

Towards an unified OpenModelica Simulation Interface - OMSI

Current development status

Andreas Heuermann¹ Willi Braun¹ Niklas Worschech² Bernhard Bachmann¹

FH Bielefeld University of Applied Sciences ¹
Bielefeld, Germany

Bosch Rexroth ²
Lohr, Germany



Rexroth
Bosch Group

February 2, 2019

SPONSORED BY THE



Federal Ministry
of Education
and Research

PARADOM

- 1 Motivation
- 2 OMSI structure
 - New C structure
 - SimCode and Templates
 - Call structure for OMSIC and OMSICpp
 - Solver interface
 - OMSI simulation runtime
- 3 Current development status
 - Implemented features
 - Modelica Standard Library Coverage
- 4 Summary



```
uniontype SimCode
  "Root data structure containing information required for
  templates to generate simulation code for a Modelica
  model."
record SIMCODE
  ...
  list <DAE.Constraint> constraints;
  list <DAE.ClassAttributes> classAttributes;
  list <BackendDAE.ZeroCrossing> zeroCrossings;
  list <BackendDAE.ZeroCrossing> relations "only used by c
  runtime";
  list <BackendDAE.TimeEvent> timeEvents "only used by c runtime
  yet";
  list <DAE.ComponentRef> discreteModelVars;
  ExtObjInfo extObjInfo;
  SimCodeFunction.MakefileParams makefileParams;
  DelayedExpression delayedExps;
  list <JacobianMatrix> jacobianMatrixes;
  Option<SimulationSettings> simulationSettingsOpt;
  String fileNamePrefix, fullPathPrefix "Used in FMI where
  files are generated in a special directory";
  String fmuTargetName;
  HpcOmSimCode.HpcOmData hpcOmData;
  ...
```

- SimCode data structure got bloated over time
- SimCode not independent of target language
- No clear separation between model and runtime data



C runtime

ModelicaTest_3.2.2 test using OpenModelica

| Total | Frontend | Backend | SimCode | Templates | Compilation | Simulation | Verification |
|-------|----------|---------|---------|-----------|-------------|------------|--------------|
| 519 | 514 | 512 | 512 | 512 | 512 | 496 | 452 |

Total time taken: 1:32:57

System info: Intel(R) Core(TM) i7-6900K CPU @ 3.20GHz, 126 GB RAM, Ubuntu 18.04.1 LTS

OpenModelica Version: OMCompiler v1.14.0-dev.59+g683050ef1

Test started: 2019-01-19 22:19:07

Tested Library: 3.2.2

C++ runtime

ModelicaTest_3.2.2_c++ test using OpenModelica

| Total | Frontend | Backend | SimCode | Templates | Compilation | Simulation | Verification |
|-------|----------|---------|---------|-----------|-------------|------------|--------------|
| 519 | 514 | 512 | 512 | 512 | 508 | 475 | 369 |

Total time taken: 2:06:21

System info: Intel(R) Core(TM) i7-6900K CPU @ 3.20GHz, 126 GB RAM, Ubuntu 18.04.1 LTS

OpenModelica Version: OMCompiler v1.14.0-dev.59+g683050ef1

Test started: 2019-01-19 22:19:07

Tested Library: 3.2.2

- New features require development for each runtime
- ⇒ Different performance of runtimes

Goal: Make developer life more simple

- Unify data structures in back-end and simulation runtimes
- Share code generation for C and C++
- Clear interface and shared functionalities
 - ▶ Base: Use common base functionalities and solvers
 - ▶ C-Runtime: Simulation runtime in ANSI C for e.g. realtime applications
 - ▶ Cpp-Runtime: Simulation runtime for e.g desktop applications



Goal: Make developer life more simple

- Unify data structures in back-end and simulation runtimes
- Share code generation for C and C++
- Clear interface and shared functionalities
 - ▶ Base: Use common base functionalities and solvers
 - ▶ C-Runtime: Simulation runtime in ANSI C for e.g. realtime applications
 - ▶ Cpp-Runtime: Simulation runtime for e.g desktop applications



