Test Procedure as an instance

- During simulation execution requirement table is updated with verdict output

Test Artifact Validation Table - Elements

Test Case Instance	Test Artifacts (TPN, TR, TP, SC)	Test Case Result	Requirement

Requirement Validation Table - Elements

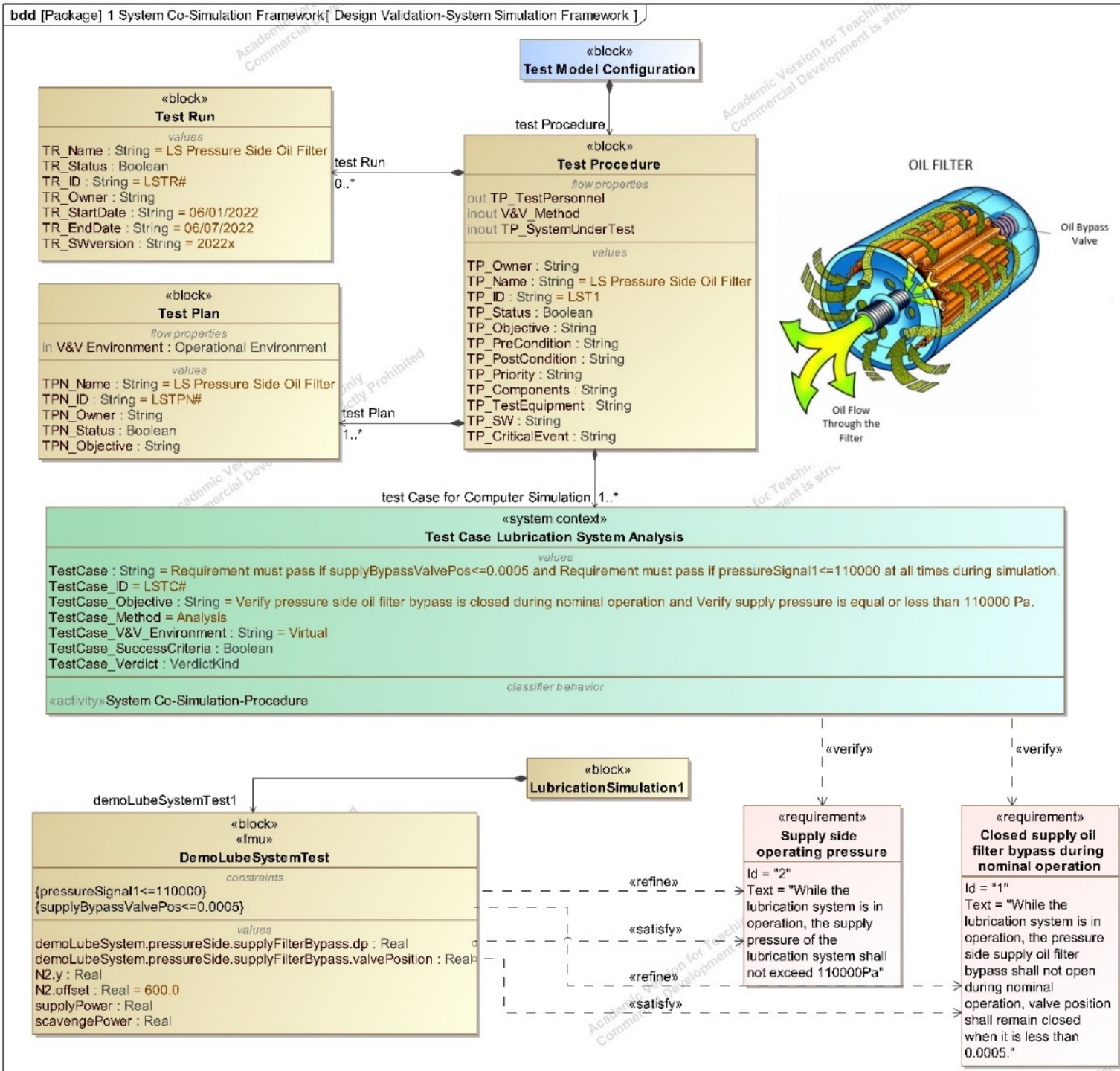| Requirement	Property (FMU)	Value (FMU)	Bounds	Margin

