FMI - AADL study

SAMARES ENGINEERING
Accelerate Systems Design

Agenda

1. Context of the Study
2. FMI Overview
   - What is FMI?
   - FMI for Model-Exchange
   - FMI for Co-Simulation
   - Master-Algorithm
3. FMU Loading Strategy
   - Moon Lander in AADL
   - FMU Import and Simulation
4. Next Steps
   - Export AADL to FMU
   - Master-Algorithm Implementation

Agenda

1. **Context of the Study**
2. FMI Overview
   - What is FMI?
   - FMI for Model-Exchange
   - FMI for Co-Simulation
   - Master-Algorithm
3. FMU Loading Strategy
   - Moon Lander in AADL
   - FMU Import and Simulation
4. Next Steps
   - Export AADL to FMU
   - Master-Algorithm Implementation

Context of the Study

- AADL has multiple capabilities for analysis
  - ✓ Model-level simulation (AADL Inspector), but no interoperability
  - ✓ Model-checking (e.g. FIACRE, Ocarina/LNT, …), but limited capability w.r.t. Physical modeling
- Code generation allows for rapid prototyping
  - ✓ Capabilities to interface with hardware/physical world
  - ✓ But difficulty to perform in-depth testing in a controlled environment
- Connection with models from other communities (Modelica, …) is a desirable feature, for both functional and environment
- Option: intermediate path to move towards simulation both at model and code levels
  - ✓ Leverage AADL code generation capabilities
  - ✓ Use the FMI standard as a way to interface with various kinds of models
    - Model of the environment as FMI models, or FMU
    - Turn AADL models into FMU for system-level tests

Timeline

- 1 year study, completed September 2017

- ISAE and Samares Engineering as co-PI

- Study interplay between AADL and FMI standards
- Prototype solutions with Ocarina and other FMI compliant tools
  - ✓ OpenModelica, Ptolemy (UCBerkeley), ANSYS Simplorer

- **WP1 (03/2017): state of the art, definition of strategies**
- WP2 (09/2017): integration of FMU in AADL, started
- WP3 (09/2017): export AADL models as FMUs

Agenda

1. Context of the Study
2. **FMI Overview**
   - What is FMI?
   - FMI for Model-Exchange
   - FMI for Co-Simulation
   - Master-Algorithm
3. FMU Loading Strategy
   - Moon Lander in AADL
   - FMU Import and Simulation
4. Next Steps
   - Export AADL to FMU
   - Master-Algorithm Implementation

FMI Overview – What is FMI? (1/2)

> Open standard

- ✓ FMI standard (Functional Mock-up Interface) defines an open interfaces
- ✓ Started in 2008 within the ITEA 2 Modelisar project
- ✓ FMI 1.0 released in 2010 - FMI 2.0 released in 2014

fmi FUNCTIONAL MOCK-UP INTERFACE

> Motivations

- ✓ Large integrated heterogeneous M&S problems (tools, solvers, formats)
- ✓ Collaborative MBSE between OEM and suppliers
- ✓ Reuse of models
- ✓ IP protection

> Added values

- ✓ Ease model integration for early V&V
- ✓ No proprietary middleware
- ✓ Tunable visibility level (white-box to black-box)

SAMARES ENGINEERING
Accelerate Systems Design

FMI Overview – What is FMI? (2/2)

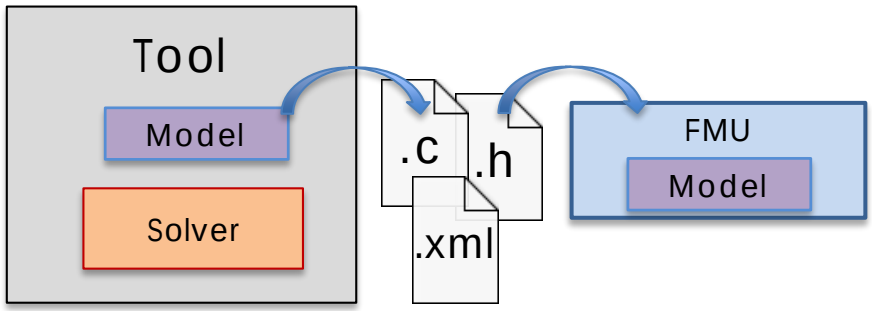
- FMI support (https://www.fmi-standard.org/tools)
  - ✓ Supported by 93 tools
  - ✓ Strong connections with physics, control/command, etc.
  - ✓ Automotive and non-automotive organizations
  - ✓ E.g. Dymola, JModelica, Simulation X, Amesim, Ptolemy II, Simplorer
- Underlying technology
  - ✓ Implemented by executable FMUs (Functional Mock-up Units)
  - ✓ FMUs support the simulation of hybrid-systems
  - ✓ FMUs can be connected
  - ✓ FMUs are binary + XML description
- Approaches
  - ✓ FMI for Model-Exchange (ME)
  - ✓ FMI for Co-Simulation (CS)

SAMARES ENGINEERING
Accelerate Systems Design

FMI Overview – FMI for Model-Exchange

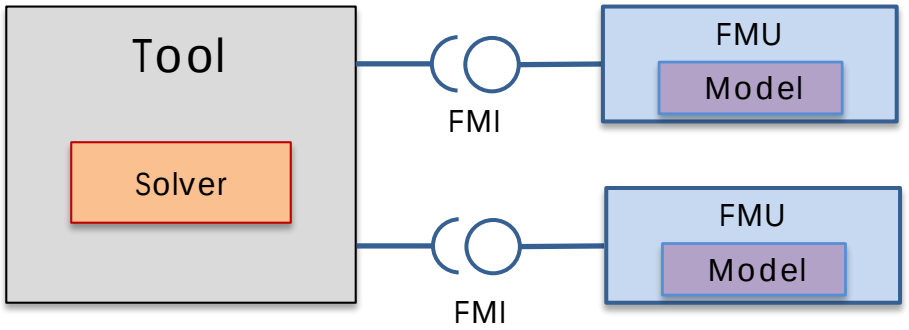
- FMI for Model-Exchange export
  - ✓ C code and XML file ("modeldescription") generation
  - ✓ Archives (zip) sources and binaries into a .fmu file
- FMI for Model-Exchange import
  - ✓ Requires external solver
  - ✓ FMUs can be connected to the rest of the model

FMU for ME export



Nom	Taille	Compressé	Modifié le
binaries	2 873 198	998 476	2016-12-12...
sources	3 775 019	869 227	2016-12-12...
modelDescription.xml	4 302	923	2016-12-12...

FMU for ME import

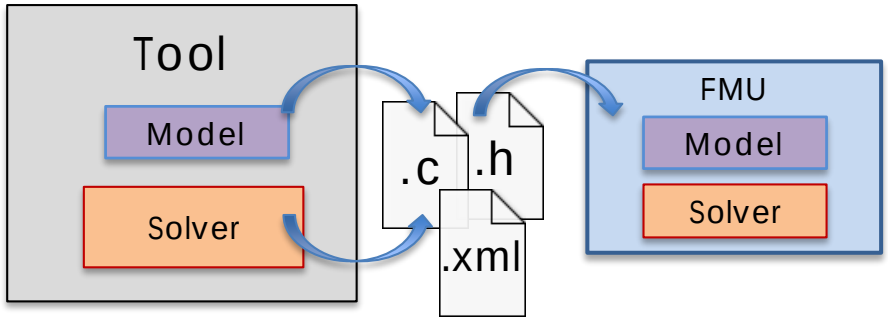


SAMARES ENGINEERING
Accelerate Systems Design

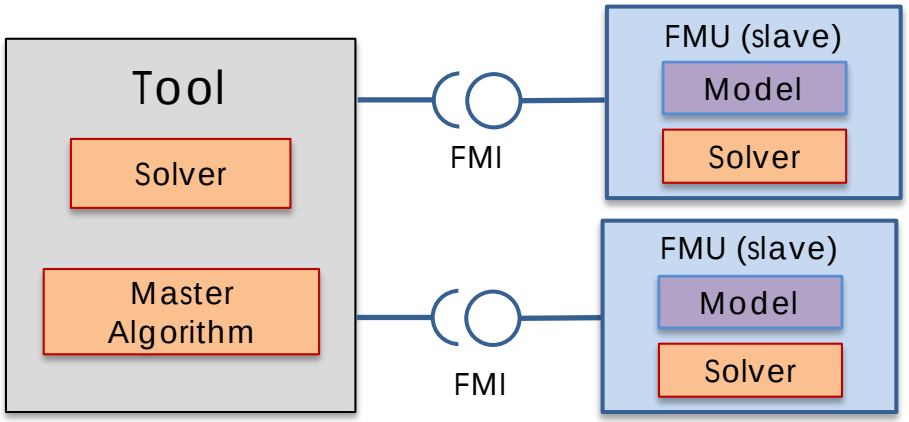
FMI Overview – FMI for Co-Simulation

- FMI for Co-Simulation export
  - ✓ C code and xml file generation with **embedded solver**
  - ✓ Archives (zip) sources and binaries into a .fmu file
- FMI for Co-Simulation import
  - ✓ Requires Master-Algorithm (MA): **synchronize slaves**
  - ✓ FMUs can be connected to the rest of the model