Goal: Make developer life more simple

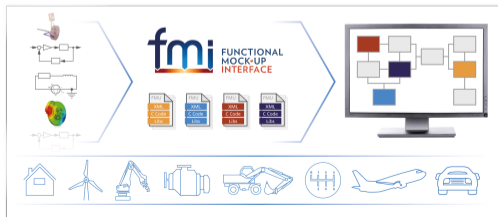
- Unify data structures in back-end and simulation runtimes
- Share code generation for C and C++
- Clear interface and shared functionalities
 - ▶ Base: Use common base functionalities and solvers
 - ▶ C-Runtime: Simulation runtime in ANSI C for e.g. realtime applications
 - ▶ Cpp-Runtime: Simulation runtime for e.g desktop applications

Goal: Make developer life more simple

- Unify data structures in back-end and simulation runtimes
- Share code generation for C and C++
- Clear interface and shared functionalities
 - ▶ Base: Use common base functionalities and solvers
 - ▶ C-Runtime: Simulation runtime in ANSI C for e.g. realtime applications
 - ▶ Cpp-Runtime: Simulation runtime for e.g desktop applications

Free extra

- Better FMI support





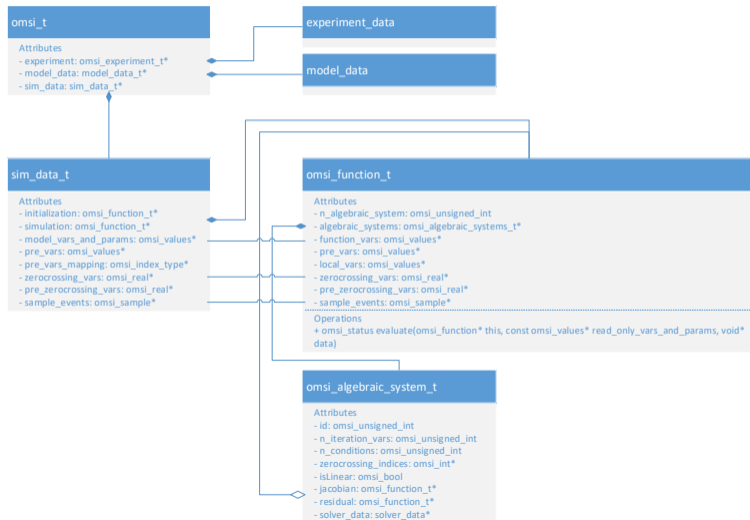
OpenModelica Simulation Interface (OMSI)

- 1 Motivation
- 2 OMSI structure
 - New C structure
 - SimCode and Templates
 - Call structure for OMSIC and OMSICpp
 - Solver interface
 - OMSI simulation runtime
- 3 Current development status
 - Implemented features
 - Modelica Standard Library Coverage
- 4 Summary



Targets

- Separate data to smaller segments
- Store data thread-safe
- Forward only necessary informations
- Don't copy to much memory





- Include new OMSIData in SimCode

```
uniontype OMSIData()
--"contains data for code generation for OMSI"()
--record OMSI_DATA()
----OMSIFunction initialization "contains equations and variables for initialization problem";()
----OMSIFunction simulation "contains equations and variables for simulation problem";()
--end OMSI_DATA;()
end OMSIData;()
()
uniontype OMSIFunction()
--"contains equations and variables for initialization or simulation problem"()
--record OMSI_FUNCTION()
----list<SimEqSystem> equations "list of single equations and systems of equations";()
----list<SimCodeVar.SimVar> inputVars "list of simcode variables determining input variables for equation(s)";()
----list<SimCodeVar.SimVar> outputVars "list of simcode variables determining output variables for equation(s)";()
----list<SimCodeVar.SimVar> innerVars "list of simcode variables determining inner variables for equation(s), e.g. $DER(x)";()
----Integer nAllVars "number of input, inner and output vars";()
----SimCodeFunction.Context context "contains crefToSimVar hash table for lookup function in templates";()
----Integer nAlgebraicSystems "number of linear and non-linear algebraic systems in OMSI_FUNCTION.equations";()
--end OMSI_FUNCTION;()
end OMSIFunction;()
```

