



MathWorks Model-Based Systems Engineering

Practical use-cases → bridging the gap

Stephan van Beek

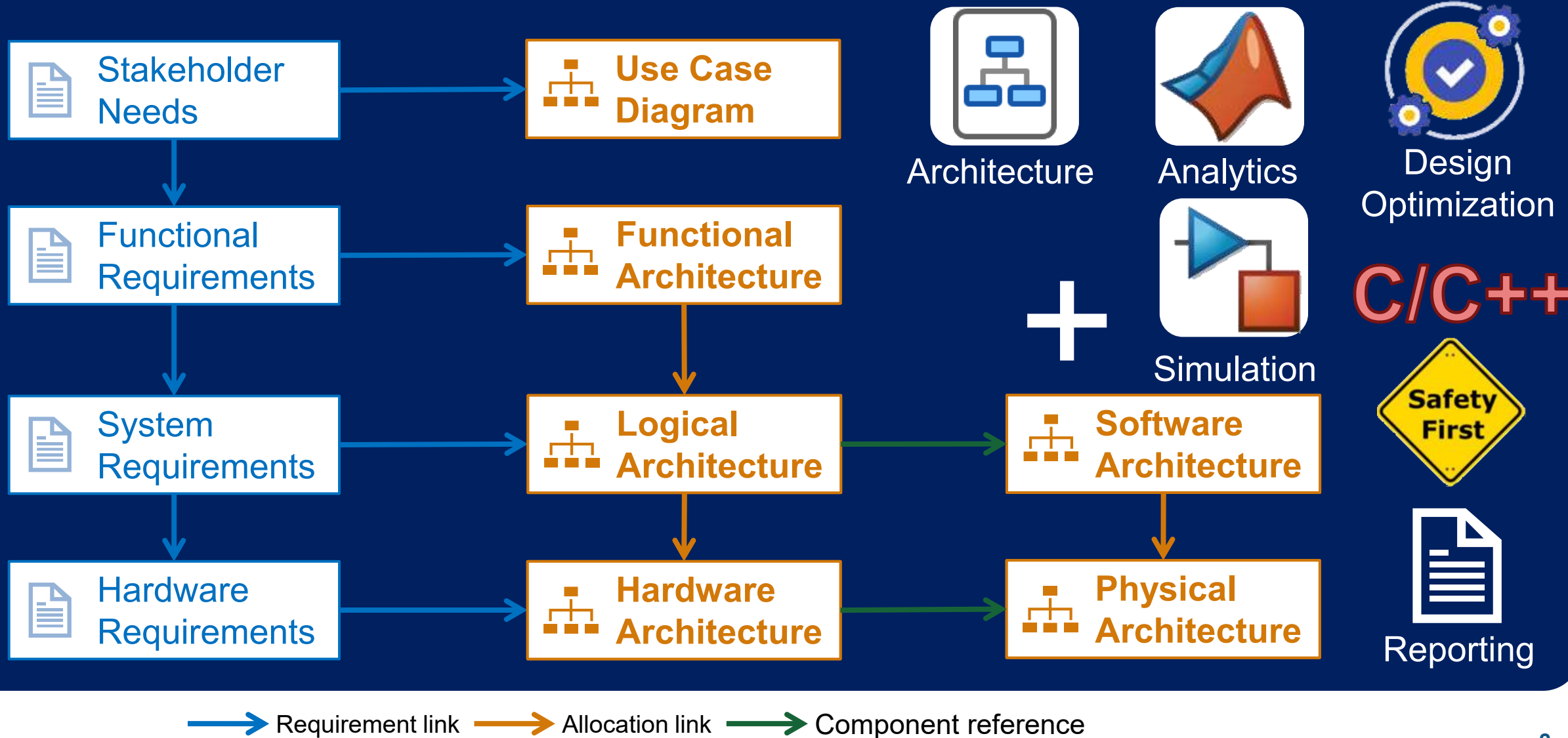
Principal Application Engineer
svanbeek@mathworks.com

Marco Bimbi

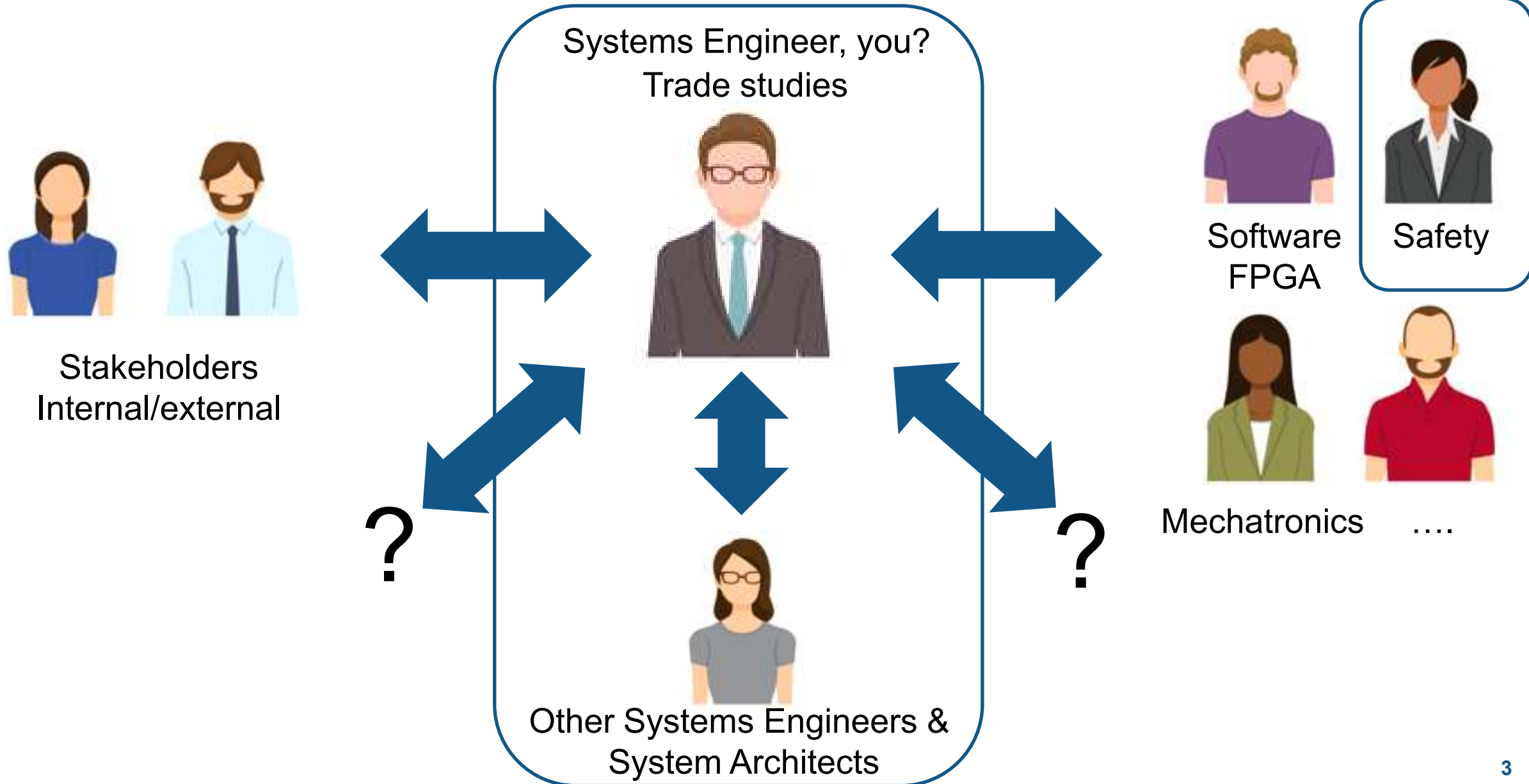
Principal Application Engineer
mbimbi@mathworks.com



MathWorks Model-Based Systems Engineering



Practical use-cases in context



MathWorks Model-Based Systems Engineering, Trade Studies

Stakeholder Needs

Functional Requirements

System Requirements

Hardware Requirements

Functional Architecture

Logical Architecture

Hardware Architecture



Architecture



Analytics



Design Optimization



Simulation



Reporting

Requirement link
 Allocation link



Trade studies



Enabling MBSE with Simulation to perform System Analysis for Space-Based Solar-Power

Stephan van Beek

Principal Application Engineer
svanbeek@mathworks.com

Marco Bimbi

Principal Application Engineer
mbimbi@mathworks.com



Acknowledgement

This work has been done in collaboration with
Thales Alenia Space Italy:

