



Cosimate - FMI Coupling

KiasTek SAS

Functional Mock-up Interface

- FMI for Model Exchange
 - Provides a standard access to simulate model equations
- FMI for Co-simulation
 - Provides a standard interface for coupling 2 or more simulation tools.

FMI for Co-simulation Overview

- Standard interface between master and slaves:
 - Master = Simulation tool or Co-simulation backplane
 - Slave = Simulation solver
- 2 “co-simulation” approaches:
 - Master loads a DLL containing both model code and the FMI implementation = 1 or 2 processes
 - Master controls a solver using a DLL = 2 processes
- Implementation:
 - Minimal set of C-functions
 - The master algorithm is **NOT** part of the standard.

FMU: Functional Mock-up Unit

- ZIP archive including:
 - A FMI description of the model: XML file
 - Documentation
 - Shared Library (implements the FMI API)
- Master opens the FMU file
 - Loads the model description
 - Loads the DLL to either
 - Simulate model equations
 - Run the simulator slave
- Notice: Component-connection graph (netlist): **no definition available.**

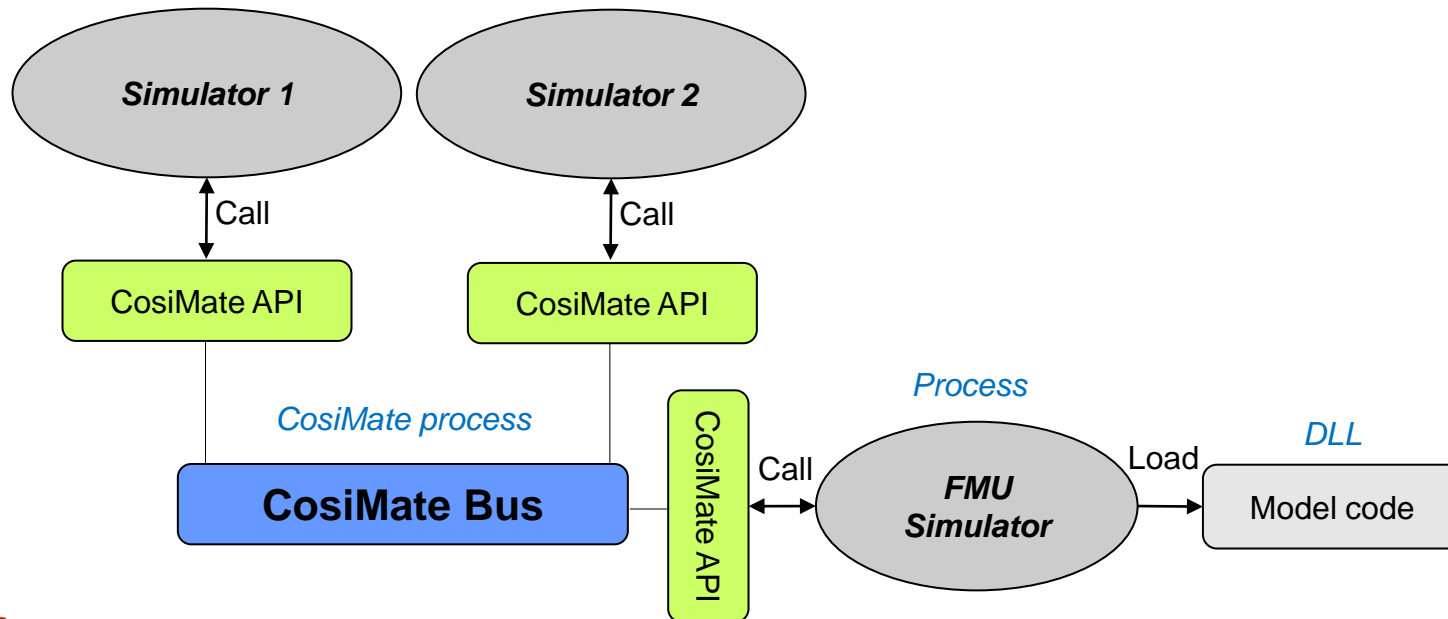
```
// Structure of zip-file of an FMU
modelDescription.xml // Description of model (required file)
model.png // Optional image file of model icon
documentation // Optional directory containing the model
documentation
_main.html // Entry point of the documentation
<other documentation files>
sources // Optional directory containing all C-sources
// all needed C-sources and C-header files to compile and link the
model
exception of: fmiPlatformTypes.h and fmiFunctions.h
// with binaries // Optional directory containing the binaries
win32 // Optional binaries for 32-bit Windows
<modelIdentifier>.dll // DLL of the model interface implementation
Libraries for a particular compiler
// Optional object VisualStudio8 // Binaries for 32-bit Windows
generated with
// Microsoft Visual Studio 8 (2005)
<modelIdentifier>.lib // Binary libraries
gcc3.1 // Binaries for gcc 3.1
... win64 // Optional binaries for 64-bit Windows
...
linux32 // Optional binaries for 32-bit Linux
...
linux64 // Optional binaries for 64-bit Linux
... resources // Optional resources needed by the model
< data in model specific files which will be read during initialization
```