- Shared functions for code generation



- Include new OMSIData in SimCode

```
uniontype OMSIData¶
  ·· "contains data for code generation for OMSI"¶
  ·· record OMSI_DATA¶
    ··· OMSIFunction initialization "contains equations and variables for initialization problem";¶
    ··· OMSIFunction simulation "contains equations and variables for simulation problem";¶
    ·· end OMSI_DATA;¶
end OMSIData;¶
¶
uniontype OMSIFunction¶
  ·· "contains equations and variables for initialization or simulation problem"¶
  ·· record OMSI_FUNCTION¶
    ··· list<SimEqSystem> equations ··· "list of single equations and systems of equations";¶
    ··· list<SimCodeVar.SimVar> inputVars ··· "list of simcode variables determining input variables for equation(s)";¶
    ··· list<SimCodeVar.SimVar> outputVars ··· "list of simcode variables determining output variables for equation(s)";¶
    ··· list<SimCodeVar.SimVar> innerVars ··· "list of simcode variables determining inner variables for equation(s), e.g. $DER(x)";¶
    ··· Integer nAllVars ··· "number of input, inner and output vars";¶
    ··· SimCodeFunction.Context context ··· "contains crefToSimVar hash table for lookup function in templates";¶
    ··· Integer nAlgebraicSystems ··· "number of linear and non-linear algebraic systems in OMSI_FUNCTION.equations";¶
    ·· end OMSI_FUNCTION;¶
end OMSIFunction;·¶
```

- Shared functions for code generation



- Include new OMSIData in SimCode

```
uniontype OMSIData¶
  ··· "contains data for code generation for OMSI"¶
  ··· record OMSI_DATA¶
    ··· OMSIFunction initialization "contains equations and variables for initialization problem";¶
    ··· OMSIFunction simulation "contains equations and variables for simulation problem";¶
  ··· end OMSI_DATA;¶
end OMSIData;¶
¶
uniontype OMSIFunction¶
  ··· "contains equations and variables for initialization or simulation problem"¶
  ··· record OMSI_FUNCTION¶
    ··· list<SimEqSystem> equations ··· "list of single equations and systems of equations";¶
    ··· list<SimCodeVar.SimVar> inputVars ··· "list of simcode variables determining input variables for equation(s)";¶
    ··· list<SimCodeVar.SimVar> outputVars ··· "list of simcode variables determining output variables for equation(s)";¶
    ··· list<SimCodeVar.SimVar> innerVars ··· "list of simcode variables determining inner variables for equation(s), e.g. $DER(x)";¶
    ··· Integer nAllVars ··· "number of input, inner and output vars";¶
    ··· SimCodeFunction.Context context ··· "contains crefToSimVar hash table for lookup function in templates";¶
    ··· Integer nAlgebraicSystems ··· "number of linear and non-linear algebraic systems in OMSI_FUNCTION.equations";¶
  ··· end OMSI_FUNCTION;¶
end OMSIFunction;¶
```

- Shared functions for code generation



Generate some code

- Use common functions in templates
- Files for `omsi_function` and `omsi_alg_systems`
- Generate comparable code for equations



Generate some code

- Use common functions in templates
- Files for `omsi_function` and `omsi_alg_systems`
- Generate comparable code for equations



