Functional Mockup Interface 2.0 and HiL Applications

New Features of FMI 2.0 and beyond

Torsten Blochwitz
R&D Manager, ITI Dresden

FMI MAP Leader

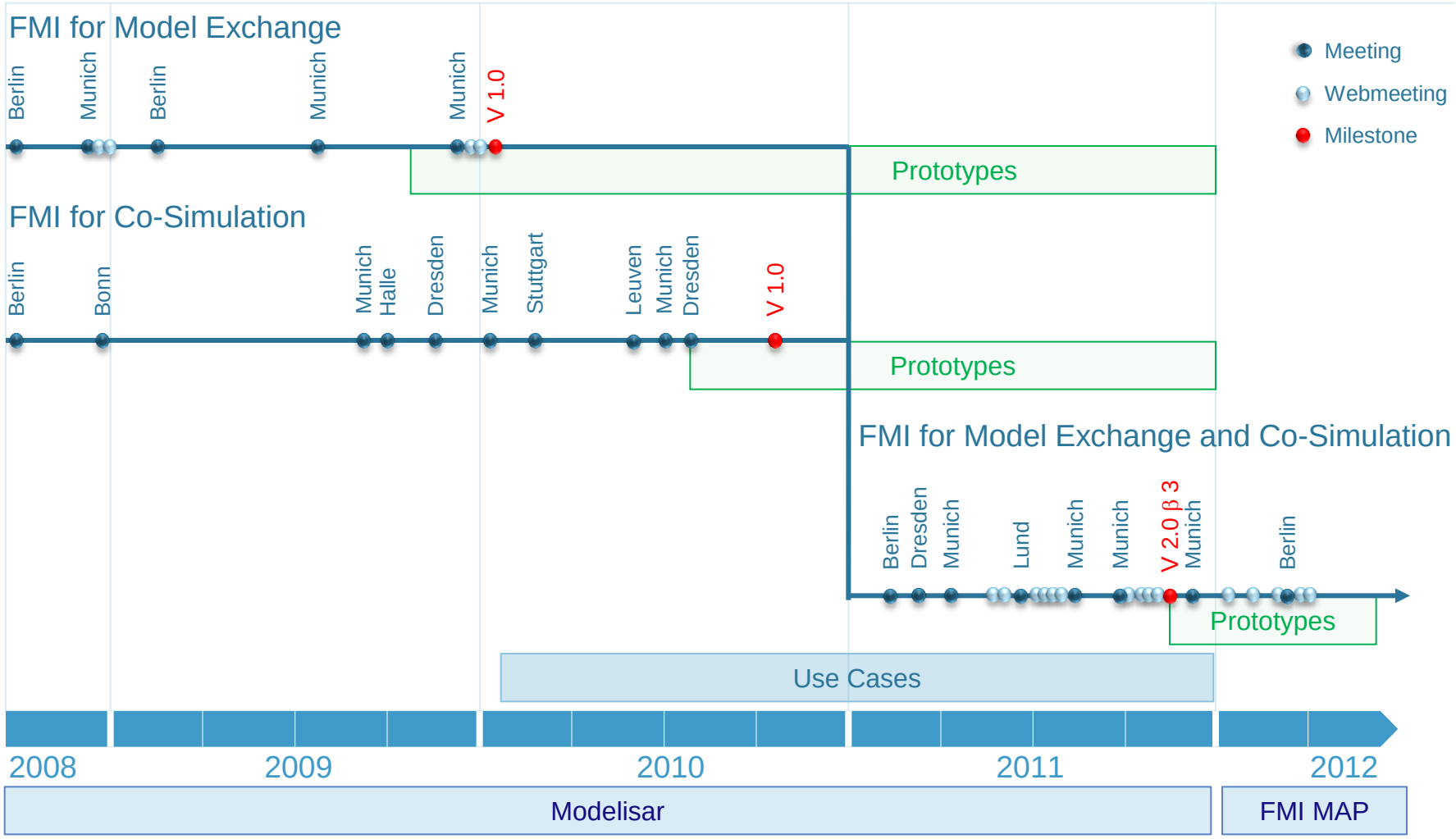
© 2014 FMI Modelica Association Project | www.fmi-standard.org