FMU for CS export



FMU for CS import

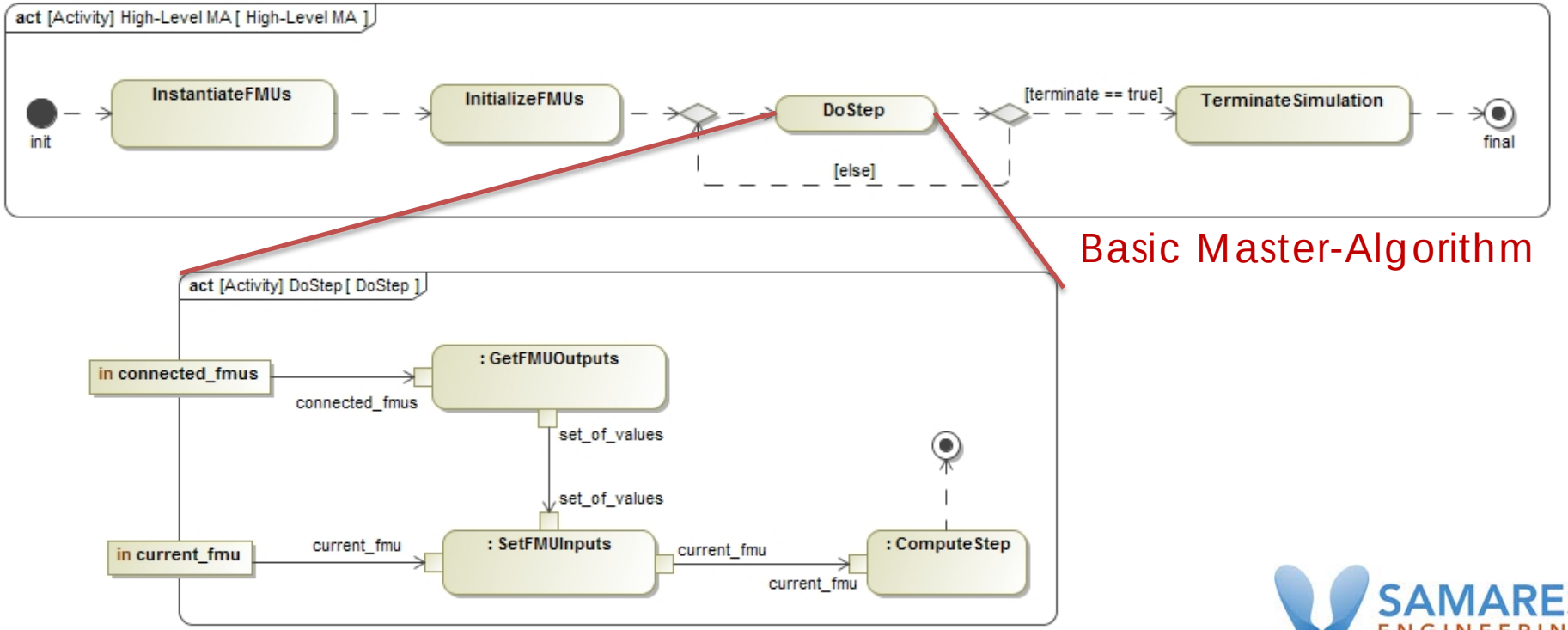


SAMARES ENGINEERING
Accelerate Systems Design

FMI Overview – Master Algorithm

> Master-Algorithm for Co-Simulation

- ✓ Instantiate and initialize each FMU
- ✓ For each FMU at communication step: values exchange (outputs/inputs) and step calculation



Basic Master-Algorithm

Basic "DoStep" function

SAMARES ENGINEERING
Accelerate Systems Design

Agenda

1. Context of the Study
2. FMI Overview
   - What is FMI?
   - FMI for Model-Exchange
   - FMI for Co-Simulation
   - Master-Algorithm
3. **FMU Loading Strategy**
   - Moon Lander in AADL
   - FMU Import and Simulation
4. Next Steps
   - Export AADL to FMU
   - Master-Algorithm Implementation

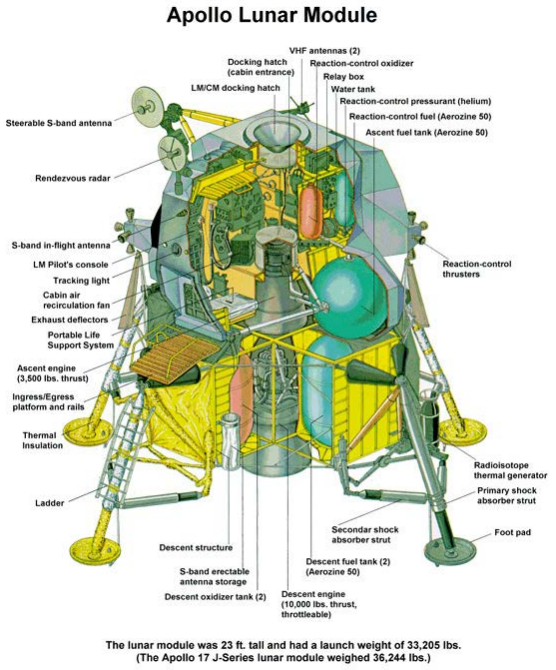
FMU Loading Strategy – MoonLander in AADL

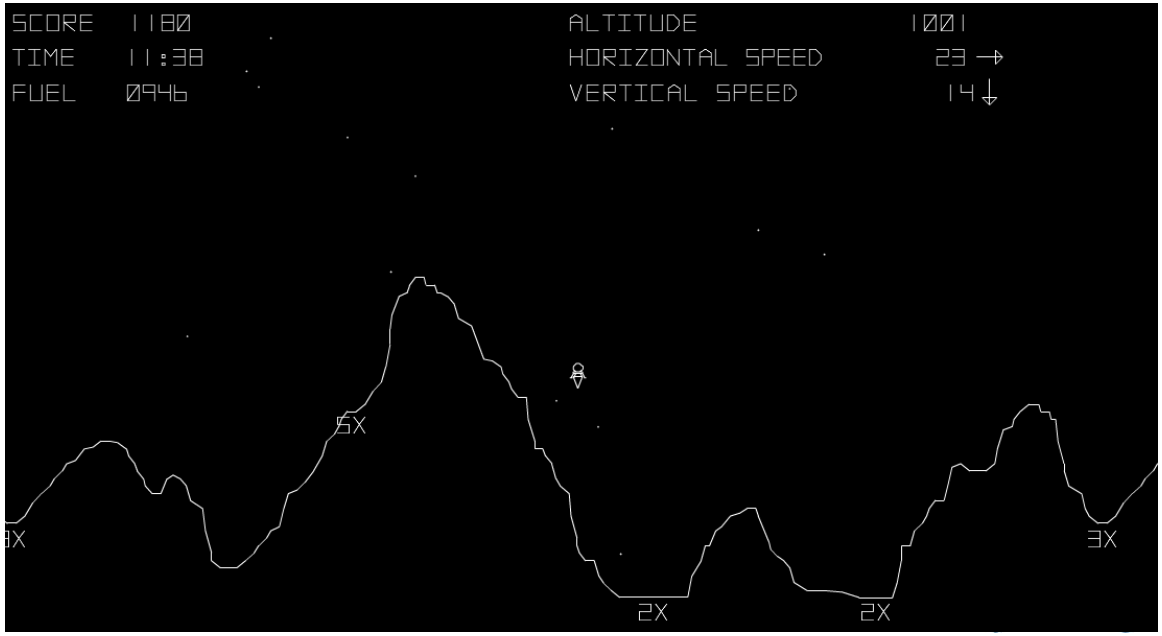
- **Part of WP1 + WP2 prototyping**
- Use case studies to drive requirements and implementations
  - ✓ Use a basic MoonLander as example as first test
- WP2 focus: have the capability to test AADL systems
  - ✓ AADL : embedded control part, to be tested
  - ✓ FMU: model of the environment, the plant
- Study objective: limit transition from simulation to real systems
  - ✓ FMU are mock-up for the AADL devices
  - ✓ FMU are triggered just like regular device drivers