Use Case: Lubrication System

- TF for Design Validation by System Co-Simulation Applied to Lubrication System
- Design validation by analysis
- Requirements are checked against constraints and value from lubrication system analytical results (FMU box)
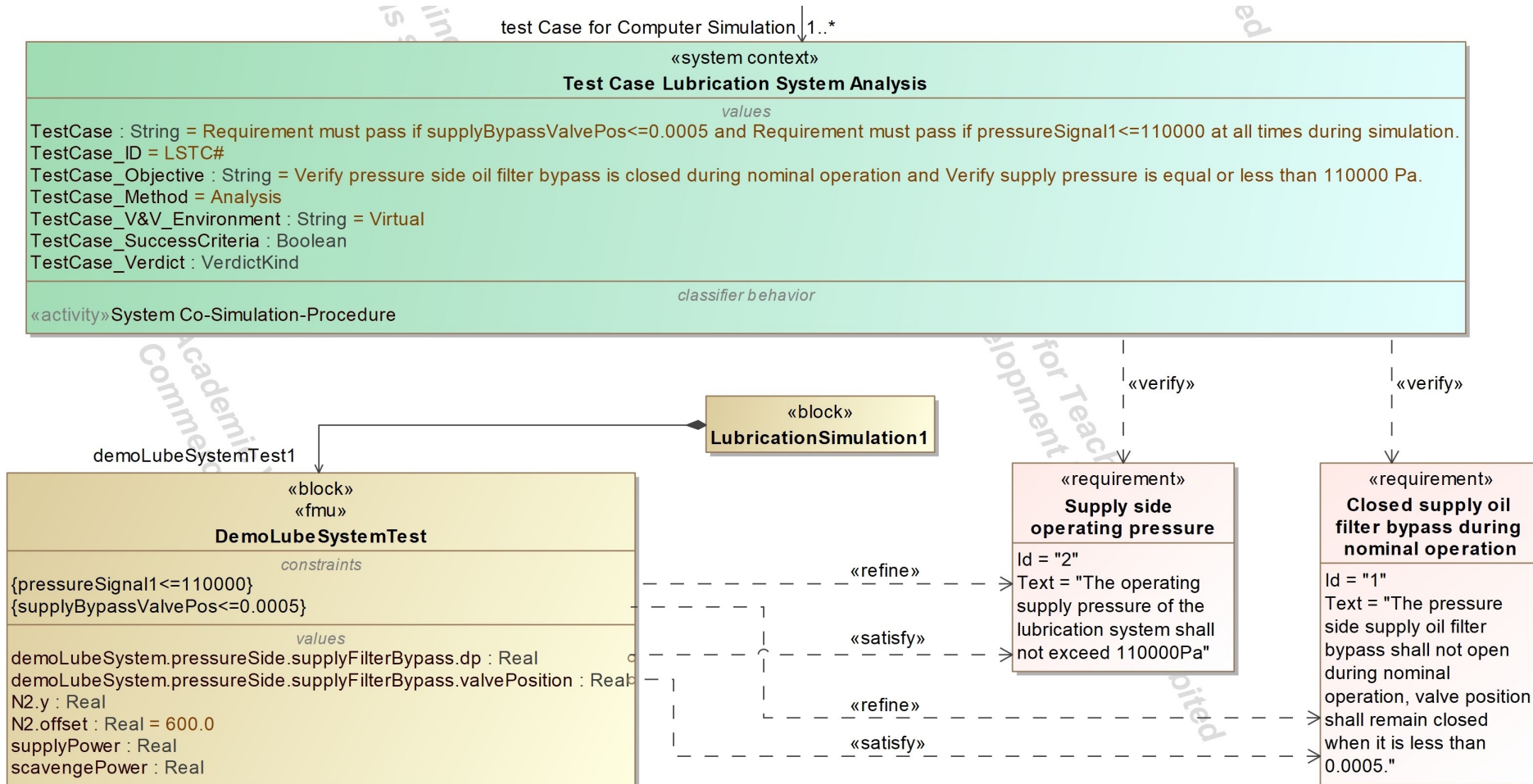
This page does not contain any export regulated technical data.
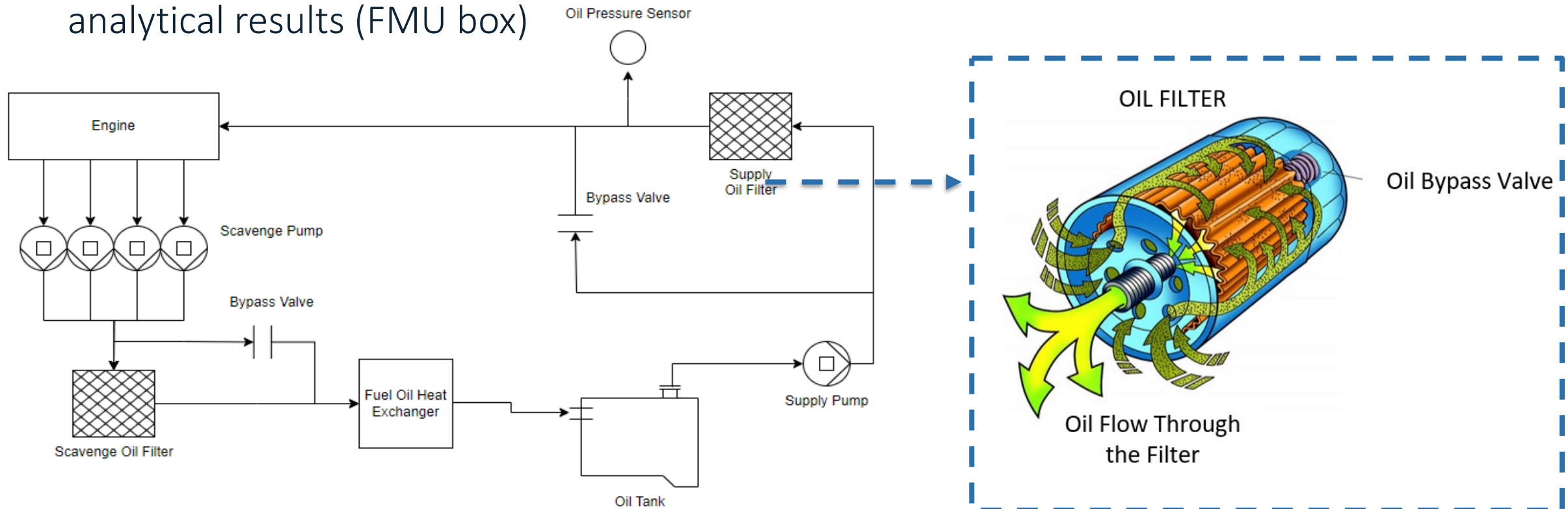
Applying V&V Test Framework to the Lubrication System Descriptive Model