Generate some code

- Use common
- Files for omsi
- Generate com

```
helloWorldOMSI_omsi.c | helloWorldOMSI_sim_eqns.c | helloWorldOMSI_init_eqns.c
72 //~
73 /* Evaluation functions for each equation *///~
74 //~
75 equation index: 3//~
76 type: SIMPLE_ASSIGN//~
77  $der(x) = a * x$ //~
78 //~
79 void helloWorldOMSI_eqFunction_3(omsi_function_t* this_function, const omsi_values* model_vars_and_params){//~
80 --this_function->function_vars->reals[1] /* der(x) STATE_DER *///~
81 -- = (model_vars_and_params->reals[2] /* a PARAM */) * (this_function->function_vars->reals[0] /* x STATE(1) */);//~
82 }//~
83 //~
84 //~
85 //~
86 /* Equations evaluation *///~
87 omsi_status helloWorldOMSI_sim_eqns_allEqns(omsi_function_t* sim_eqns, omsi_values* model_vars_and_params, void* data){//~
88 //~
89 --/* Variables *///~
90 --omsi_status status;//~
91 //~
92 --status = omsi_ok;//~
93 //~
94 --helloWorldOMSI_eqFunction_3(sim_eqns, model_vars_and_params);//~
95 //~
96 --return status;//~
97 }//~
98
```

```
simpleNonLinLoop_0_init_eqns.c | simpleNonLinLoop_0_sim_eqns.c | simpleNonLinLoop_0_sim_eqns_algSyst_0.c | simpleNonLinLoop_0_sim_eqns_derMat_0.c
156 }
157 //
158 /* Algebraic system evaluation */
159 //
160 equation index: 10
161 type: ALGEBRAIC_SYSTEM
162 is linear: false
163 depending functions indices: 13, 12, 11
164 dimension: 3
165 iteration vars: b._SeedNLSJac3, a._SeedNLSJac3
166 //
167 c._$pDERNLSJac3._dummyVarNLSJac3 = (-a.SeedNLSJac3) - (-1.5) * b.SeedNLSJac3
168 $res1._$pDERNLSJac3._dummyVarNLSJac3 = 2.0 * (c * c._$pDERNLSJac3._dummyVarNLSJac3 + b * b.SeedNLSJac3 + a * a.SeedNLSJac3)
169 $res2._$pDERNLSJac3._dummyVarNLSJac3 = a.SeedNLSJac3 + b.SeedNLSJac3 + c._$pDERNLSJac3._dummyVarNLSJac3
170 //
171 omsi_status simpleNonLinLoop_0_sim_eqns_algSystFunction_0(omsi_algebraic_system_t* this_alg_system,
172     const omsi_values* model_vars_and_params, void* data){
173 //
174 ///* Variables */
175 // omsi_status status;
176 //
177 ///* Log function call */
178 // filtered_base_logger(global_logCategories, log_all, omsi_ok,
179 //     "fmi2Evaluate: Solve algebraic system 0.");
180 //
181 ///* call API function something */
182 // status = omsi_solve_algebraic_system(this_alg_system, model_vars_and_params);
183 //
184 // return status;
185 }
```

Generate s

- Use co
- Files fo
- Genera



Generate some code

- Use common functions in templates
- Files for `omsi_function` and `omsi_alg_systems`
- Generate comparable code for equations



Generate some code

- Use common functions in templates
- Files for `omsi_function` and `omsi_alg_systems`
- Generate comparable code for equations