- More complex timing semantics (e.g. continuous time) could be added to simulate faults and their propagation. Outside the scope of this study. Creates issues in terms of simulation algorithm

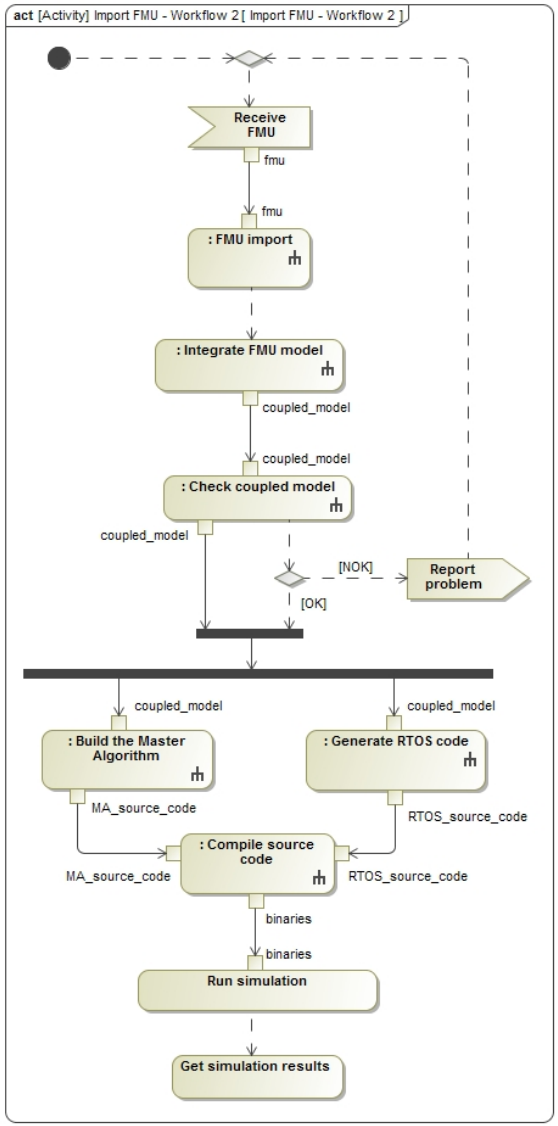
About the Moon Lander

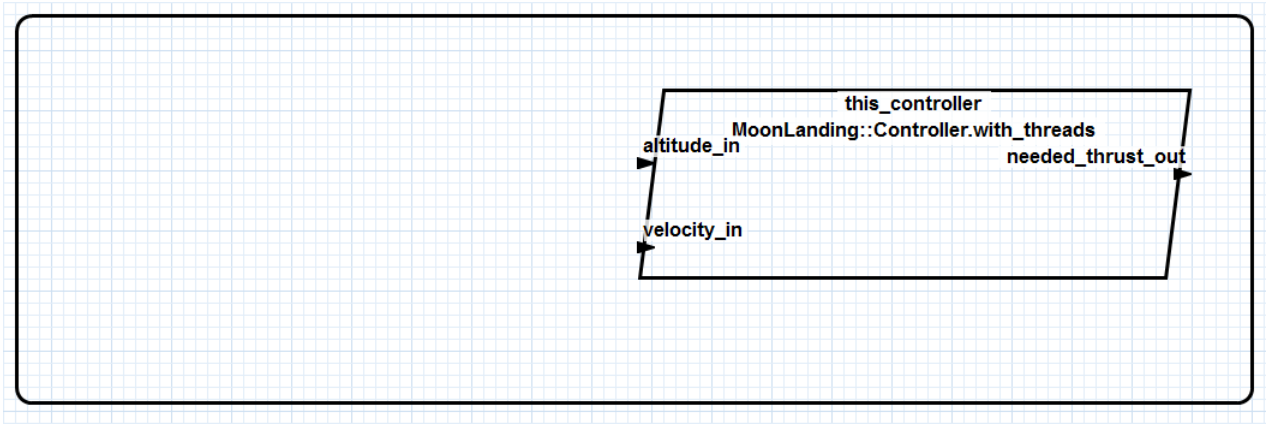
- Mix of physics and control,
  - ✓ Physics driven by basic equations
    - Gravity + thrusters
  - ✓ 2 x Thrusters controlled by a basic controller
- A popular game several years ago !



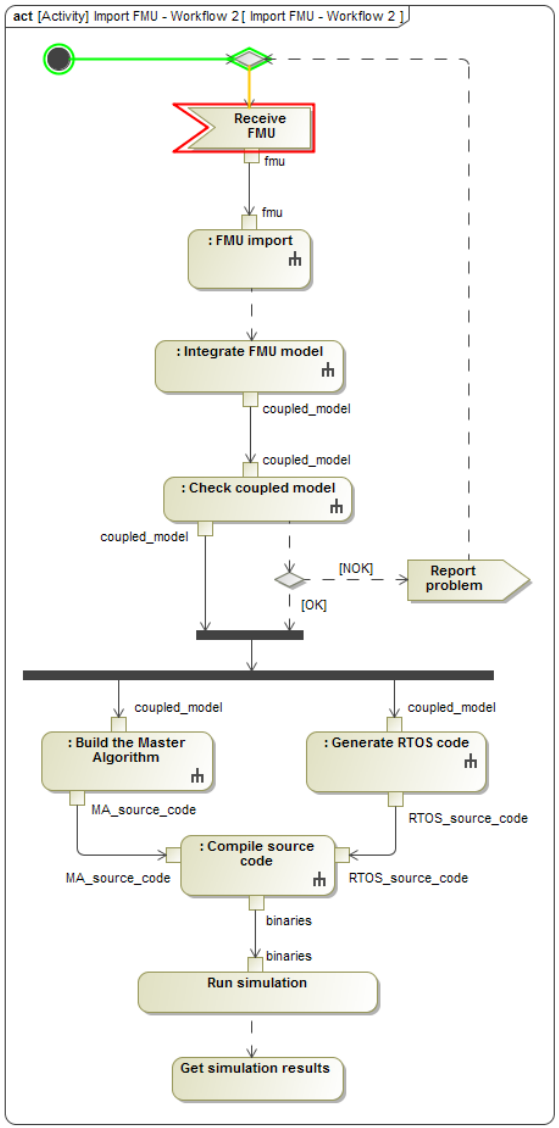


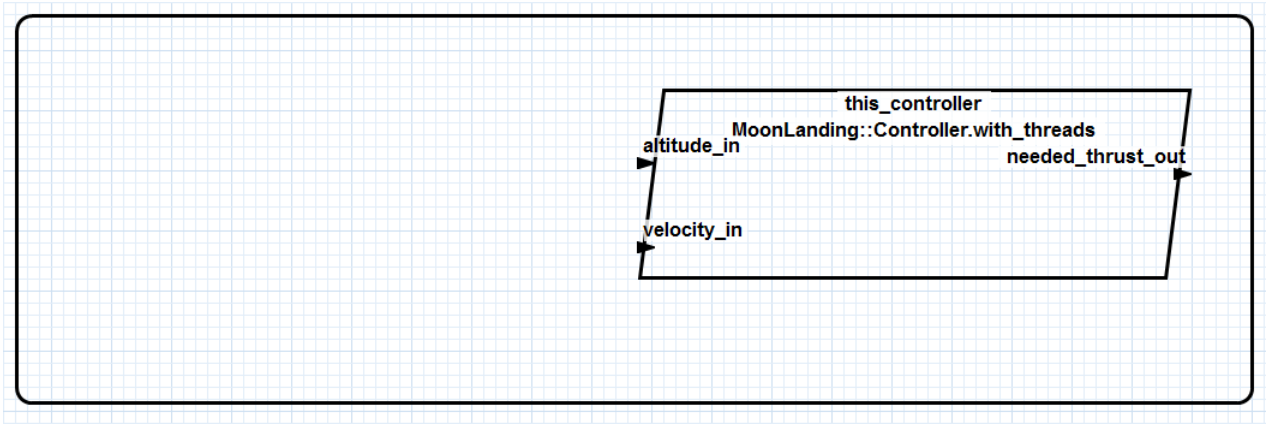
FMU Loading Strategy – FMU Import and Simulation