7/20/2023

Applying V&V Test Framework to the Lubrication System Descriptive Model

test Case for Computer Simulation 1..*

«system context»
Test Case Lubrication System Analysis

values

TestCase : String = Requirement must pass if supplyBypassValvePos<=0.0005 and Requirement must pass if pressureSignal1<=110000 at all times during simulation.
TestCase_ID = LSTC#
TestCase_Objective : String = Verify pressure side oil filter bypass is closed during nominal operation and Verify supply pressure is equal or less than 110000 Pa.
TestCase_Method = Analysis
TestCase_V&V_Environment : String = Virtual
TestCase_SuccessCriteria : Boolean
TestCase_Verdict : VerdictKind

classifier behavior

«activity»System Co-Simulation-Procedure

«block»
LubricationSimulation1

demoLubeSystemTest1

«block»
«fmu»
DemoLubeSystemTest

constraints

{pressureSignal1<=110000}
{supplyBypassValvePos<=0.0005}

values

demoLubeSystem.pressureSide.supplyFilterBypass.dp : Real
demoLubeSystem.pressureSide.supplyFilterBypass.valvePosition : Real
N2.y : Real
N2.offset : Real = 600.0
supplyPower : Real
scavengePower : Real

«verify»

«verify»

«refine»

«satisfy»

«refine»

«satisfy»

«requirement»
**Supply side
operating pressure**

Id = "2"
Text = "The operating
supply pressure of the
lubrication system shall
not exceed 110000Pa"

«requirement»
**Closed supply oil
filter bypass during
nominal operation**

Id = "1"
Text = "The pressure
side supply oil filter
bypass shall not open
during nominal
operation, valve position
shall remain closed
when it is less than
0.0005."

Requirement Validation Table for Lubrication System

- Test case execution generates test case pass or fail results

Requirement Verification: ☐ Pass ☐ Fail …

#	Name	Text	Property	Value	▽ Bounds	Margin
1	[R] 1 Closed supply oil filter bypass during nominal operation	The pressure side supply oil filter bypass shall not open during nominal operation, valve position shall remain closed when it is <u>less than 0.0005</u>.	demoLubeSystemTest1.demoL [V] ubeSystem.pressureSide.supply FilterBypass.valvePosition : Real	0.0066505098	<5.0E-4	-0.0061505098
2	[R] 2 Supply side operating pressure	The operating supply pressure of the lubrication system shall <u>not exceed 110000Pa</u>	demoLubeSystemTest1.demoL [V] ubeSystem.pressureSide.supply FilterBypass.dp : Real	110986.9903741054	<=110000	-986.9903741054