Overview

Templates

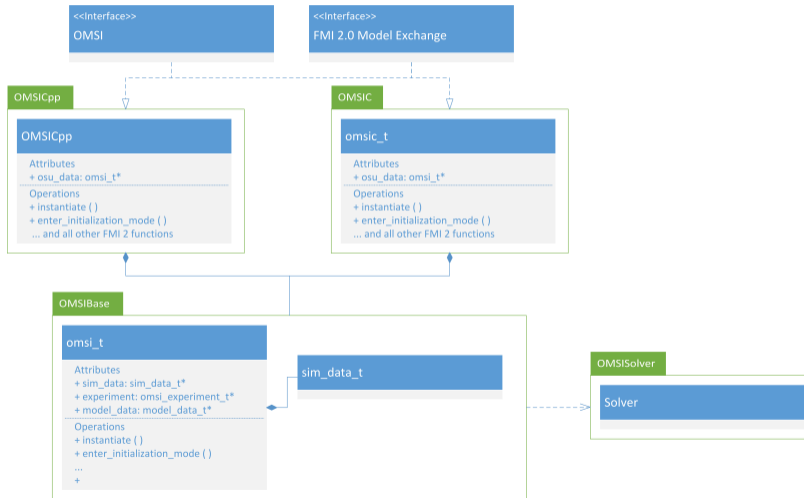


```
bouncingBall_sim_eqns.c
194 /*
195 equation_index: 23
196 type: SIMPLE_ASSIGN
197 SubenCondition3 = h <= 0.0 and v <= 0.0
198 */
199 void bouncingBall_eqFunction_23(oms_i_function_t* this_function,
200                               const oms_i_values* model_vars_and_params){
201     /* Variables */
202     oms_i_bool tmp3;
203     oms_i_bool tmp4;
204
205     tmp3 = this_function->function_vars->reals[0] /* h STATE(1,v) */ <= 0.0;
206     tmp4 = this_function->function_vars->reals[1] /* v STATE(1) */ <= 0.0;
207     this_function->function_vars->bools[2] /* SubenCondition3 DISCRETE */ =
208         (oms_i_function_zero_crossings(this_function, tmp3, 0, oms_i_get_model_state()) &&
209          && oms_i_function_zero_crossings(this_function, tmp4, 1, oms_i_get_model_state()));
210 }
211
212 /* Equations evaluation */
213 oms_i_status bouncingBall_sim_eqns_allEqns(oms_i_function_t* sim_eqns,
214                                           const oms_i_values* model_vars_and_params,
215                                           void* data){
216
217     /* Variables */
218     oms_i_status status, new_status;
219
220     status = oms_i_ok;
221     new_status = oms_i_ok;
222
223     bouncingBall_eqFunction_15(sim_eqns, model_vars_and_params);
224     bouncingBall_eqFunction_16(sim_eqns, model_vars_and_params);
225     bouncingBall_eqFunction_17(sim_eqns, model_vars_and_params);
226     bouncingBall_eqFunction_18(sim_eqns, model_vars_and_params);
227     bouncingBall_eqFunction_19(sim_eqns, model_vars_and_params);
228     bouncingBall_eqFunction_20(sim_eqns, model_vars_and_params);
229     bouncingBall_eqFunction_21(sim_eqns, model_vars_and_params);
230     bouncingBall_eqFunction_22(sim_eqns, model_vars_and_params);
231     bouncingBall_eqFunction_23(sim_eqns, model_vars_and_params);
232
233     return status;
234 }
235
```

```
OMCcppBouncingBallOMSEquations.cpp
200 /*
201 equation_index: 23
202 type: SIMPLE_ASSIGN
203 SubenCondition3 = h <= 0.0 and v <= 0.0
204 */
205 void BouncingBall::oms_i_evaluate_23(oms_i_function_t* this_function,
206                                     const oms_i_values* model_vars_and_params){
207     /* Variables */
208     oms_i_bool tmp3;
209     oms_i_bool tmp4;
210
211     tmp3 = this_function->function_vars->reals[0] /* h STATE(1,v) */ <= 0.0;
212     tmp4 = this_function->function_vars->reals[1] /* v STATE(1) */ <= 0.0;
213     this_function->function_vars->bools[2] /* SubenCondition3 DISCRETE */ =
214         (oms_i_function_zero_crossings(this_function, tmp3, 0, oms_i_get_event_mode()) &&
215          && oms_i_function_zero_crossings(this_function, tmp4, 1, oms_i_get_event_mode()));
216 }
217
218 /* Equations evaluation */
219 oms_i_status BouncingBall::oms_i_evaluateAll(oms_i_function_t* sim_eqns,
220                                             const oms_i_values* model_vars_and_params,
221                                             void* data){
222
223     /* Variables */
224     oms_i_status status, new_status;
225
226     status = oms_i_ok;
227     new_status = oms_i_ok;
228
229     oms_i_evaluate_15(sim_eqns, model_vars_and_params);
230     oms_i_evaluate_16(sim_eqns, model_vars_and_params);
231     oms_i_evaluate_17(sim_eqns, model_vars_and_params);
232     oms_i_evaluate_18(sim_eqns, model_vars_and_params);
233     oms_i_evaluate_19(sim_eqns, model_vars_and_params);
234     oms_i_evaluate_20(sim_eqns, model_vars_and_params);
235     oms_i_evaluate_21(sim_eqns, model_vars_and_params);
236     oms_i_evaluate_22(sim_eqns, model_vars_and_params);
237     oms_i_evaluate_23(sim_eqns, model_vars_and_params);
238
239     return status;
240 }
241
```

Overview

Call structure for OMSIC and OMSICpp





OMSISolver

<<Interface>>

Solver Api

Attributes

+ solver_callbacks

Operations

+ allocate ()

+ free ()

+ prepare_specific_data ()

+ set_start_vector ()