New Features of FMI 2.0 and beyond

- History
- New Features of FMI 2.0
- Current Status
- Future Plans

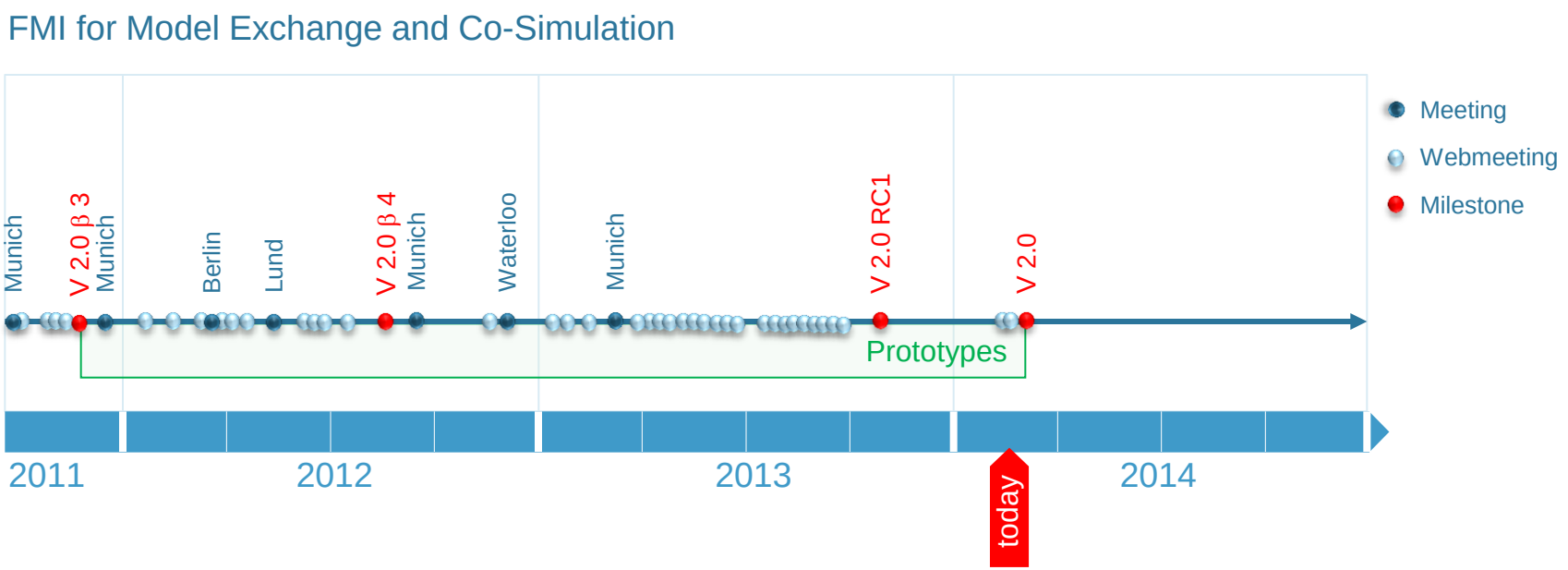
© 2014 FMI Modelica Association Project | www.fmi-standard.org

Development Process



© 2014 FMI Modelica Association Project | www.fmi-standard.org

Development Process



FMI 2.0:

- Valid for several years
- Backwards-compatible enhancements in minor releases

© 2014 FMI Modelica Association Project | www.fmi-standard.org

New features of FMI 2.0

Clarification of specification:

- Instantiation
- Classification of variables
- Mathematical model
- Calling sequence

Features:

- Improved initialization
- Semantics of event handling
- Tunable parameters
- Improved unit handling
- Save and restore FMU state
- Detailed dependency information (inputs, outputs, derivatives)
- Efficient interface to Jacobian matrices

© 2014 FMI Modelica Association Project | www.fmi-standard.org

FMI 2.0
Harmonization

FMI 1.0:

- `fmiInstantiateModel, fmiInstantiateSlave`

FMI 2.0:

- `fmiInstantiate(…, fmiType fmuType,…)`
- With `fmiType: fmiModelExchange` or `fmiCoSimulation`
- Allows for extensions without disturbing existing semantics
- E.g.: Hybrid Co-Simulation, Observers

© 2014 FMI Modelica Association Project | www.fmi-standard.org

FMI 2.0
Classification of interface variables

`causality`

- `parameter`
- `input`: output of another model
- `output`: input for another model
- `local`: not to be used by other models

`variability`

- `constant`
- `fixed`: constant after initialization
- `tunable`: constant between events
- `discrete`: changes at event instances
- `continuous`

- Combination of `causality` and `variability` allows clear classification of all kinds of variables