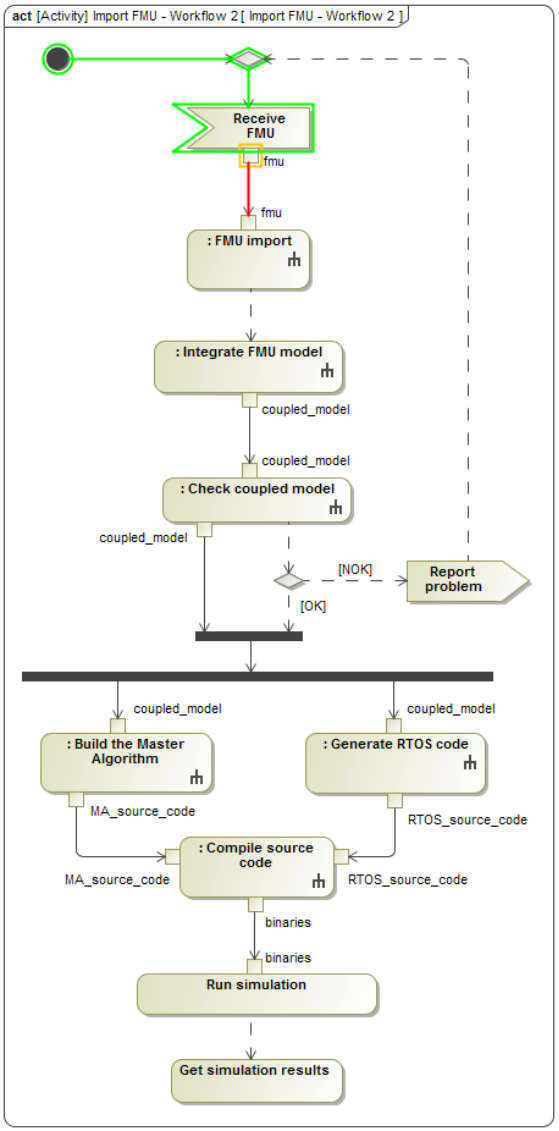


FMU Loading Strategy – FMU Import and Simulation

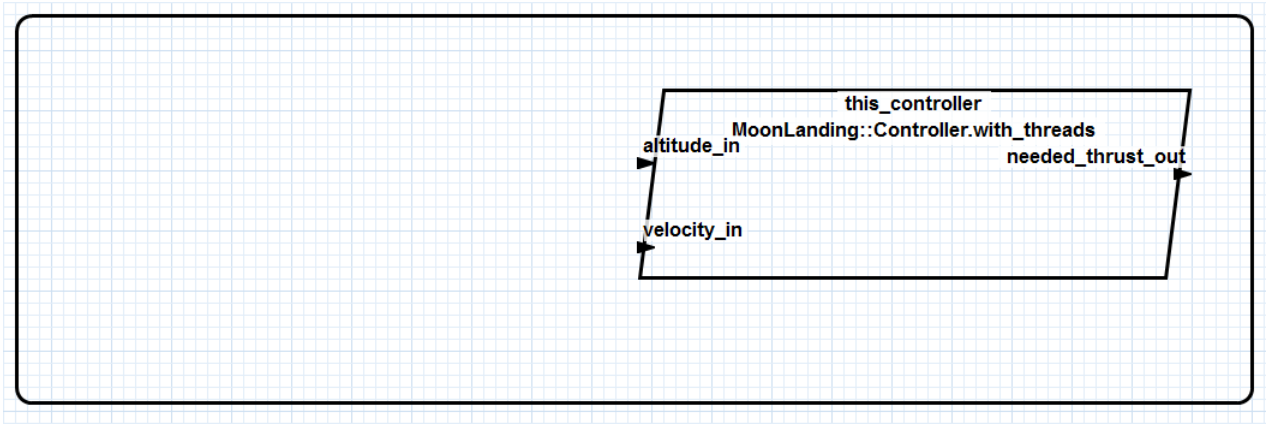




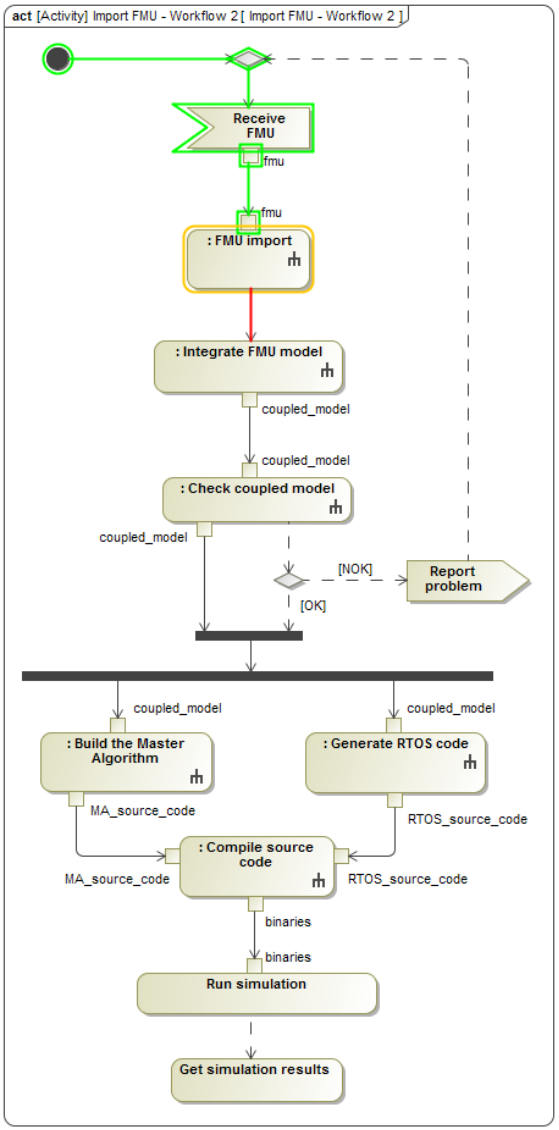
FMU Loading Strategy – FMU Import and Simulation



Nom	Taille	Compressé	Modifié le
binaries	2 873 198	998 476	2016-12-12...
sources	3 775 019	869 227	2016-12-12...
modelDescription.xml	4 302	923	2016-12-12...