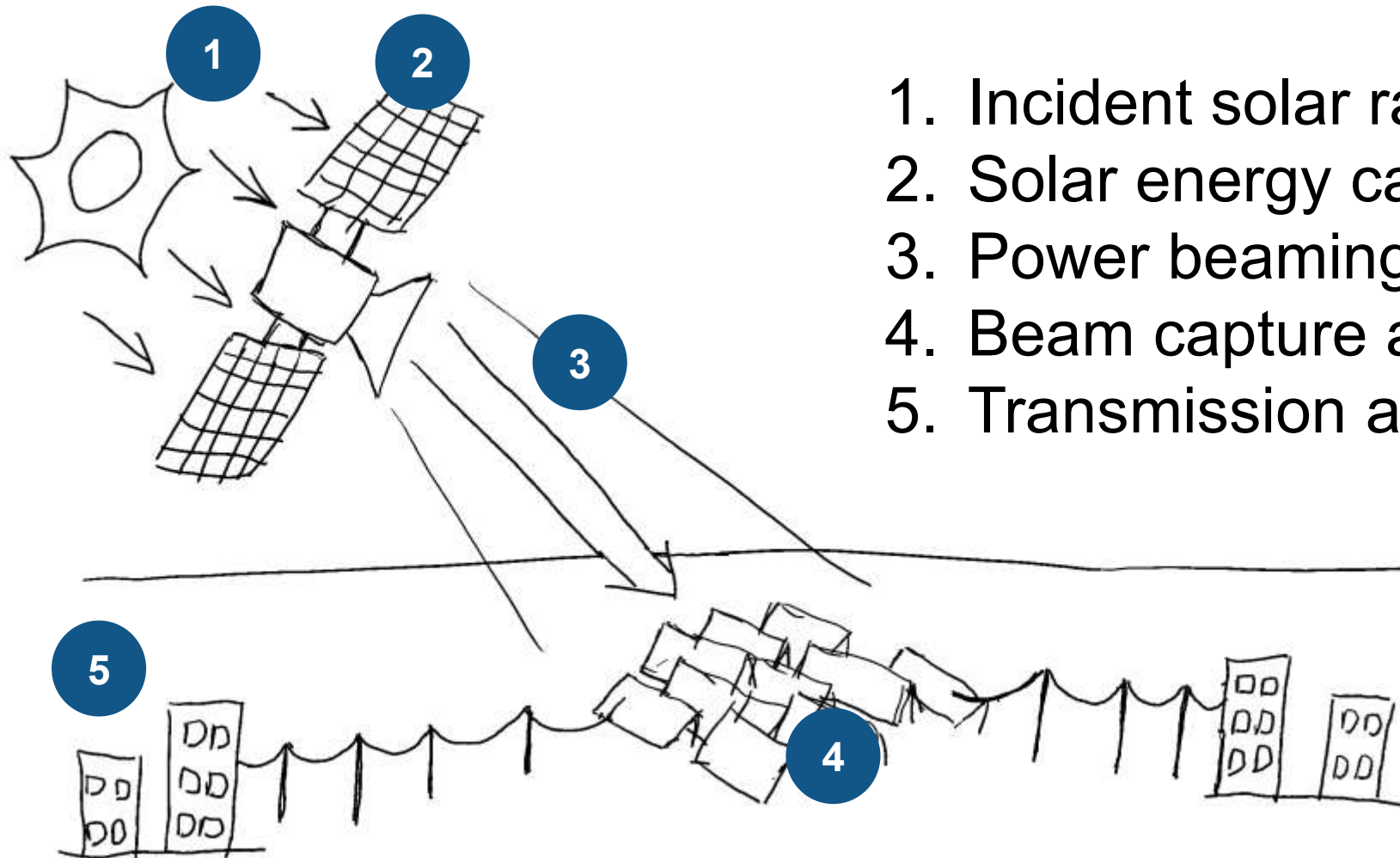
Serena Brizio, Lorenzo Guarino, Umberto Di Tommaso

Agenda

- What is Space-Based Solar-Power?
- What are the main challenges/needs?
- Solution
 - Process/methodology
 - Bridge between Capella and System Composer
 - Analysis Workflow
- Outcomes & Concluding Remarks
- Q&A

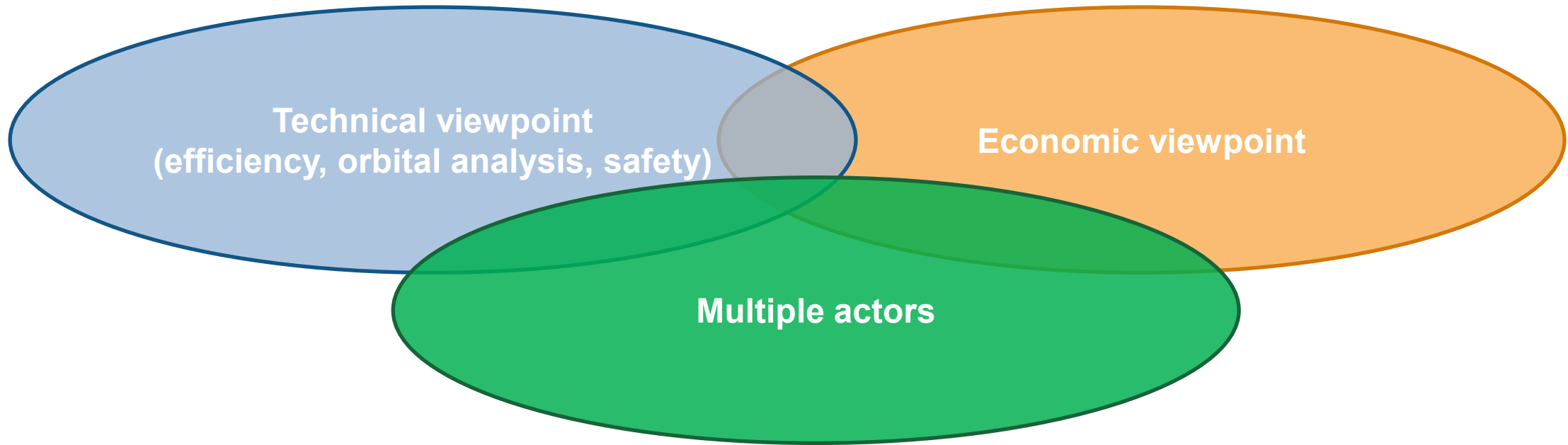
What is Space-Based Solar-Power (SBSP) about?

- Space-Based Solar Power involves harvesting sunlight from Earth orbit then beaming it down to the surface where it is needed.



1. Incident solar radiation
2. Solar energy capture and regulation
3. Power beaming
4. Beam capture and conversion
5. Transmission and distribution

What are the main challenges/needs?