- New: distinction between `tunable` and `fixed parameters`
  - Stop simulation, set tunable parameters, resume simulation

© 2014 FMI Modelica Association Project | www.fmi-standard.org

FMI 2.0
Initialization, Events

FMI 1.0:

- Only one function call for initialization: `fmiInitialize`
- No iteration during initialization possible
- Critical, if initial conditions depend on variables in algebraic loops

FMI 2.0:

- Introduction of “Initialization Mode” (`fmiEnterInitializationMode`, `fmiExitInitializationMode`)
- Solution of algebraic loops by calling of `fmiSetXX`, `fmiGetXX` in the same way as in “Continuous Time Mode”
- For Model Exchange and Co-Simulation
- Introduction of “Event Mode” and a semantics for enter, exit and setting of new discrete state

© 2014 FMI Modelica Association Project | www.fmi-standard.org

FMI 2.0
Save and Restore FMU State

- FMI 1.0: implicite save and restore depending on arguments of `fmiDoStep`
- FMI 2.0: explicite function calls for Model Exchange and Co-Simulation

```
fmiStatus fmiGetFMUstate(fmiComponent c, fmiFMUstate* FMUstate)
fmiStatus fmiSetFMUstate(fmiComponent c, fmiFMUstate  FMUstate)
```

- Iterative Co-Simulation algorithms
  - Repeat more than one communication step
- Model Predictive Control
  - Simulate some steps starting from the same state with different sets of input values
  - Use the optimal set as control value for the real system
- FMU state can be serialized into a byte vector
  - Usage: start a training simulator from a certain scenario

© 2014 FMI Modelica Association Project | www.fmi-standard.org

FMI 2.0
Dependency Information

FMI 1.0:

- Only dependencies of outputs on inputs can be indicated

FMI 2.0:

- Dependencies of outputs on continuous states
- Dependencies of derivatives on continuous states and inputs

Usage:

- Detection of algebraic loops
- Definition of sparsity pattern of Jacobian matrices

© 2014 FMI Modelica Association Project | www.fmi-standard.org

FMI 2.0
Dependency Information

Kind of dependency is also defined:

- `nonlinear`: Jacobian entry is not constant
- `fixed`: Jacobian entry is constant
- `discrete`: Jacobian entry may change after events

Allows optimizations:

- Generate linear systems of equations for solution of algebraic loops if possible
- Reduce number of Jacobian computations

© 2014 FMI Modelica Association Project | www.fmi-standard.org

FMI 2.0
Partial Derivatives, Jacobian Matrices

- Jacobians are needed for:
  - Implicit integration methods
  - Solution of systems of equations resulting from algebraic loops
  - Linearization of FMU
  - Extended Kalman filters
- Numerical computation is expensive for large models
- Optional function for obtaining directional derivatives
  - `fmiGetDirectionalDerivative`
  - Allows obtaining of single directional derivatives, spares or dense Jacobian

© 2014 FMI Modelica Association Project | www.fmi-standard.org

Current Status

- October 2013: Release of FMI 2.0 Release Canditate1
- FMI 2.0 will be released as soon as test cases are passed by a certain number of tools
- 8 tool vendors are involved in testing:
  - **AVL**: Import of Model Exchange and Co-Simulation FMUs, export in in development
  - **Dassault (Dymola)**: Exported test FMUs are available on SVN, FMU test library runs successful
  - **dSPACE**: Successful running of Co-Simulation FMUs (Dymola, **MapleSim**, SimulationX) in source code and binary form in SCALEXIO and VEOS
  - **ITI (SimulationX):** Exported Test FMUs are available on SVN, import is in development
  - **QTronic**: Import of FMUs from SimulationX into Silver, Release of FMU SDK 2.0.0
  - **Modelon**: FMU Checker for FMI 2.0 RC1 is available, FMI library is updated, FMI Export for JModelica is nearly finished

Current Status

- No major concerns have been reported until now
- Some tickets for minor improvement of specification have been filed

- FMI Design meetings are scheduled for the next weeks:
  - Presentation and discussion of test results
  - Working on filed tickets
  - Decision about release

- Offers for FMI-logo have been from seven international design studios

Future Plans

- Backwards-compatible extensions (FMI 2.1):
  - Handling of clocked and sampled data systems
  - Hybrid Co-Simulation
  - Observers
  - Support of arrays
  - Hierarchical (structured) data
  - Bus and physical connectors
  - DAE representation
  - Graphical appearance

© 2014 FMI Modelica Association Project | www.fmi-standard.org