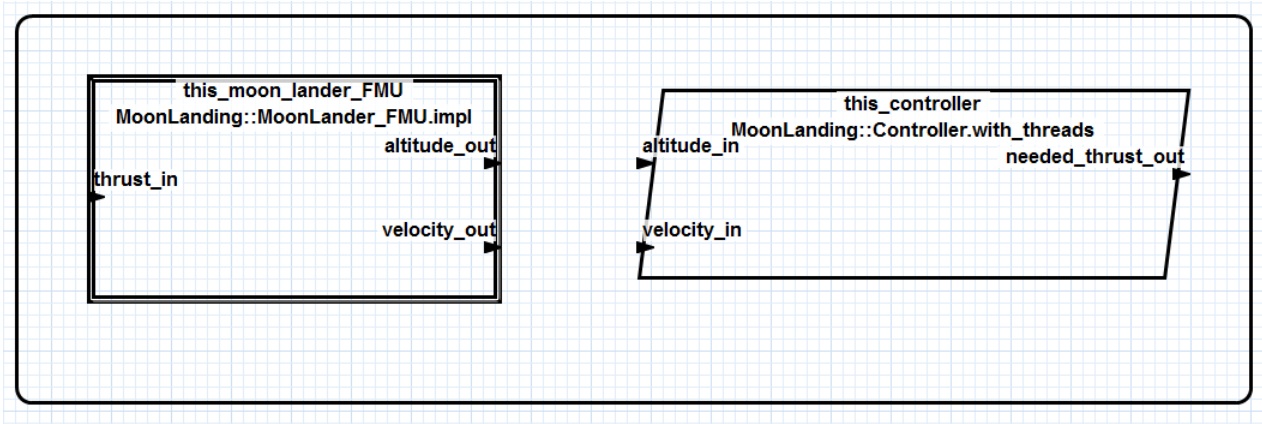


FMU Loading Strategy – FMU Import and Simulation

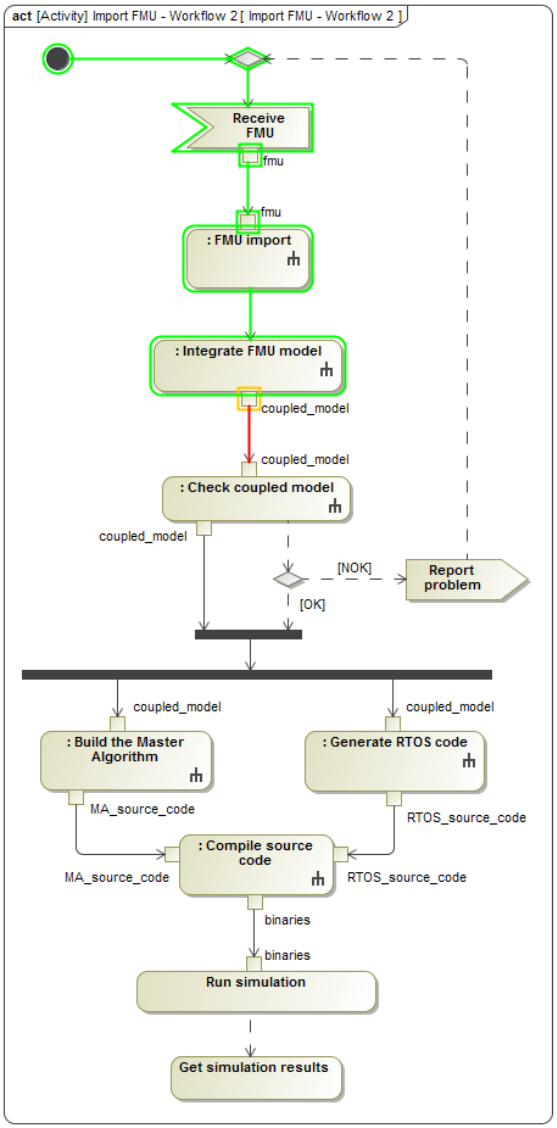


> FMU import

- ✓ Unzip fmu file
- ✓ Parse modeldescription.xml
- ✓ Load dll file

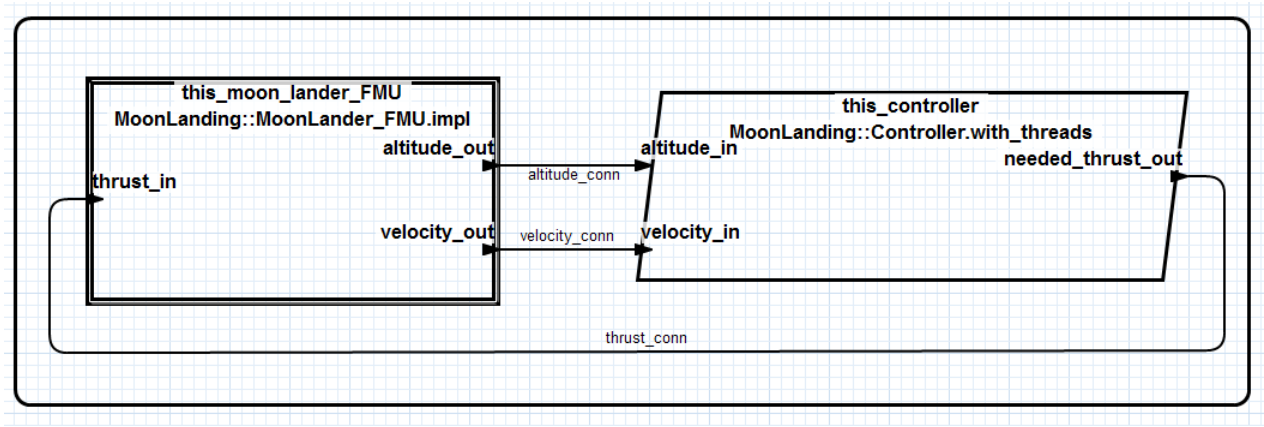


FMU Loading Strategy – FMU Import and Simulation

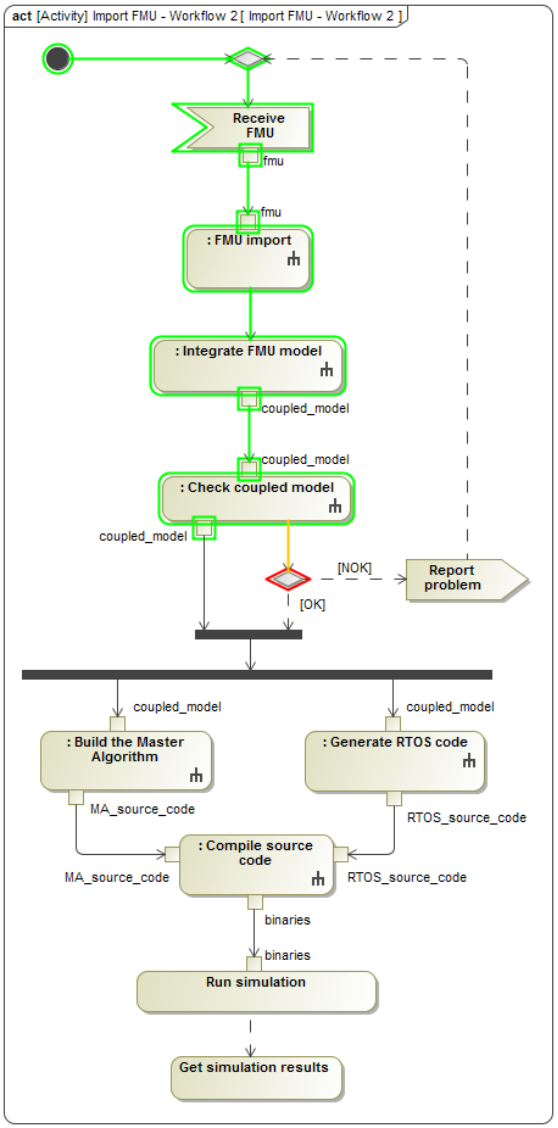


> Integrate FMU

- ✓ Create connectors between ports
- ✓ The result is a "coupled model"