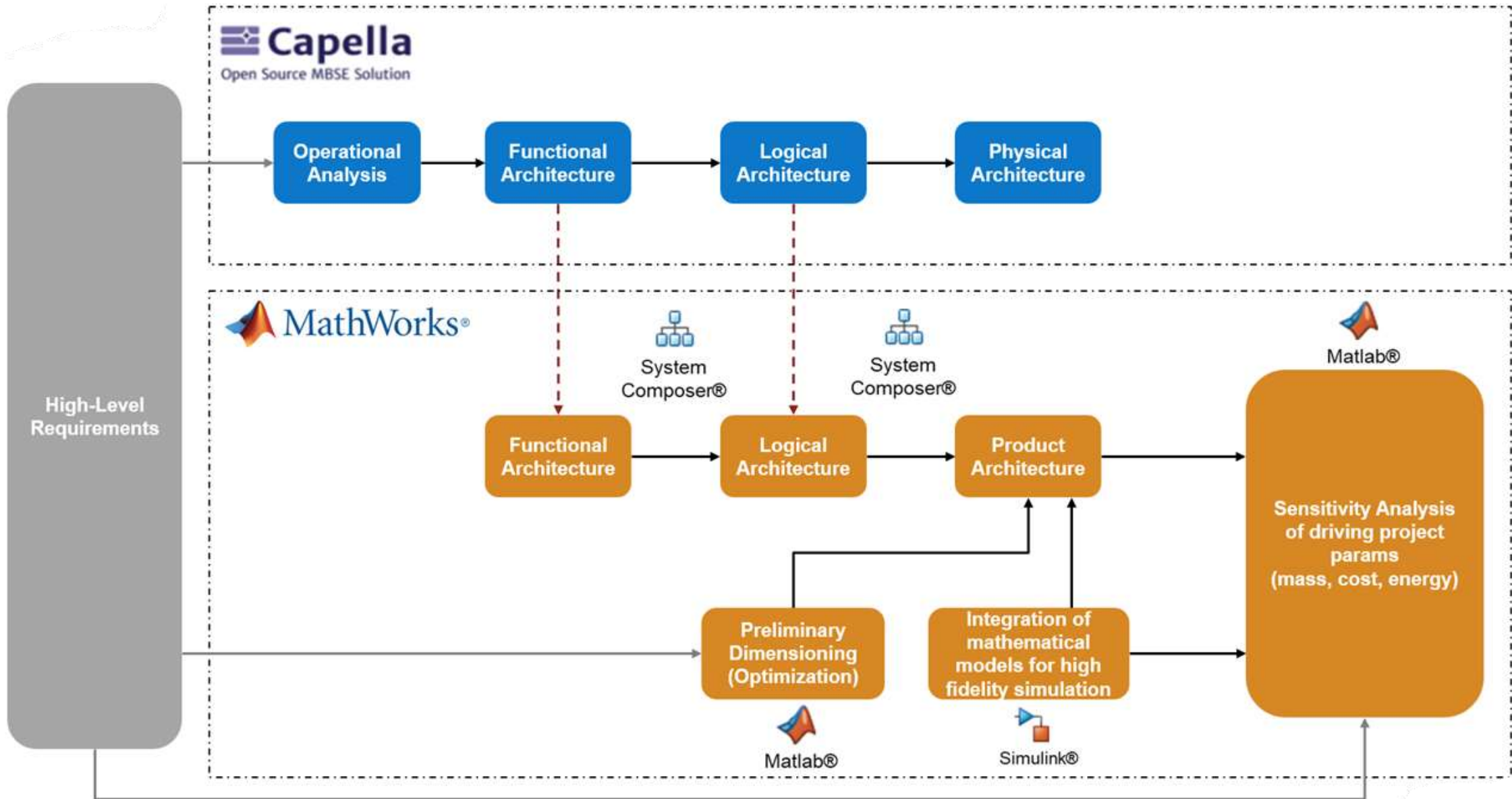


Static point of view



Dynamic point of view

Solution - Process/methodology



Solution - Bridge between Capella and System Composer

Satellite

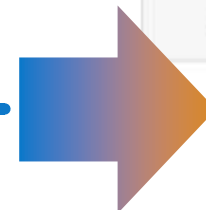
Capella Model

System Composer

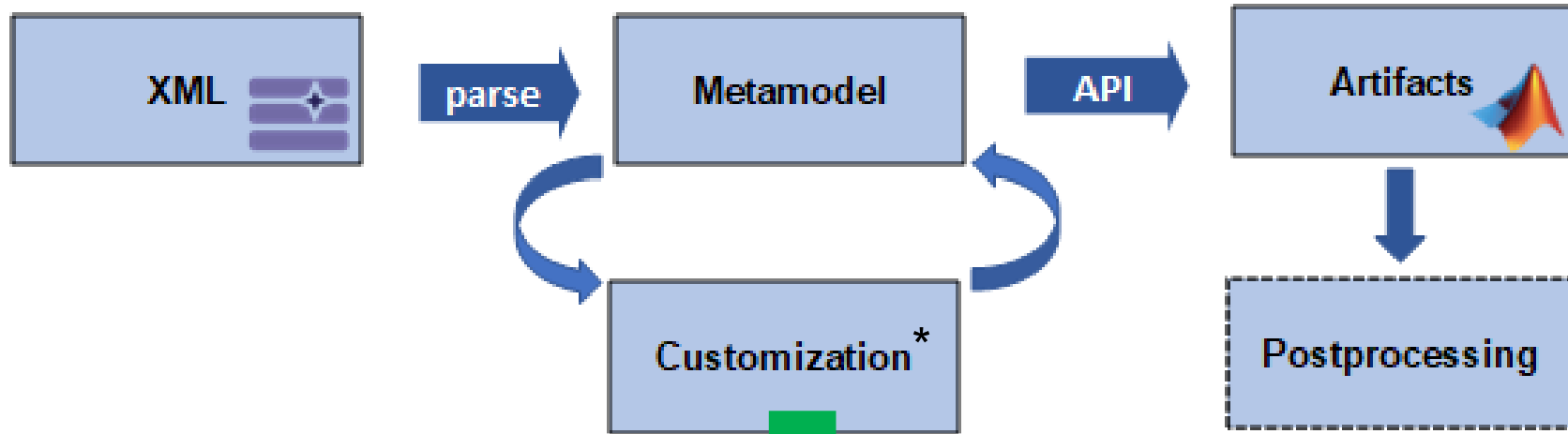
Product Architecture

Logical Architecture

Ground station



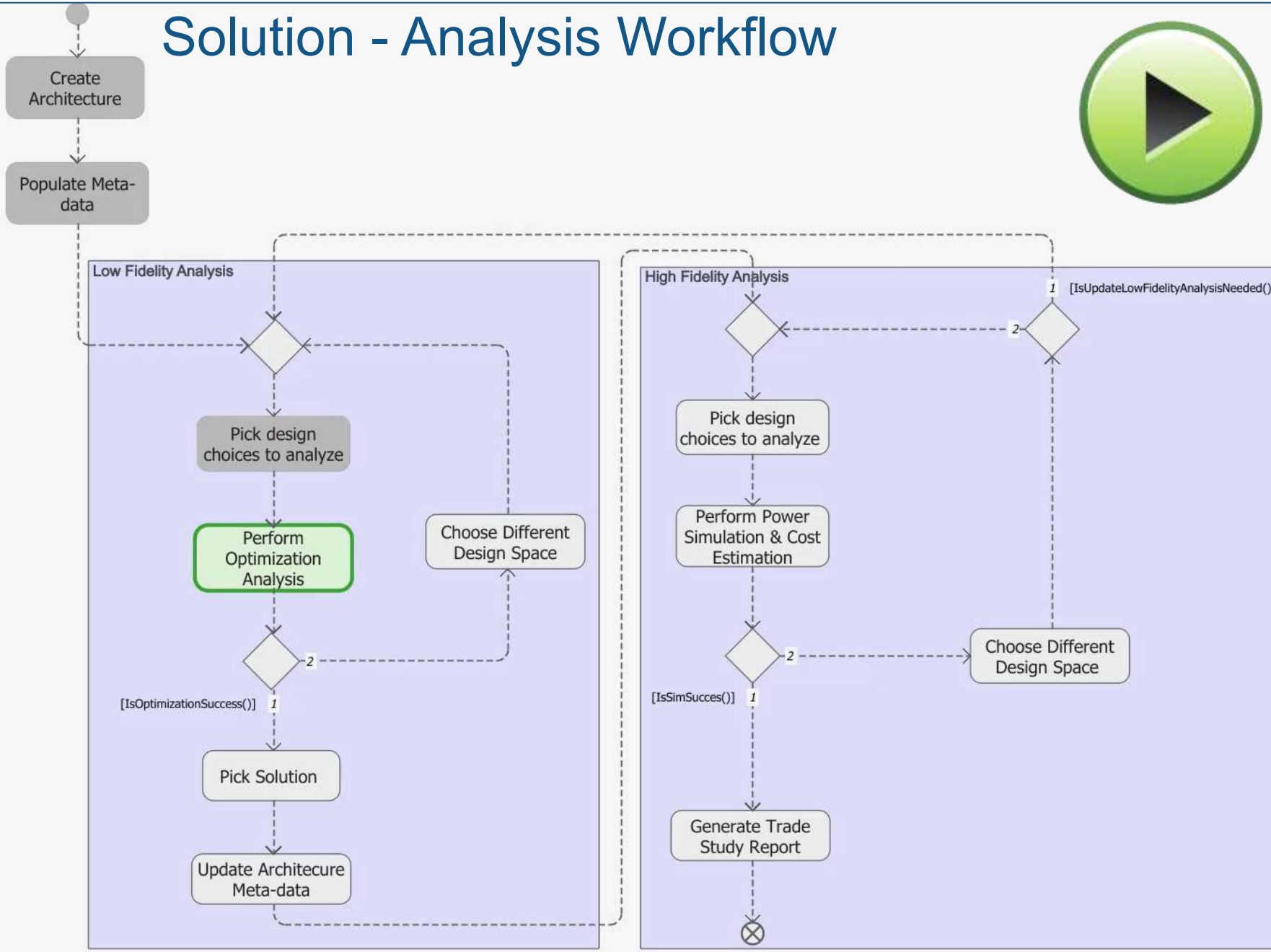
Solution - Bridge between Capella and System Composer



Capella →	→ System Composer
LogicalArchitecture	Architecture model
LogicalComponent	Architecture model
ComponentExchange	Root-level port with composite interface
FunctionalExchange	Connector
ExchangeItem	Interface item
Datatype	Interface item type
Class	Composite interface

* can be customized for other objects and mappings like requirements, profiles, etc.

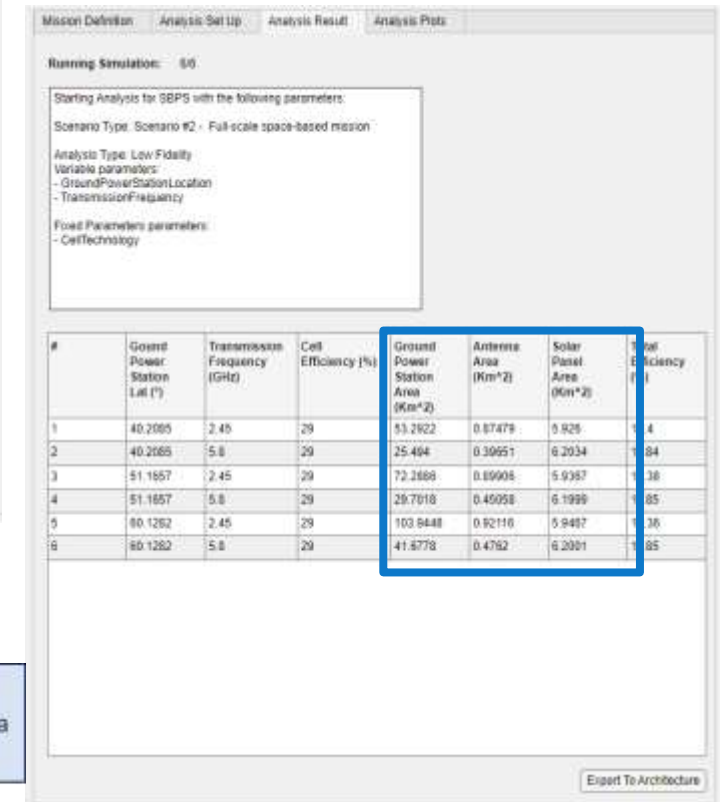
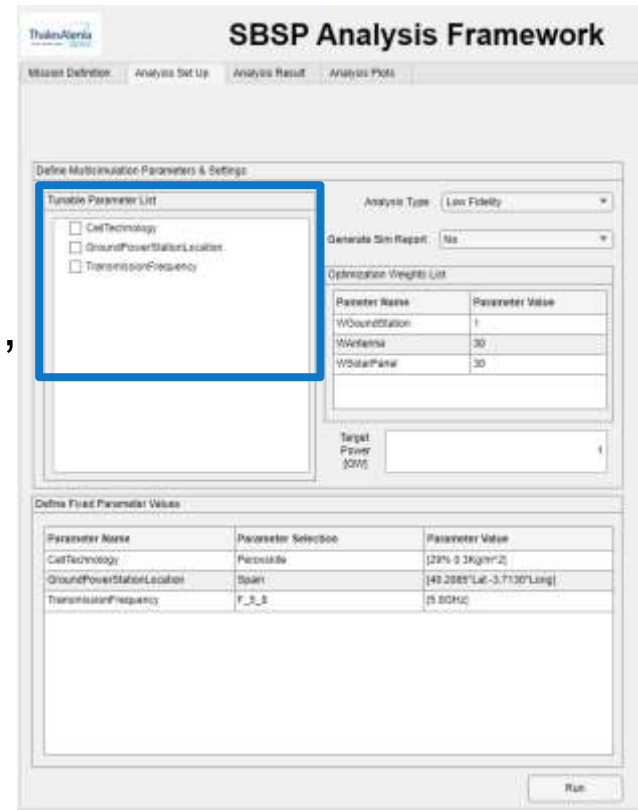
Solution - Analysis Workflow