CosiMate – FMI comparison

- CosiMate Bus
 - Synchronization points based on a time step
 - Fixed time step
 - Multiple frequencies on the bus
 - Data exchange sequence
 - Send inputs to simulators/models
 - Synchronize (time step)
 - Read outputs from simulators/models
 - Data prediction/calculation
 - Extrapolation algorithm
 - User algorithms can be used
 - Controls the tools (start, shutdown, etc...)
 - Remote server for distribution
- FMI Master
 - Communication points based on a time step
 - Fixed and variable time steps
 - Multiples frequencies on the bus
 - Data exchange sequence
 - Send inputs to slaves
 - Do the step
 - Receive slaves outputs
 - Data prediction/calculation
 - Not yet in the standard
 - Controls the slaves (start, shutdown, etc...)
 - No solution provided
 - **Reminder:** master implementation may differ from a SW provider to another.

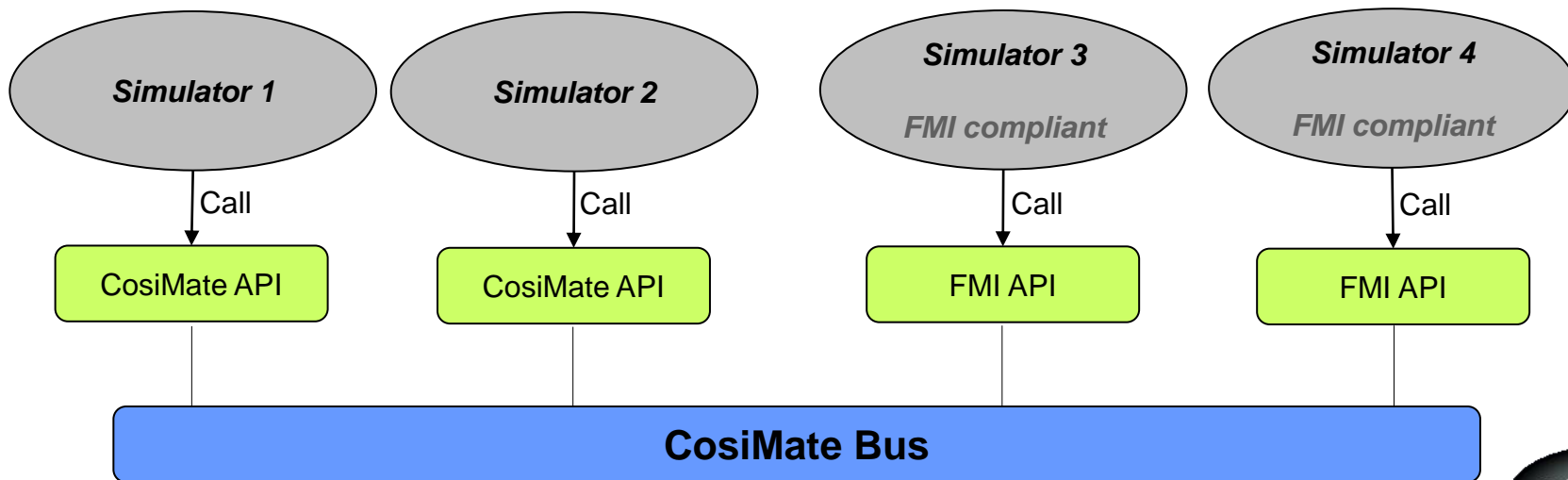
CosiMate Use Case #1: FMU Import

- Creation of a FMU simulator
 - To load and run the FMU model (DLL)
 - Bridge between the CosiMate API and the FMI API
- FMI for Model Exchange



CosiMate Use Case #2: FMI Master

- CosiMate is the FMI Master
 - FMI API implementation
 - Does not require external FMU simulator
- FMI for Co-Simulation



FMI support in tools

- AMESim (FMU export and import)
- Dymola 7.4 (FMU export and import)
- EXITE ACE (FMU export and import)
- EXITE (FMU import)
- FMU SDK (FMU export and import)
- JModelica.org (FMU export and import)
- NI VeriStand (**FMU Co-Simulation**)
- NI LabVIEW (FMU import)
- Silver 2.0 (FMU import)
- SIMPACK (FMU import)
- SimulationX 3.4 (FMU export and import, **FMU Co-Simulation**)
- Simulink (FMU export now available by Dymola 7.4 via Real-Time Workshop)
- Simulink (FMU export now available via @Source)
- TISC (FMU import)
- ... and **CosiMate 6.x** (FMU import, **FMU Co-Simulation**)