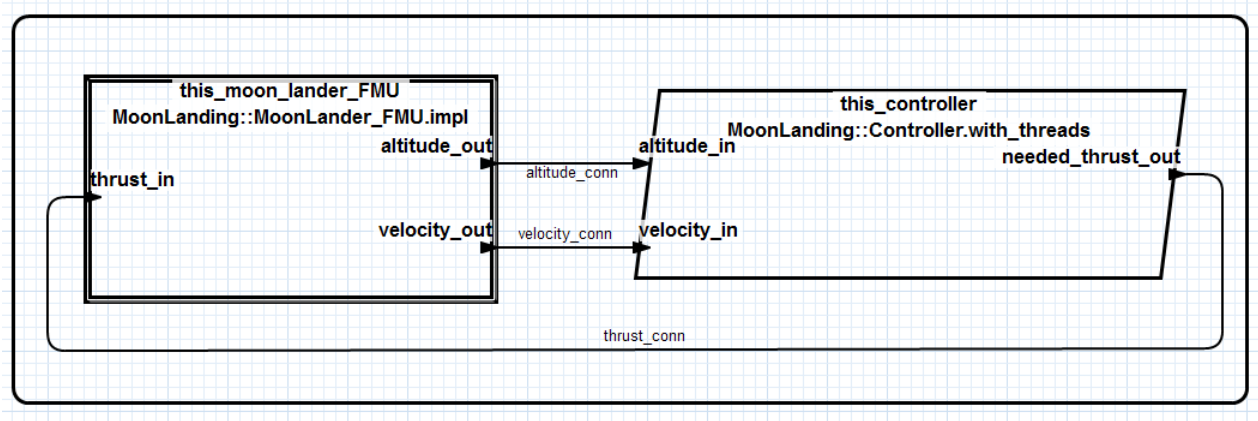


FMU Loading Strategy – FMU Import and Simulation



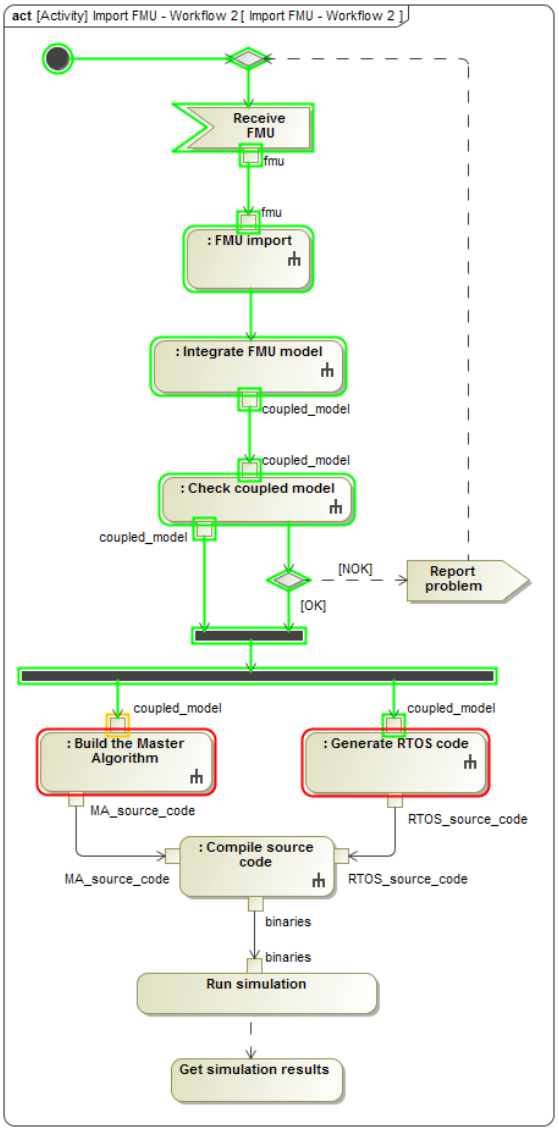
> Check coupled model

- ✓ Outputs -> Inputs types compatibility
- ✓ Ports causality and variability (events, data)



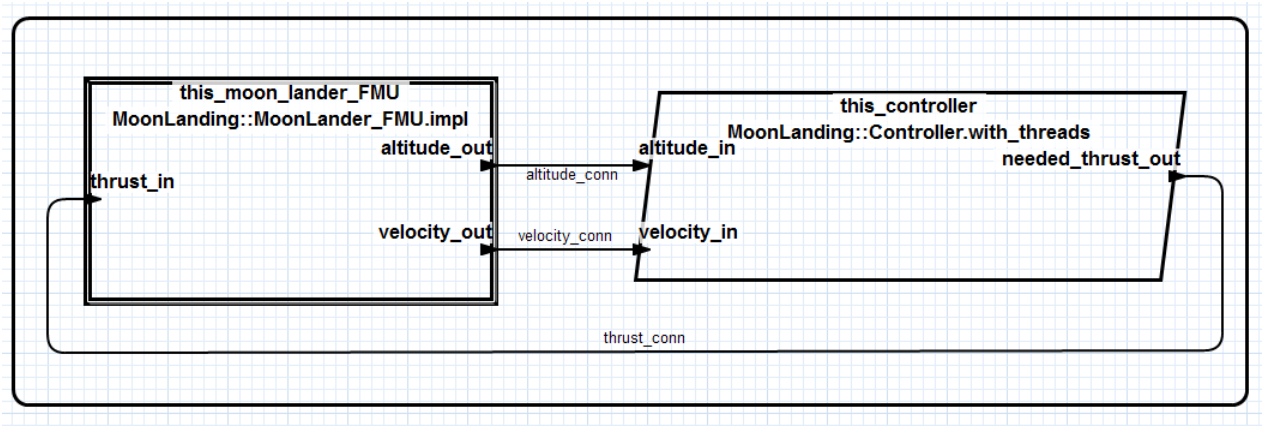
SAMARES ENGINEERING
Accelerate Systems Design

FMU Loading Strategy – FMU Import and Simulation

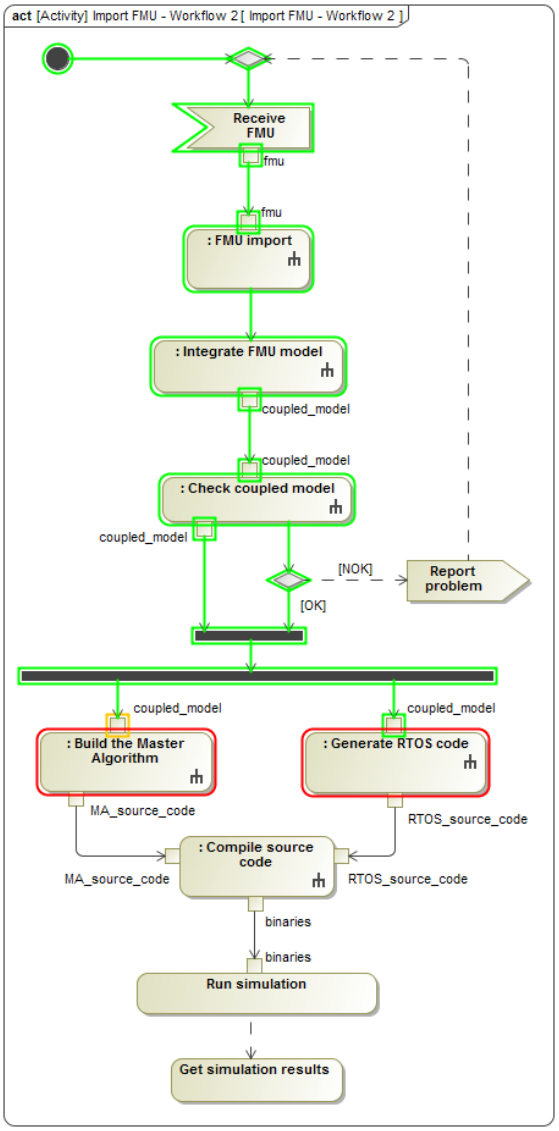


> Build or generate the Master Algorithm

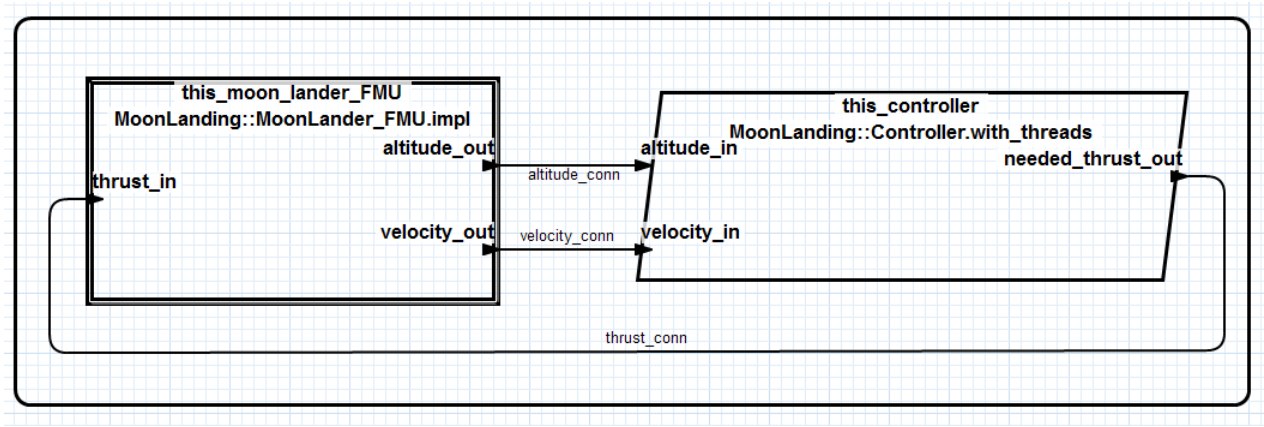
- ✓ Generate C code:
  - instantiate, initialize, do step
  - setup the communication step size
- ✓ Need to compute the dependency graph
  - values exchange between components
  - detection of algebraic loops