Solution - Analysis Workflow

Low Fidelity Analysis

- Objective: Find optimal combination of the Photovoltaic (PV) area, antenna area, and GPS area
- Design Choices
 - (3) Cell technology
 - (3) Ground Station Location
 - (2) Transmission Frequency



$$\text{objective} = w1 \cdot x(1) + w2 \cdot x(2) + w3 \cdot x(3)$$



3 x 3 x 2 = 18 unique variant combinations

Weight factor 1

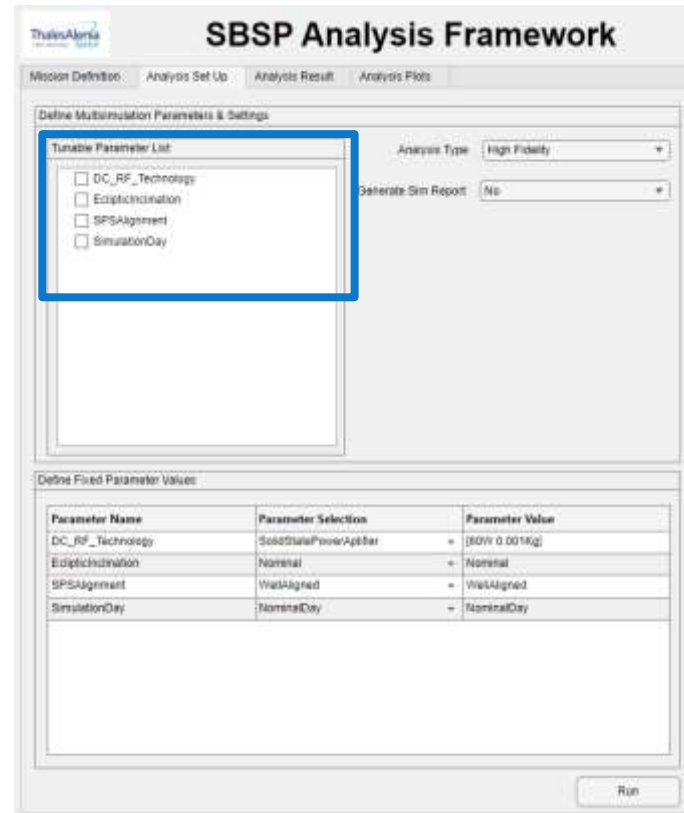
Weight factor 2

Weight factor 3

Solution - Analysis Workflow

High Fidelity Analysis

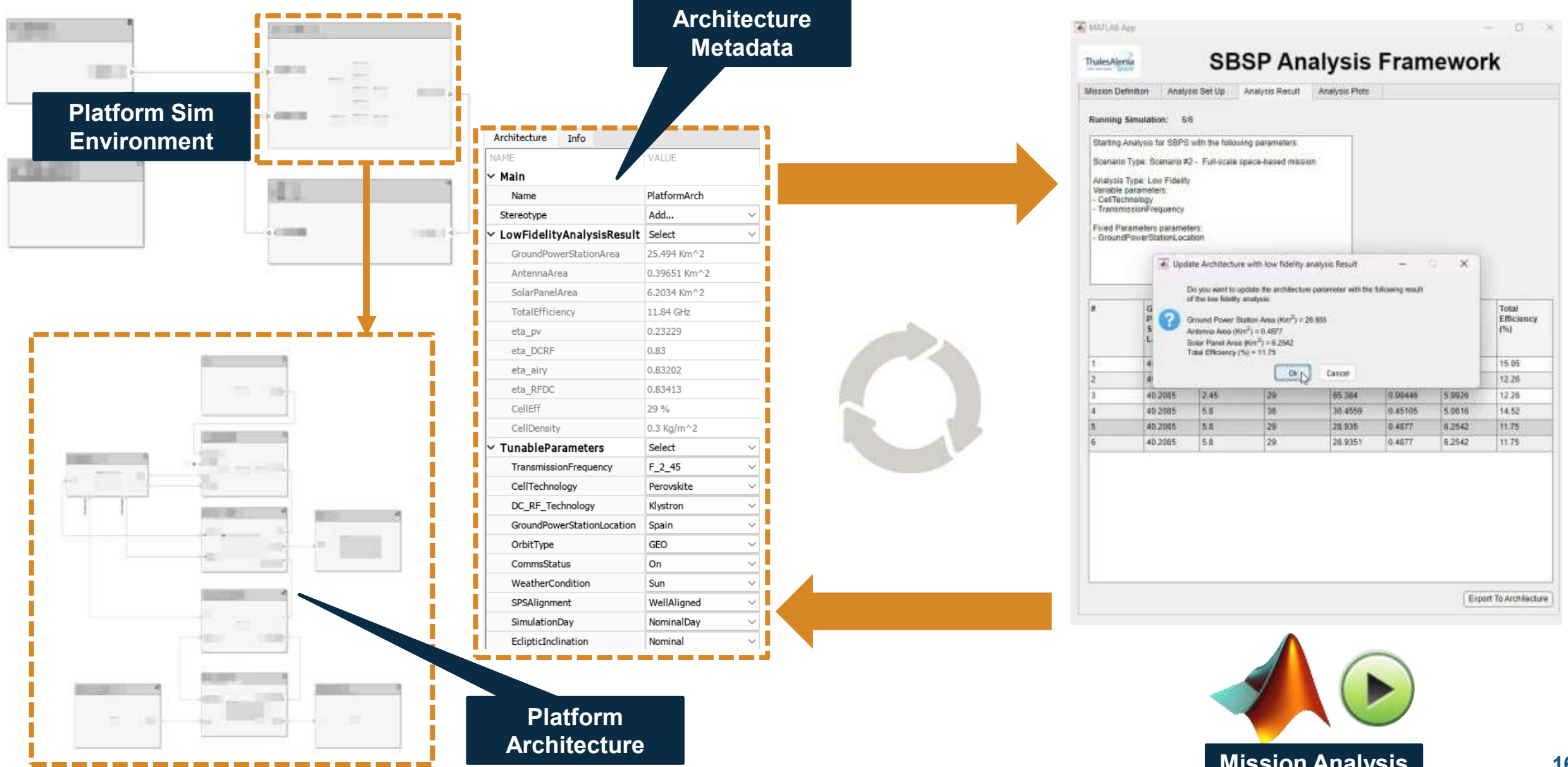
- Objective:
 - High-fidelity power simulations in various mission scenarios
 - Preliminary mass and cost estimation
- Design Choices
 - (3) DF-RF Technology
 - (2) Simulation Day
 - (2) Ecliptic inclination
 - (2) SPS Alignment



Ecliptic Inclination [-]	Average Transmission Power (MW)	Total Mass (T) & Total Launch(-)	Mission Cost (B\$) & LCOE (\$/MWh)	EROEI (-) & Energy Paybacktime (days)
Nominal	<ul style="list-style-type: none"> • @PVA=2064 • @PMainBus=1979 • @On-board Antenna=1577 • @GPS=1051 • @Grid=993 	<ul style="list-style-type: none"> • tot_mass=6591 • tot_launch=106 	<ul style="list-style-type: none"> • miss_cost=14 • LCOE=191 	<ul style="list-style-type: none"> • EROEI=42 • EPBT=219
Nominal	<ul style="list-style-type: none"> • @PVA=1883 • @PMainBus=1805 • @On-board Antenna=1439 • @GPS=959 • @Grid=993 	<ul style="list-style-type: none"> • tot_mass=6591 • tot_launch=106 	<ul style="list-style-type: none"> • miss_cost=14 • LCOE=191 	<ul style="list-style-type: none"> • EROEI=42 • EPBT=219

