Co-Simulation between detailed building energy performance simulation and Modelica HVAC component models

Andreas Nicolai[1] Anne Paepcke[1]

[1]Institut for Building Climatology, Faculty of Architecture, TU Dresden, Germany,
andreas.nicolai@tu-dresden.de

Abstract

We discuss the application of the FMI Co-Simulation technology to building energy performance simulation, where detailed physical building models are coupled to Modelica-based HVAC component and plant models. First, we describe the generation process of the building FMU from our stand-alone building simulation program NANDRAD and sketch out internal algorithms for FMI version 2 capabilities. Then, coupling scenarios are described and physical interface conventions are presented. Usability is addressed by automatic generation of building-model specific adapters and wrappers. The building FMU and plant FMUs are then simulated together using different Co-Simulation master algorithms. Finally, based on simulation results and performance analysis we conclude with recommendations on suitable master algorithm options and specific features of suitable building FMUs.

Keywords: FMI, Co-Simulation, Energy, Building Simulation, HVAC System, Physical Interface, Master Algorithm

1 Introduction

Building energy performance simulation is a technology used by planners and building designers in the planning process. A typical usage scenario includes evaluation of different options regarding building envelope construction, HVAC systems and control strategies. Currently, available simulation tools, such as EnergyPlus TRNSYS (Klein et al., 1976; Dols et al., 2014), IDA-ICE (Sahlin et al., 2004) and our own development NANDRAD (Nicolai and Paepcke, 2012; Paepcke and Nicolai, 2014) (in C/C++) are concepted as stand-alone tools. Modeling and simulation of integrated modern buildings requires flexible plant and equipment models, which are often case-specific. Extending the source code of existing building simulation models is often only possible by original model developers and also very difficult and time consuming.

Alternatively, Modelica as one example for a modeling language can be used to express such equipment and control systems. There are a number of libraries providing suitable components for modeling building systems, for example the Annex60-based libraries AixLib, BuildingSystems, Buildings and Idias (Wetter et al., 2013; Wetter, 2009; Nytsch-Geusen et al., 2013; Sahlin et al., 2004) or the GreenBuilding library[1]. However, modeling the entire building with sufficient physical detail in Modelica alone is not meaningful for several reasons:

- larger building complexes may involve many zones, constructions, facade elements, thermal storage members resulting in thousands of differential equations,
- Modelica code may become huge and may cause problems with the generic Modelica solvers, even symbolic analysis may be extremely slow,
- modeling the building in Modelica without suitable BIM-style data import or code generation will not be possible for realistic buildings, it is too time-consuming and thus too expensive, and
- manual connection of many building components with corresponding equipment and control models may be extremely time-consuming and error-prone.

For practical purposes, planners and engineers will not accept a procedure that involves creation of such complex models with current Modelica user interfaces, alone. There are, however, tools under development that assist with prototyping Modelica-based building and equipment models, for example TEASER[2]. However, limitations with respect to the detail of the building model and simulation efficiency persist.

1.1 Benefits of Simulation Coupling within the Building Energy Simulation Context

The use of stand-alone simulation tools or Modelica-only based building modeling may not be a satisfying strategy. Instead, a hybrid approach appears meaningful:

- using existing building simulation software tailored to the building engineering user group, preferably Building Information Model (BIM) preprocessing software packages (DesignBuilder[3], BIM-HVAC

[1]Green City/ SimulationX – Planungstool, http://www.ea-energie.de/de/products/green-city-simulationsbibliothek-2-2

[2]TEASER - Tool for Energy Analysis and Simulation for Efficient Retrofit, https://github.com/RWTH-EBC/TEASER

[3]https://www.designbuilder.co.uk

tool[4], etc.) with database support, graphical representation of the building, and input error control with automatic generation of input data to building simulation engines (e.g. IDF-files for EnergyPlus, or nandrad-files for NANDRAD), and

- use of Modelica and suitable libraries by HVAC system planners to model building equipment (heater/chiller/ventilation systems) and required control strategies.