Summary of V&V Test Framework Method
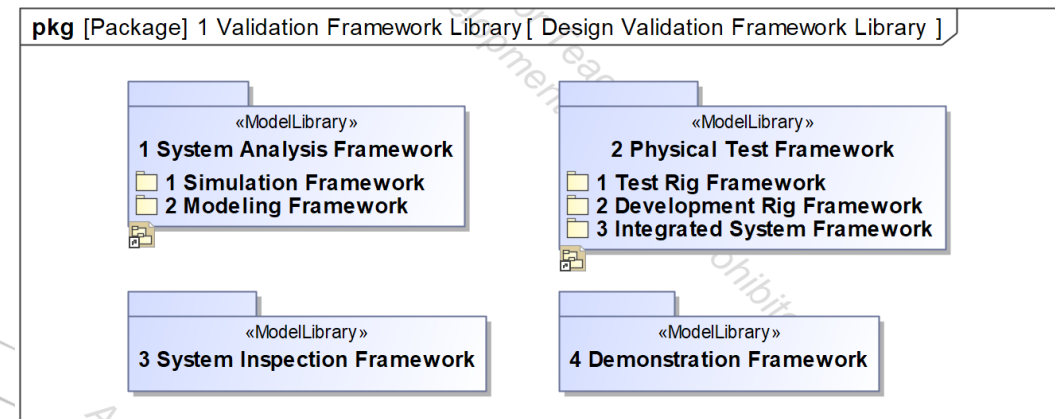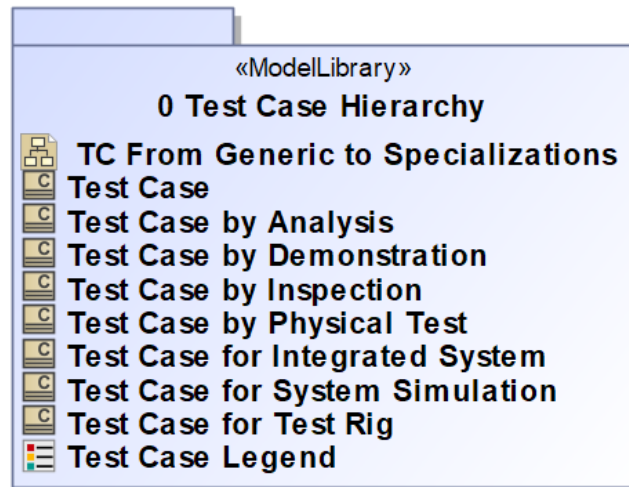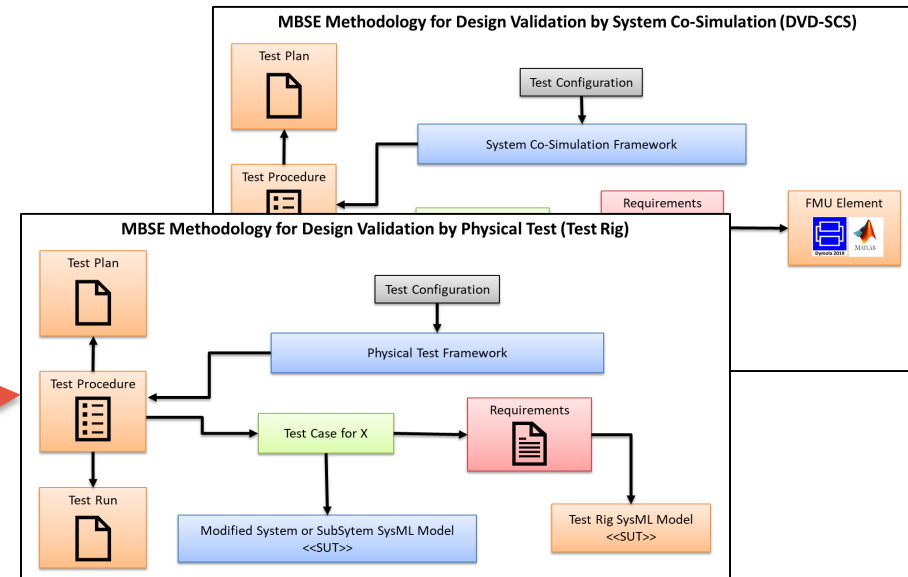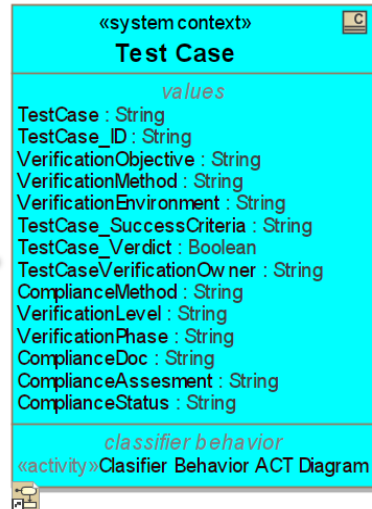
What is the value of the V&VTF?

- Test cases and patterns are defined in the systems model to improve V&V planning

- Explicit relation between test elements to manage design changes and their propagation

- Avoided customization with new profiles, this method was developed using current SysML elements

- A comprehensive approach to handle different test types and system abstraction levels.

- The V&VTF is demonstrated through the testing of a lubrication system, providing design validation through co-simulation.

UCONN | UNIVERSITY OF CONNECTICUT
PRATT & WHITNEY INSTITUTE FOR
ADVANCED SYSTEMS ENGINEERING

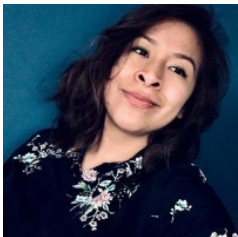**33rd** Annual **INCOSE**
international symposium

hybrid event

Honolulu, HI, USA
July 15 - 20, 2023

Amy Thompson, Ph.D.
Associate Professor-In-Residence, Systems Engineering
Associate Director, Pratt & Whitney Institute for Advanced Systems
Engineering
University of Connecticut
Email: amy.2.thompson@uconn.edu

Clara Ramirez
Doctoral Student, Mechanical Engineering
Graduate Research Assistant
University of Connecticut
Email: clara.ramirez@uconn.edu