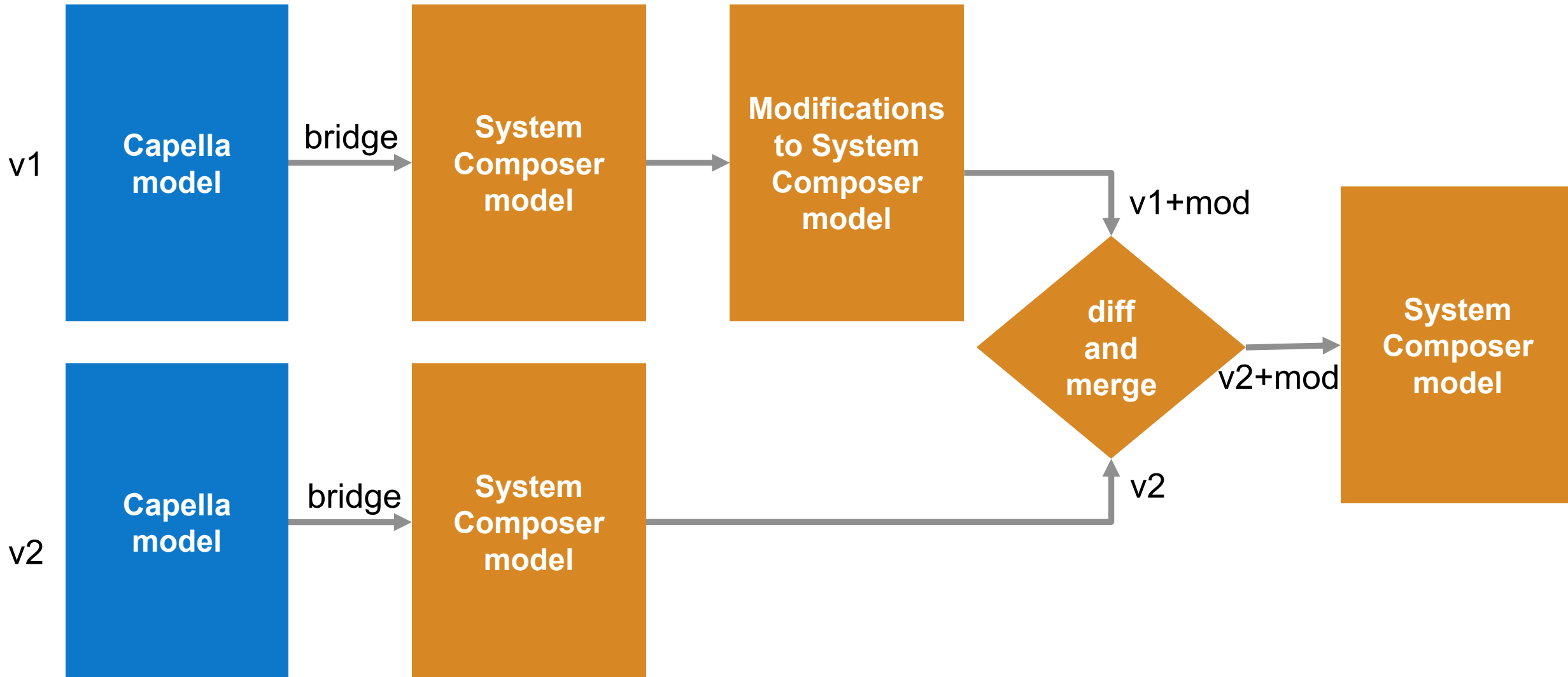
3 x 2 x 2 x 2 = 24 unique variant combinations

Solution - Analysis Workflow



Solution - Bridge between Capella and System Composer

Digital continuity



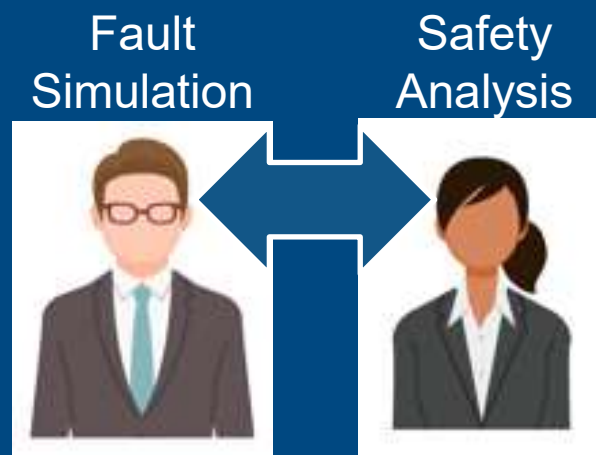
Outcomes

- **Comprehensive Understanding**, systematic analysis of the mission
- **Simulation of Complex Scenarios**, different solar conditions, orbit variations, etc.
- **Data-Driven Insights** using digital models
- **Efficiency Improvements**, optimize system components
- **Risk Mitigation**, identify challenges early
- **Iterative Design**, refine and improve the mission design over time
- **Cost and Resource Savings**, reduce the need for physical prototypes
- **Communication and Collaboration**, models facilitate effective communication

Concluding Remarks

- **Expandability:**
This framework (Bridge and Analysis) is designed to be expandable to the next phases of this study and other missions/projects.

- **Cross-tool operability:**
This framework demonstrates operability between Capella and System Composer and other MBSE tools .