FMU Loading Strategy – FMU Import and Simulation

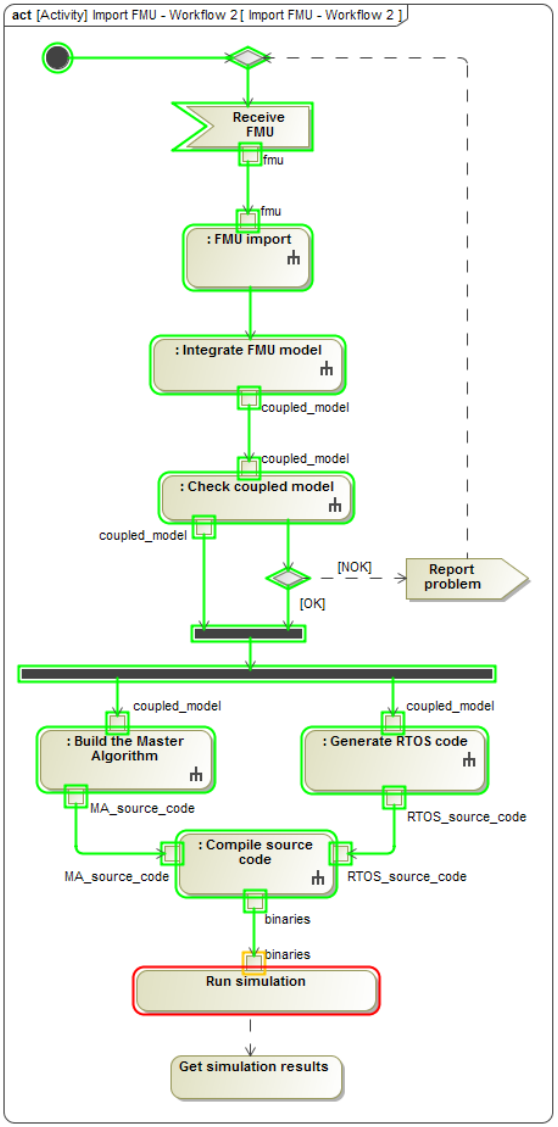


- Generate "RTOS" code
  - ✓ Generate C for the controller, with Ocarina
  - ✓ The FMUs are embedded as library
- "RTOS" = true RTOS or simulated scheduler
  - ✓ RT-POSIX tested
  - ✓ Simulated scheduler for WP3

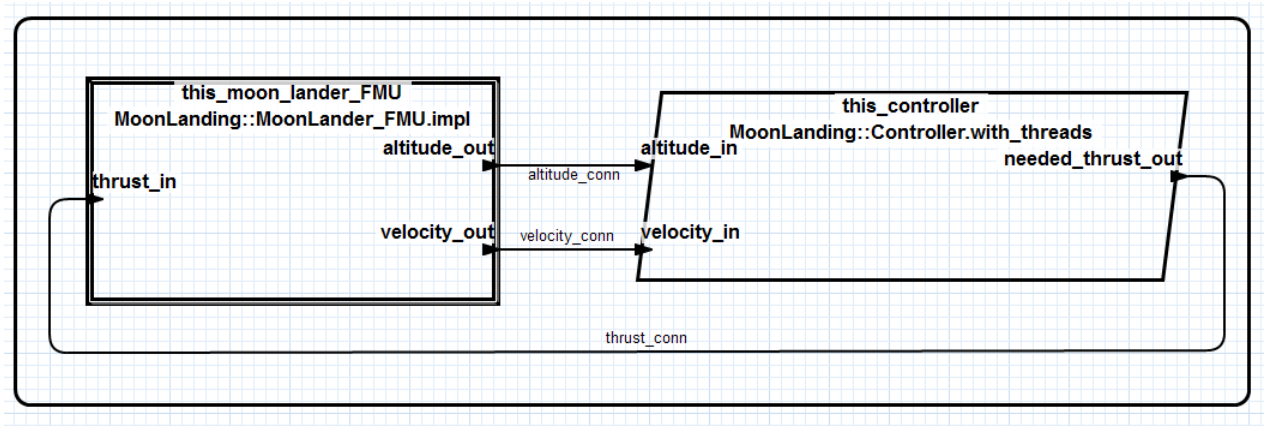


SAMARES ENGINEERING
Accelerate Systems Design

FMU Loading Strategy – FMU Import and Simulation

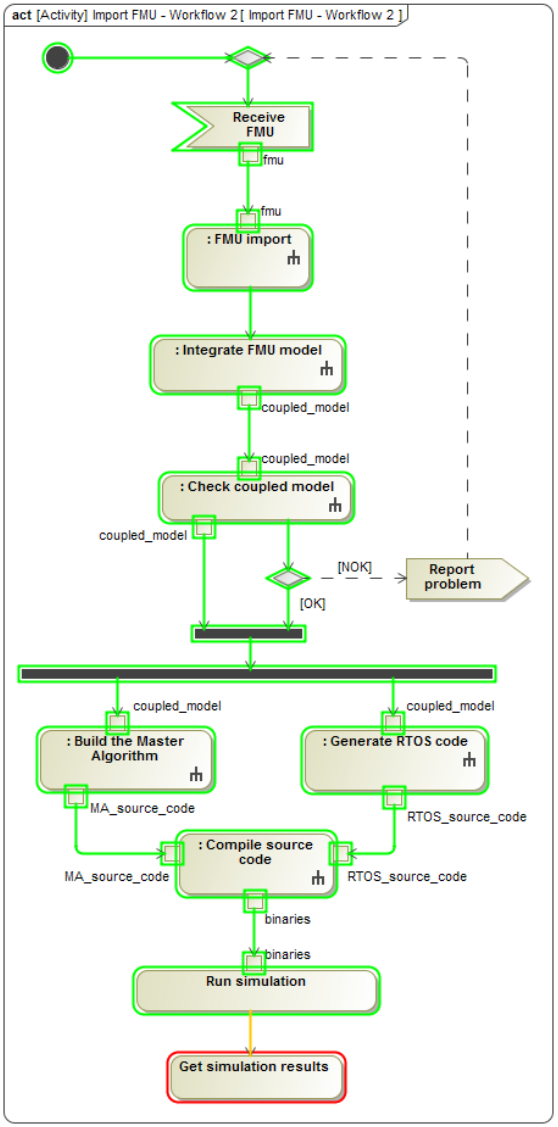


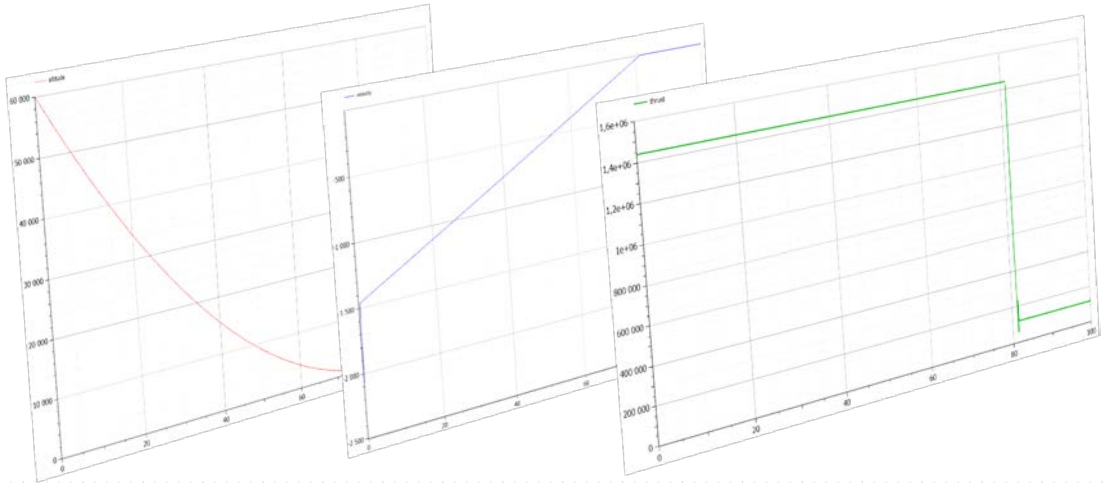
- Run simulation
  - ✓ Setup end time
  - ✓ Setup step-size
  - ✓ Setup log level
  - ✓ Launch simulation

