Model-based multi-disciplinary co-simulation and co-modelling

Peter Gorm Larsen, Aarhus University

and

Cláudio Gomes, University of Antwerp
Outline

Co-simulation
- Motivation for M&S in MSBE
- Co-simulation
- Remote simulators
- Functional-Mock-up Interface
- Competent co-simulation

The DESTECS project
- Embedded Control Systems
- Co-Modelling, Co-Simulation
- Fault tolerance and handling

The INTO-CPS project
- Cyber Physical Systems (CPSs)
- System Vision
- Multi-modelling
- FMI-based co-simulation

The practical assignment
- A line-following robot
- Assumptions
- Development lines
The Modern Car

- Complexity
  - 40-100 subsystems
- Competitive Market
- Concurrent Development
  - Late Integration Problems
- Distributed Development
  - Specialized suppliers
  - OEM wants to
    - Evaluate multiple components
    - Perform early system integration
  - Supplier IP protection
M&S in MBSE

• V-Process
  – Design
    • Requirements (0D model)
    • Dynamics (1D model)
    • Mesh (3D model)
  – Validation
    • Reuse design experimentation results

• Simulation in all stages
• V-process also applies to more complex systems
M&S in MBSE

Automotive MBSE

Function Model
- Discretion, Scaling, Augmentation, Embedding

Production Model
- Autocoding

Production Code
- MvL-Test
- RCP

Simulation System
- Prototyping HW + Vehicle

Controller + Simulation of Environment
- Controller + Vehicle

Integration
- PiL/HiL-Test
- SiL-Test

Simulation System
M&S in MBSE

• Early access to models of components.
  – Test different control approaches
  – Evaluate same component from different suppliers

• Challenges:
  – Different teams/suppliers use different modelling tools
  – IP Protection

• Exchange Models
  – Leads to Vendor Lock-in
  – Simulation tools must support import

from www.ni.com/
Co-simulation

- Simulation of a system
  - Coupling of multiple simulators
  - Optionally as black-boxes
  - Each simulating one or more models
  - Built with different formalisms/tools.

- Co-simulation scenario
  - Description of the system
  - The simulators and their dependencies
  - Data about the capabilities of each simulator.
Remote Simulators

• Suppliers make a simulator available through an API
  – Integrator takes care of programming an interface
  – Good IP Protection
  – Different suppliers require different interfaces
Functional Mock-up Interface Standard

• Simulator and model exported as a standardized C library
• Standard interaction with any simulator
• Every simulator is a black box.
  – Executed locally but can communicate with a remote server
A Functional Mockup Unit is a zip-file (.fmu) consisting of
- C Library (.dll or .so)
- XML file (metadata)

The coupling (master algorithm) must be provided
Inside an FMU

Real or Simulated

Model

Solver

Symbolic Info

Compiled Model

Solver
FMU Example

```c
fmi2Status fmi2DoStep(fmi2Component fc, fmi2Real currentCommPoint, fmi2Real commStepSize, fmi2Boolean noPrevFMUState)
{
    FMUInstance* fi = (FMUInstance*)fc;
    fmi2Status simStatus = fmi2OK;
    printf("%s in fmiDoStep()\n", fi->instanceName);
    fi->currentTime = currentCommPoint + commStepSize;
    printf("Motor_in: %f\n", fi->r[_motor_in]);
    printf("slave CBD_PART2 now at time: %f\n", fi->currentTime);
    fi->r[_position] = fi->r[_position] + fi->r[_velocity] * commStepSize;
    fi->r[_velocity] = fi->r[_velocity] + fi->r[_acceleration_after_fric] * commStepSize;
    fi->r[_friction] = fi->r[_velocity] * 5.81;
    fi->r[_motor_acceleration] = fi->r[_motor_in] * 40;
    fi->r[_acceleration_after_fric] = fi->r[_motor_acceleration] - fi->r[_friction];

    return simStatus;
}

fmi2Status fmi2GetReal(fmi2Component fc, const fmi2ValueReference vr[], size_t nvr, fmi2Real value[])
{
    FMUInstance* comp = (FMUInstance*)fc;
    int i;
    for (i = 0; i < nvr; i++)
    {
        value[i] = comp->r((vr[i]));
    }
    return fmi2OK;
}
```
FMI Co-Simulation Scenario

Covered by the FMI standard
FMU States

• Synchronization algorithm (master)
  – Communicates with each individual simulator
  – Moves data from one simulator to the other
  – Coordinates time
Research Challenges

• Can we trust the co-simulation results?
  – Computer Science
  – Numerical
  – Physics
Research Challenges: Computer Science

- Real-time constraints
  - E.g., Hardware (HiL)
- Make the most of heterogeneous capabilities
  - Fixed or adaptive time-step
  - no/single/multiple rollback support
- Hierarchical co-simulation
- Different information exposed about each simulator
  - IO Dependencies
  - Numerical algorithm
  - Recommended step size
  - Jacobian matrices
  - Operating conditions (e.g., range of stability)
- Parallelism
  - Determinism
  - Deadlocks
  - Fairness
Research Challenges: Numerical

• **Time synchronization**
  – Correct interleaving of the execution of each simulator.
  – Including data dependencies.

• **Time progression**
  – Handle Zeno behaviours

• **Algebraic (instant) dependencies**
  – Detect and solve.

• **Compositionality**
  – State event location
  – Stability
Research Challenges: Physics

• Extra coupling equations might be necessary
  – E.g., $c^2 = 0.5 \iff c = 0.25$ or $c = -0.25$

• Inconsistent values
  – E.g., Voltage
Validation in Industry

Abstraction/Refinement:
\[ M_i \supseteq_P M_{i+1} \equiv \forall p \in P, M_i \models p \implies M_{i+1} \models p \]

Engineering process:
\[ M_0 \supseteq_P M_1 \supseteq_P \cdots \supseteq_P M_n \supseteq_P R \]

Verification (\( M_i \models p \)) is approximate: \( M_i \models p \]

Validation becomes: \( M_i \models p \implies M_{i+1} \models p \]

Competency (\( M_i \models p \implies R \models p \)) becomes the goal
Validation in Industry (Example)

- Hydraulic connection introduces delay.
- If not modeled, refinements may be valid but not competent.
References