Safety Analysis as an extension of MBSE

Stephan van Beek

Principal Application Engineer
svanbeek@mathworks.com

Marco Bimbi

Principal Application Engineer
mbimbi@mathworks.com



MathWorks Model-Based Systems Engineering, **Safety Engineering**

Stakeholder Needs

Functional Requirements

System Requirements

Hardware Requirements

Functional Architecture

Logical Architecture

Hardware Architecture



Architecture



Analytics



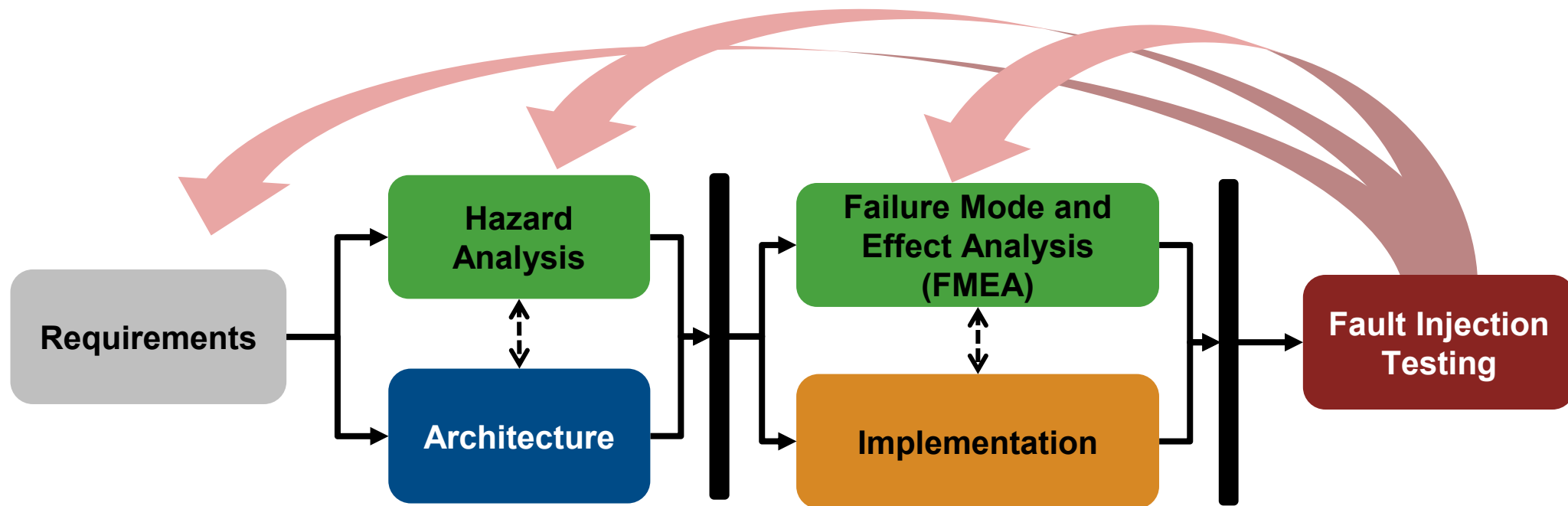
Simulation



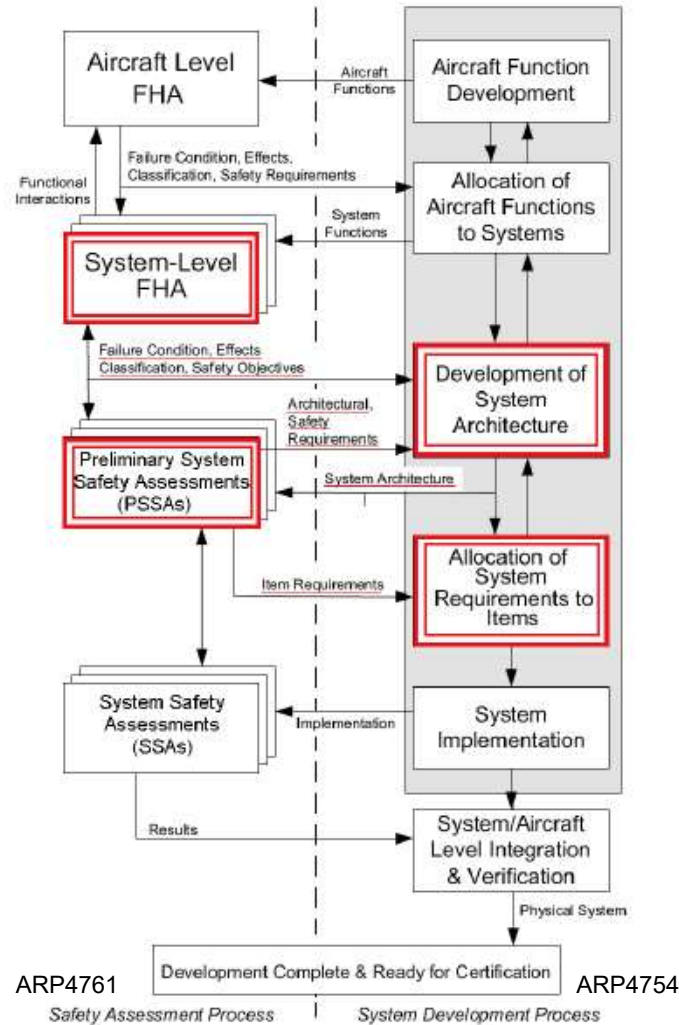
Reporting

Requirement link Allocation link

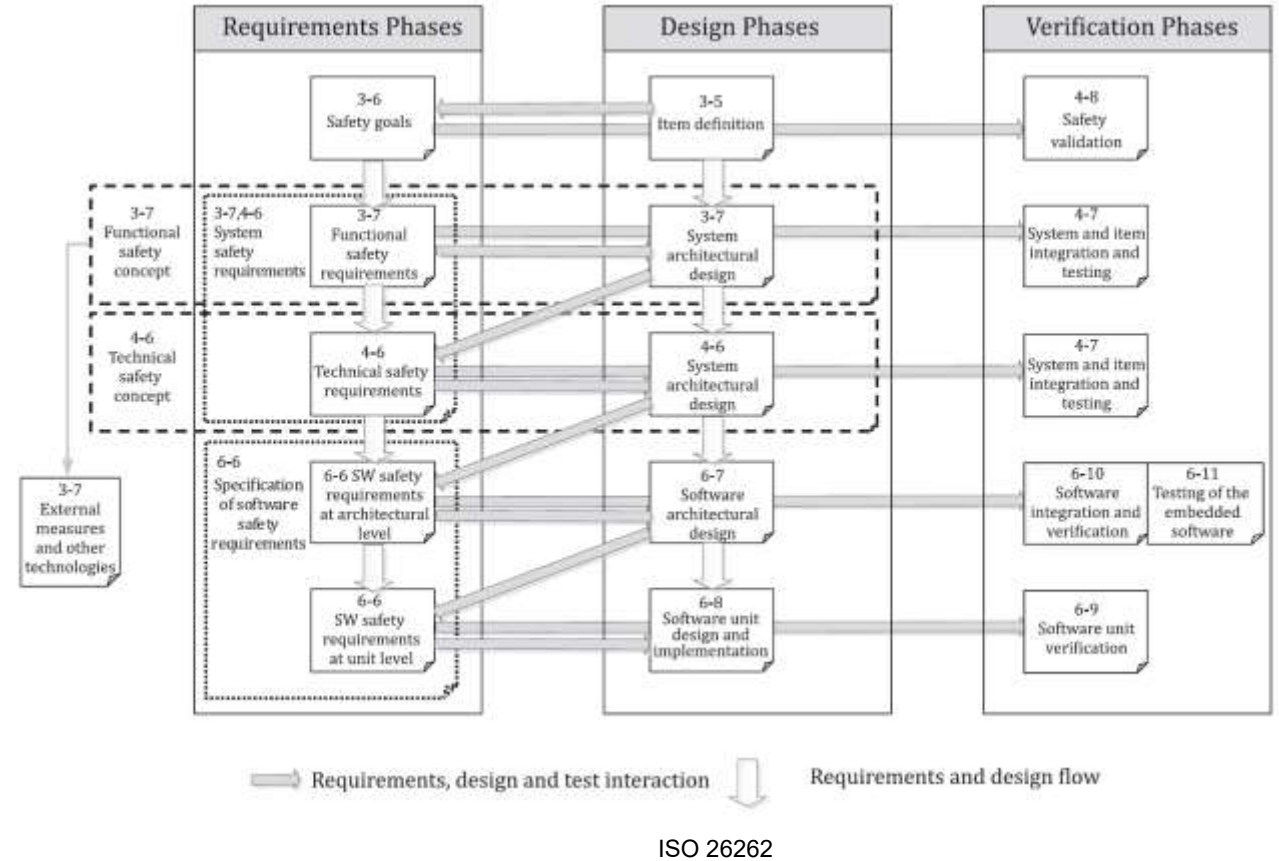
Safety Analysis, Detection / Mitigation and Verification are key activities in the design of engineered systems



Safety & Design are parallel activities – Standards Example



Aero Standards



Auto Standards

How Safety Analysis Is Done Today

ID	Function Name	Function
1	Function 1	Model/Pa
2	Function 2	Model/Pa
3	Function 3	Model/Pa
4	Function 4	Model/Pa

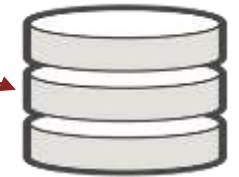
Script Type: Create Script

123

```

001' Oracle Create Table Script
002Function TableHeader(Table, Options)
003    TableHeader = "----" & vbLf & "----"
004    TableHeader = TableHeader & Object
005    TableHeader = TableHeader & vbLf &
006End Function
007
008Function nested_table_col_properties(C
009    nested_table_col_properties = ""
010    Dim Columns
011    Dim Column
012    Dim ColType
013    Dim ColTypeType
014    Dim Clause
015    Set Columns = Table.Children("Col
016    Dim ColumnTable
017    For Each Column In Columns
018        Set ColType = Column.Property
019        ColTypeType = ColType.Property
020        If ColTypeType = "Object" Then
021            Clause = nested table col
    
```

	Local Effect	System Effect	Derived Req
/Monitor	Loss of Protection	Loss of shutdown	ModuleName:#1
/Monitor1	Loss of Redundacy	None	
/Monitor2	None	None	
/Monitor1	Loss of Control	Control	ModuleName:#4



Requirements

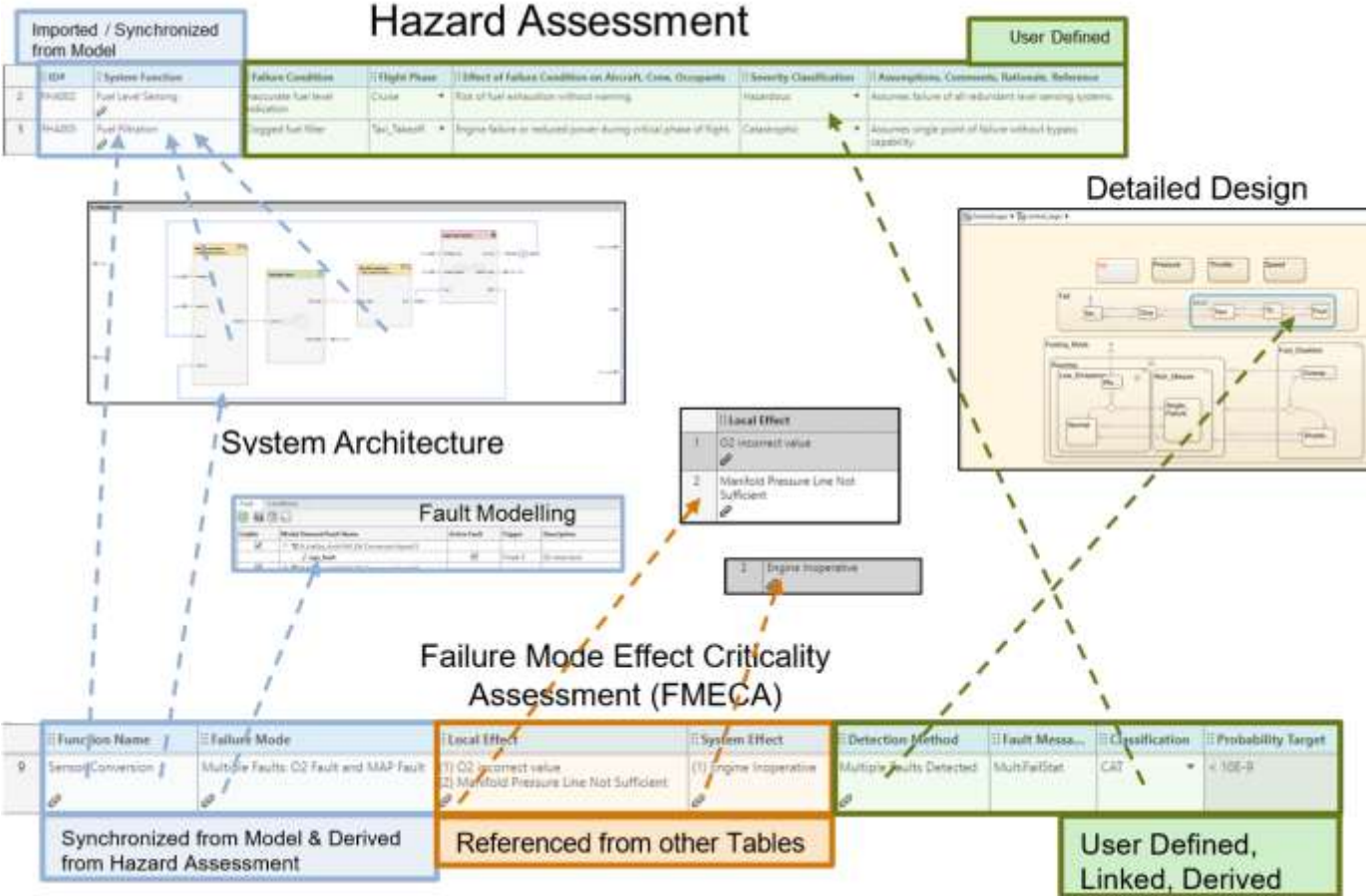
Non Model-Based Safety Analysis is ...