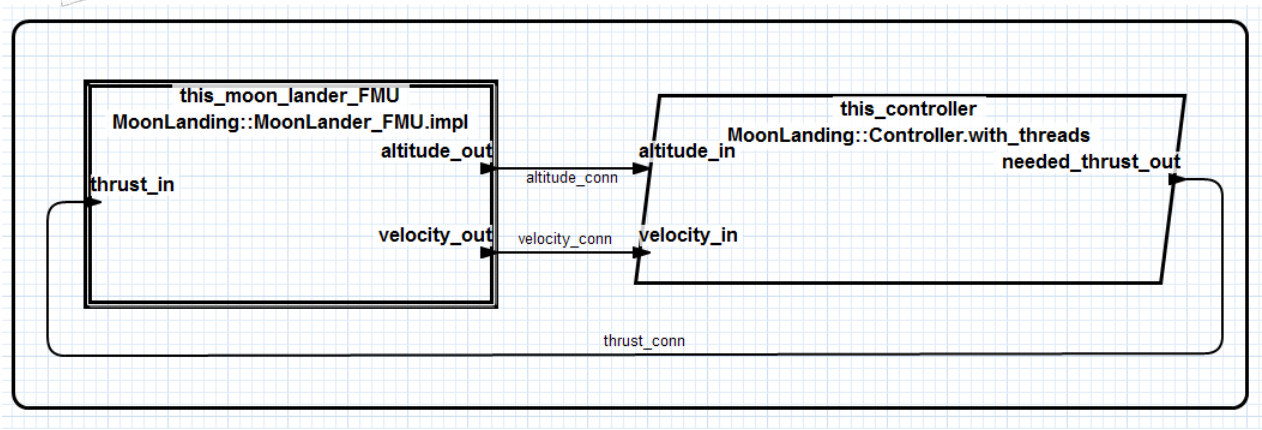


SAMARES ENGINEERING
Accelerate Systems Design

FMU Loading Strategy – FMU Import and Simulation

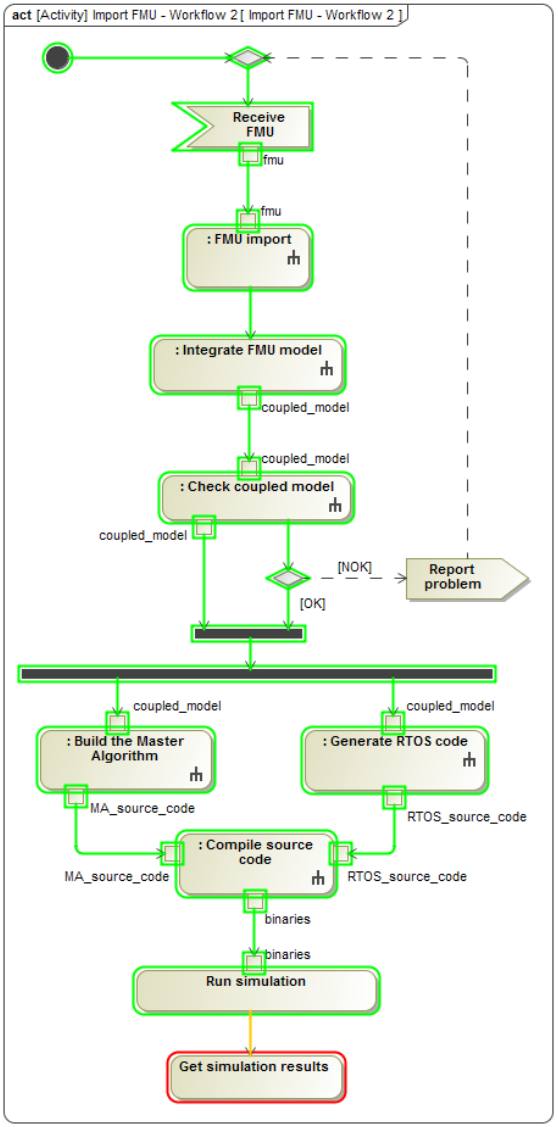




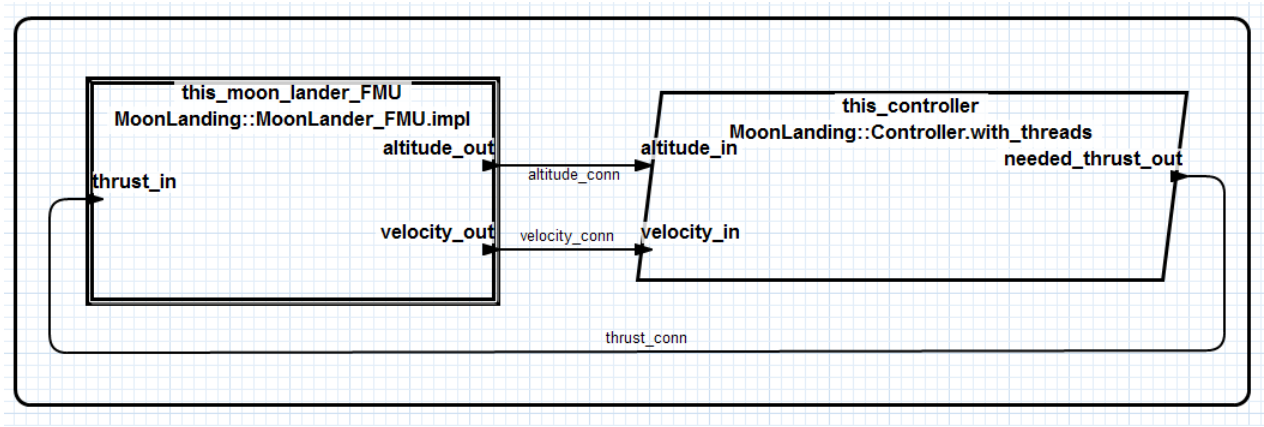


SAMARES ENGINEERING
Accelerate Systems Design

FMU Loading Strategy – FMU Import and Simulation



- First prototype
  - ✓ FMUSDK2 from Modelion, adapted by UCB
  - ✓ Generic FMU_Activate_Entrypoint to load MFU
  - ✓ FMU specific Compute_Entrypoint_MoonLanding
    - To be generated, manual for now
  - ✓ Device ”polled” by thread



WP2 roadmap strategy

- Feedback on the first prototype
  - ✓ AADL has most of required elements
  - ✓ FMU embedded as a device, either
    - Polling mode: a basic subprogram called by AADL elements
    - Interrupt mode: wrapped in an AADL thread + regular properties
    - Could work also as a thread (e.g. to embed SCADE, Simulink ..)
  - ✓ Next weeks will focus on prototyping code generation capabilities
    - Manual implementation for the moment
- Having sound simulation is a requirement
  - ✓ Step-based solvers are preferred
    - What about other solvers ? Connection with CPS work?
    - Should it be part of the visible interface ? constraints/checks ?
      - E.g. going back in time solvers
  - ✓ How to configure the simulation?
    - Timestep <-> AADL period
    - Duration of the simulation <-> ? CLI argument

Agenda

1. Context of the Study
2. FMI Overview
   - What is FMI?
   - FMI for Model-Exchange
   - FMI for Co-Simulation
   - Master-Algorithm
3. FMU Loading Strategy
   - Moon Lander in AADL
   - FMU Import and Simulation
4. **Next Steps**
   - Export AADL to FMU
   - Master-Algorithm Implementation

Next Steps – Export AADL to FMU

- Part of WP3: turn an AADL model into a FMU
- Requirements:
  - ✓ Define the interface of the component to export, e.g. a process
  - ✓ Ensure one can build a DLL from it, thanks to code generation
    - Already done
  - ✓ Ensure a consistent semantics, time-wise
    - Option#1: export to an OS API for concurrency, feasible iff real-time simulation
    - Option#2: export to a simulator of an OS, required if time-step simulation
- Both options will be explored

Conclusion

- First steps to align FMI and AADL in two simple scenarios
  - ✓ Import FMU as an AADL device,
    - Could also work to import FMU as AADL thread (e.g. Simulink or SCADE)
    - Working prototype
  - ✓ Export AADL process as an FMU
    - Considerations for time management and concurrency aspects to handle
    - Feasibility evaluated in another project
- Next 8 months will work on consolidating both tracks
  - ✓ Building more case studies: Moon Lander, UAV
  - ✓ Extending scope of tools: ANSYS Simplorer ? Others ?
  - ✓ Extending models for interoperability