+ set_solver ()

+ solve_system ()

+ getXXX ()

+ setXXX ()

Solver

Attributes

+ name: string

+ is_linear: bool

- specific_solver_data

Operations

+ solve_system ()

solver_specific_functions ()

LAPACK Solver

Attributes

name = "solver_lapack"

is_linear = true

- data: LAPACK_DATA

Operations

#

Homotopy Solver

Attributes

name = "solver_homotopy"

is_linear = false

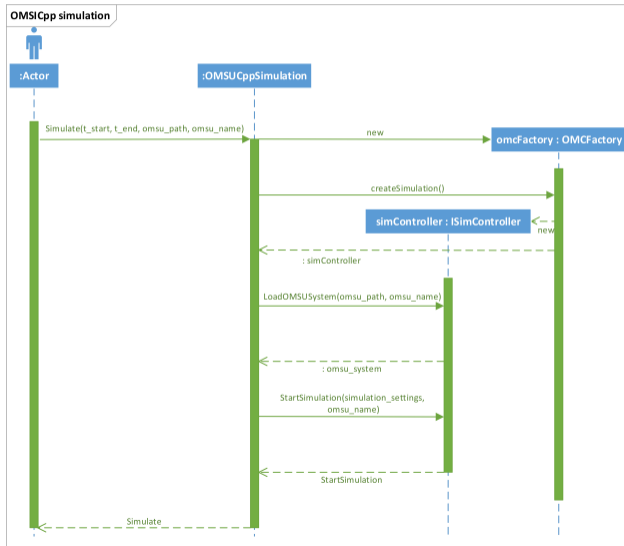
- data: HOMOTOPY_DATA

Operations



OMSU Cpp simulation runtime

- Simulate OMSU or FMI 2.0 Model Exchange FMU
- Optional arguments passed to simulation executable, e.g. experiment settings
- Aim to use OMSI ODE solver



- 1 Motivation
- 2 OMSI structure
 - New C structure
 - SimCode and Templates
 - Call structure for OMSIC and OMSICpp
 - Solver interface
 - OMSI simulation runtime
- 3 Current development status
 - Implemented features
 - Modelica Standard Library Coverage
- 4 Summary



What's working at the moment

- Code generation equations and equation systems
- Solving of linear and non-linear algebraic loops
- Event handling
- Exporting of Model Exchange FMU's



Modelica Standard Library - 3.2.2

- Build OMSIC FMU and import with OpenModelica to generate simulation executable
- Tested on Ubuntu Bionic (18.04)
- First models are building and simulating
- Just started, great improvements are to be expected



Next steps

- Complete code generation for all equations and equation systems
- Validate results automatically
- Test OMSICpp
- Merge first version into Modelica master
- Increase number of usable solvers on OMSISolver library



- 1 Motivation
- 2 OMSI structure
 - New C structure
 - SimCode and Templates
 - Call structure for OMSIC and OMSICpp
 - Solver interface
 - OMSI simulation runtime
- 3 Current development status
 - Implemented features
 - Modelica Standard Library Coverage
- 4 Summary

Summary

- Unified data structures in SimCode, Templates and runtimes
- Reduction of maintenance work
- Shared solver library
- Support of FMI 2.0 (ModelExchange)
- First working simulations

Open tasks

- Complete SimCode and Templates
- Reach high MSL coverage
- Switch from "old" C and Cpp runtime to OMSI runtime
- Exploit parallelism

Summary

- Unified data structures in SimCode, Templates and runtimes
- Reduction of maintenance work
- Shared solver library
- Support of FMI 2.0 (ModelExchange)
- First working simulations

Open tasks

- Complete SimCode and Templates
- Reach high MSL coverage
- Switch from "old" C and Cpp runtime to OMSI runtime
- Exploit parallelism

Thank you for your attention!

Summary

- Unified data structures in SimCode, Templates and runtimes
- Reduction of maintenance work
- Shared solver library
- Support of FMI 2.0 (ModelExchange)
- First working simulations

Open tasks

- Complete SimCode and Templates
- Reach high MSL coverage
- Switch from "old" C and Cpp runtime to OMSI runtime
- Exploit parallelism

Questions?