- ... **decoupled** from design work
- ... **complex and complicated**
- ... **error-prone**

Architect

C

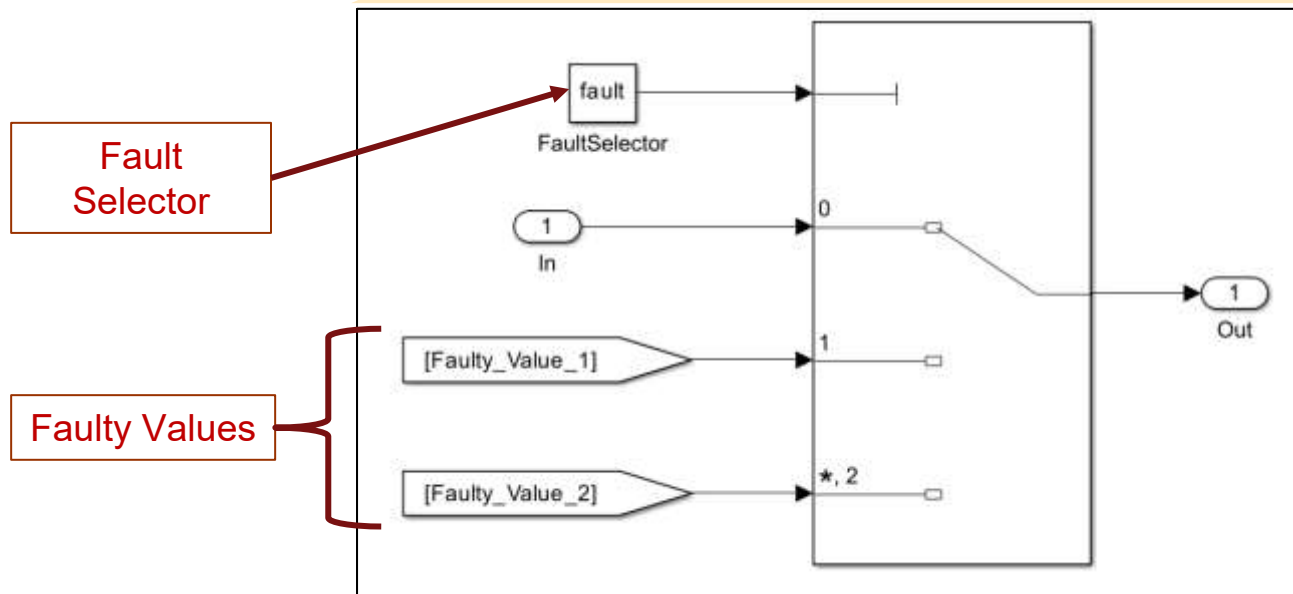
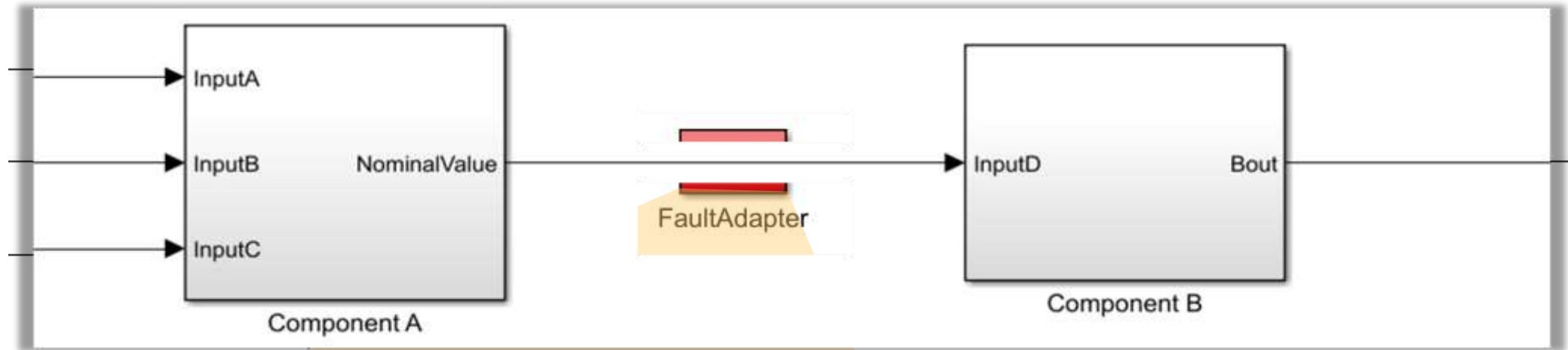
Why Model-Based Safety Analysis is the way to go



Model-Based Safety Analysis is ...

- ... **fully integrated** with design
- ... **fully traceable** (changes etc.)
- ... **consistent & validated**
- Synergy: fault modeling, FTA, tests

How is fault injection done today



- Modifies the design
- Only simple faults
- Difficult to analyze effects
- How do faults relate to hazards?

Model faults without modifying Design

Design Model

Fault Model

Property Inspector

▼ Faulted Model Element

Enable

Select fault to view:

—

SIMULATION DEBUG MODELING FORMAT APPS

Stop Time: inf

Normal

Fast Restart

Step Back Run Step Forward Stop REVIEW R

SIMULATE

ft_fuelsys_IntModel_FaultModel/ego_fault

ft_fuelsys_IntModel_FaultModel

ego_fault

Fault Table - ft_fuelsys_IntModel_faultInfo.xml

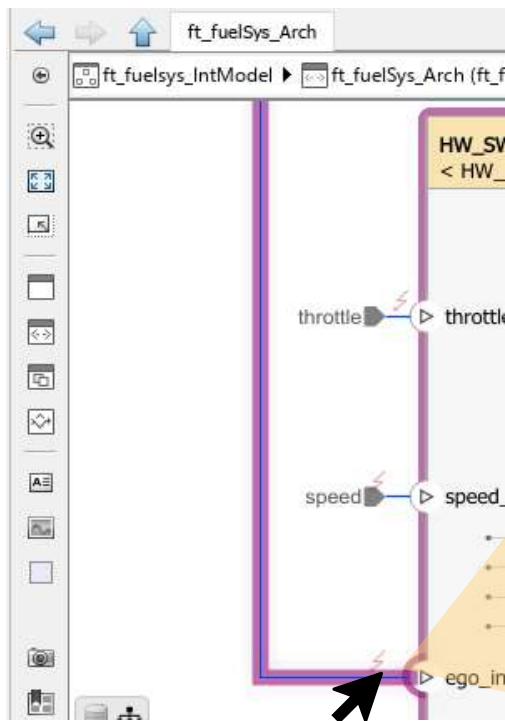
Fault: Conditional

Enable	Model Element/Fault Name	Active Fa
<input type="checkbox"/>	ft_fuelSys_Arch/HW_SW Conversion/l...	

Fault Input

Fault Output

Model faults without modifying Design



WHERE

Add a fault to a model element and specify the fault properties. To manage the fault, access the fault properties by clicking on the fault badge in the model or by opening the Fault Table pane.

Basic Properties Description

Model element: ft_fuelsys_IntModel/ft_fuelSys_Arch/HW_SW Conversion/Output/1

Fault name: sensors_fault

Fault information saved here... [Help](#)

Add fault behavior [Help](#)

Fault library: mwfaultlib Fault behavior: Stuck-at-Ground

Add fault behavior to: ft_fuelsys_IntModel_FaultModel

Trigger type: Always On

Inject fault behavior throughout the simulation.

Trigger type: Always On

Inject fault behavior: Always On

Where the faults is applied in the model

What's the behavior when injected

Time/condition when to be injected

Recap – Fault Analyzer Capabilities

Instantiate From Template

Link To Architecture

Execute Validation Checks

Function Name	Fault Mode	Model Element	Properties
Engine Conversion	Oil Alert	Oil Pressure Sensor	Oil Pressure Sensor
Engine Conversion	Oil Alert	Oil Pressure Sensor	Oil Pressure Sensor
Engine Conversion	Oil Alert	Oil Pressure Sensor	Oil Pressure Sensor
Engine Conversion	Oil Alert	Oil Pressure Sensor	Oil Pressure Sensor

Capture Analysis, Link to Architecture & Perform Validation Checks

Model Faults Without Modifying Design

Simulink Fault Analyzer™

WHERE

Where the faults is applied in the model

HOW

What's the behavior when injected

WHEN

Time/condition when to be injected

Characterize Faults Behavior

Fault Injected

Fuel Mode Reconfigure for faulty behavior

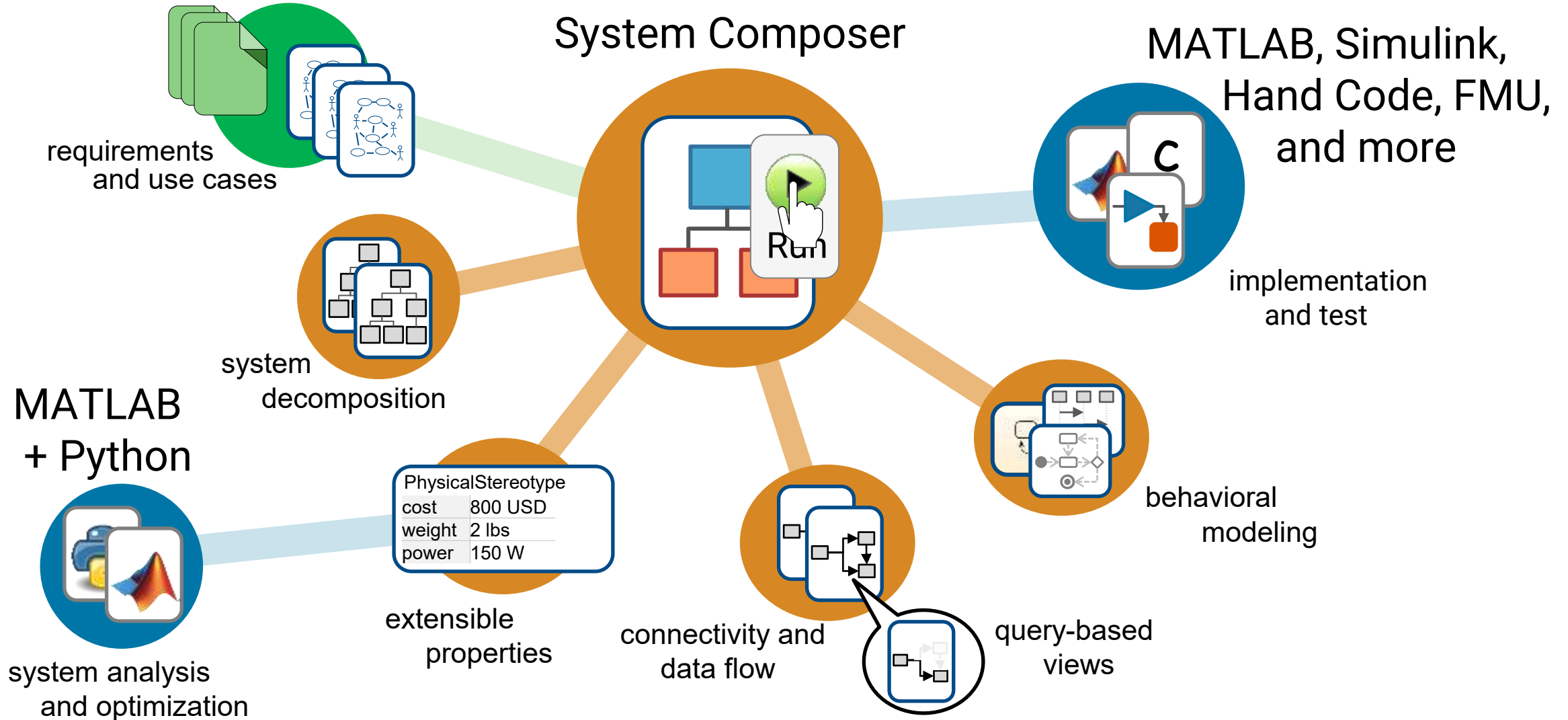
Run For All Faults

System Element	Detection Method	Fault Mode	Description	Product
(1) Engine Operation Interrupted	Oil Pressure Sensor	OilAlert	Oil	+ 100-0
(2) Engine Operation Interrupted	Oil Pressure Sensor	OilAlert	Oil	+ 100-0
(3) Engine Operation Interrupted	Oil Pressure Sensor	OilAlert	Oil	+ 100-0
(4) Engine Operation Interrupted	Oil Pressure Sensor	OilAlert	Oil	+ 100-0
(5) Engine Operation Interrupted	Oil Pressure Sensor	OilAlert	Oil	+ 100-0
(6) Engine Operation Interrupted	Oil Pressure Sensor	OilAlert	Oil	+ 100-0
(7) Engine Operation Interrupted	Oil Pressure Sensor	OilAlert	Oil	+ 100-0
(8) Engine Operation Interrupted	Oil Pressure Sensor	OilAlert	Oil	+ 100-0
(9) Engine Operation Interrupted	Oil Pressure Sensor	OilAlert	Oil	+ 100-0
(10) Engine Operation Interrupted	Oil Pressure Sensor	OilAlert	Oil	+ 100-0

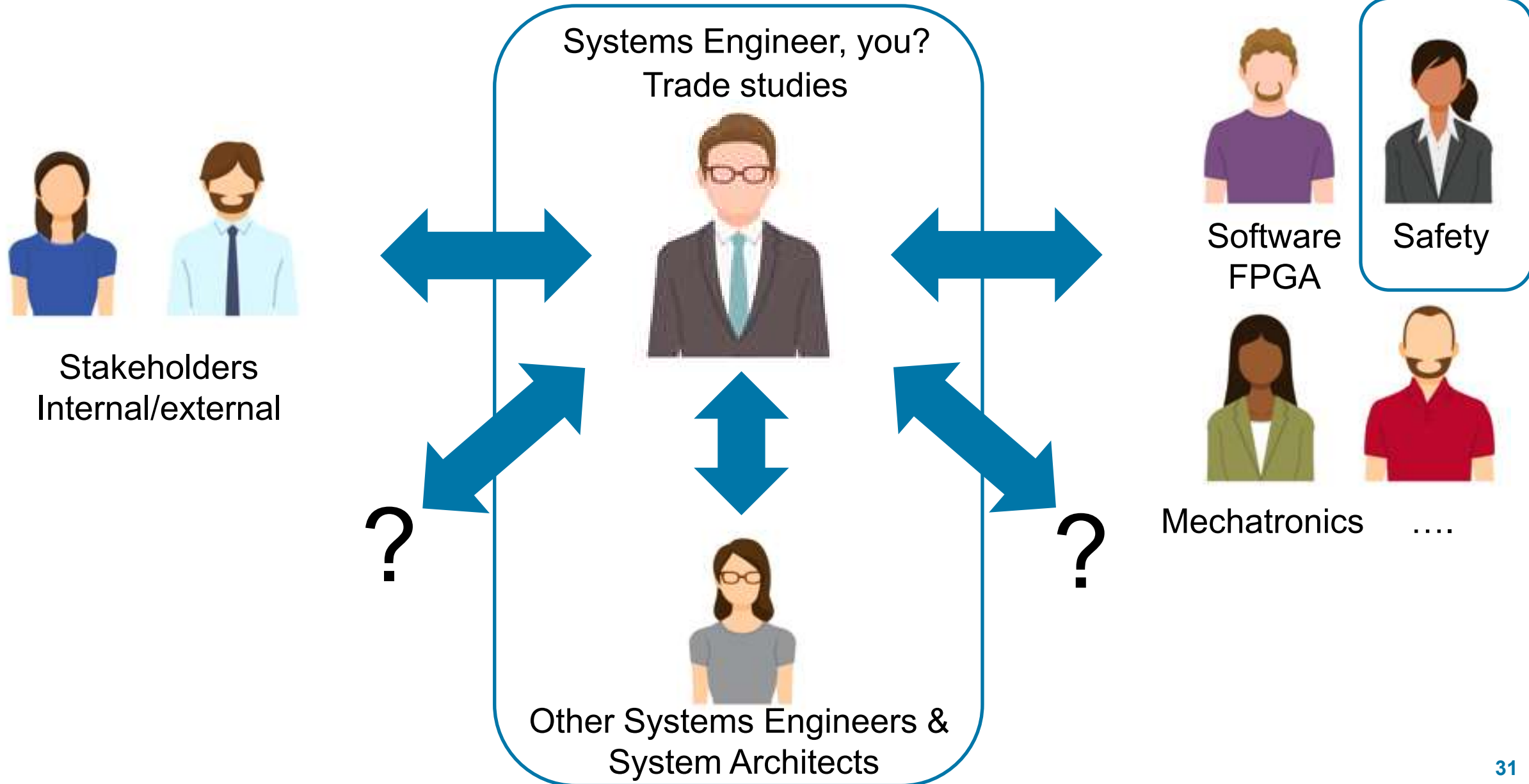
Use Simulation to Validate Safety Analysis

Model-Based Systems Engineering at MathWorks

Managing Requirements

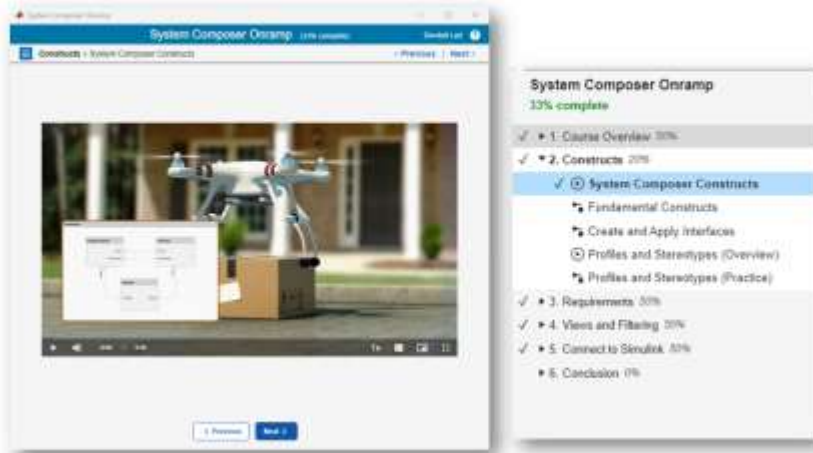


Practical use-cases in context



Summary

System Composer Onramp Course



Scan me for digital handouts

Interchange with SysML v1 models

SysML Connector

Import SysML models into System Composer

[Request support package](#)



Import your SysML v1 models into System Composer.

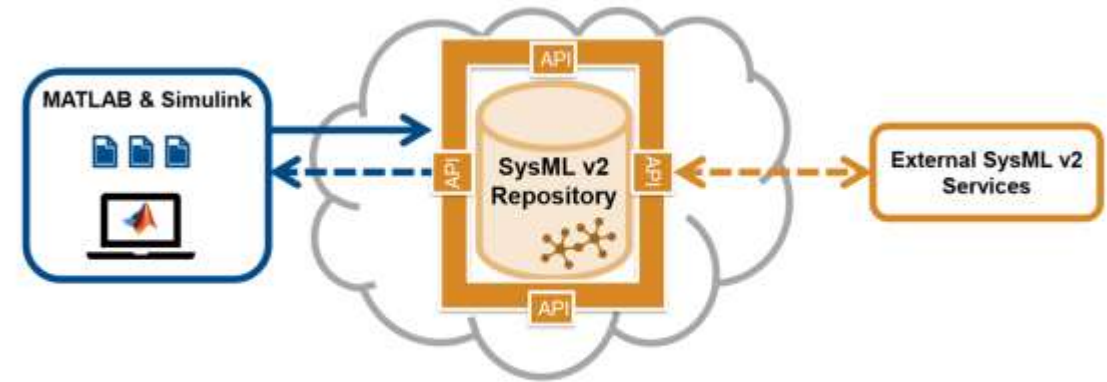
Export your System Composer models to SysML v1.

- Limited to architectures and interfaces
- We are investing in a SysML v2 solution to replace this

To request the SysML Connector:
<https://www.mathworks.com/products/sysml.html>



Our Vision for Tool Interoperability: SysML v2 support



Please feel free to ask about our plans to support SysML v2