Joining both models in a coupled simulation will combine also the benefits of both modeling approaches. With the FMI standard a unified methodology and technical description for coupled simulation is available. With respect to the two described operation modes ModelExchange and Co-Simulation, we prefer the latter variant that allows individual FMUs to use their own dedicated solver engines. However, it can be expected that the Co-Simulation approach and gained flexibility implies a simulation overhead and performance penalty. In the remainder of the article we always refer to Co-Simulation according to the FMI standard when discussing coupled simulation.

1.2 Envisioned Usage of Co-Simulation

We envision two suitable scenarios of combining a dedicated building energy simulation FMU with one or more HVAC component FMUs created with Modelica. In the first scenario, the user will model the equipment system in Modelica and import a previously generated building FMU into the modeling environment, connect it to the Modelica components and run the simulation within the Environment (Figure 1).

Alternatively, HVAC component or control models may be designed with Modelica and than exported by the modeling tool into FMUs. These are then combined with the building FMU and simulated by an alternative Co-Simulation master. This approach allows prefabrication of HVAC component sub-models.

1.3 Co-Simulation Requirements

A central requirement for the application of Co-Simulation is that obtained results are of a similar accuracy as if the entire model would be calculated standalone. Accuracy shall be defined in this respect such that the global error, i.e. the difference between numerical solution and true solution is bounded to a defined limit. In practice, within each integration step the local error is controlled. Every FMU should implement such an error control algorithm to be considered a consistent model.

In the building energy simulation side, this demand restricts the choice of suitable simulation tools, for example, older simulation engines like EnergyPlus and TRNSYS do not implement such an error testing procedure. Our building simulation models THERAKLES and NANDRAD (Nicolai, 2013; Nicolai and Paepcke, 2012) belong to a class of modern solvers that use dynamic time step adjustment schemes based on local error estimates, with the advantage of maintaining required accuracy while improving simulation speed whenever possible (Hindmarsh et al., 2005). This is an important feature, since different building equipment may be active during different annual seasons and may enforce different time integration regimes. For example, heating systems are turned off during summer, and if air conditioning is not used, simulation can speed up since no interaction with actively controlled equipment occurs. Simulation time steps typically vary between 1 second and 30 minutes in annual simulations.

The requirement on error control made for FMUs should also be fulfilled by the Co-Simulation master, which effectively needs to adjust communication interval sizes. When separating control and equipment systems from the building's thermal response in a Co-Simulation scenario, the use of larger communication intervals may cause stability and accuracy problems. Such problems can be avoided by choosing a sufficiently small time step size. In realistic simulation cases it is generally not possible to predict the allowed maximum of the communication step size. Also, using a fixed tiny communcation step size leads to inacceptable long simulations and would limit the advantage of performance optimized FMU-internal solvers. Therefore, a master algorithm which supports error/stability control and dynamic adjustment of communication step sizes is desirable. This, in return, requires FMI Version 2.0 capabilities of the slaves, in particular the get and set state functionality (FMI, 2014).

Note, that an error control algorithm within a Co-Simulation master will also detect and compensate, by reducing communication step size, potential numerical instabilities, again leading to excessive and inacceptable simulation times. Phenomena of instability may grow with increased coupling strength of FMUs interface quantities and often arising from the choice of the model interface.

2 Choice of the FMU Interface

The separation of a complex building energy simulation model into subcomponents is not trivial. A natural choice for separation of the entire model into FMUs may be to keep the passive building and its physics regarding interaction with climate and user loads within the building simulation FMU. All active components such as heating, cooling, ventilation and associated equipment and control models will be in one ore more HVAC-FMUs. In this article we use a single FMU with all HVAC equipment and control models written in Modelica.

2.1 Building Simulation FMU Input/Output Variables

One option for a flexible interface would be to export all relevant states like temperatures and solar radiation loads from the building simulation FMU, and import calculated

[4] http://www.building-engineering.de

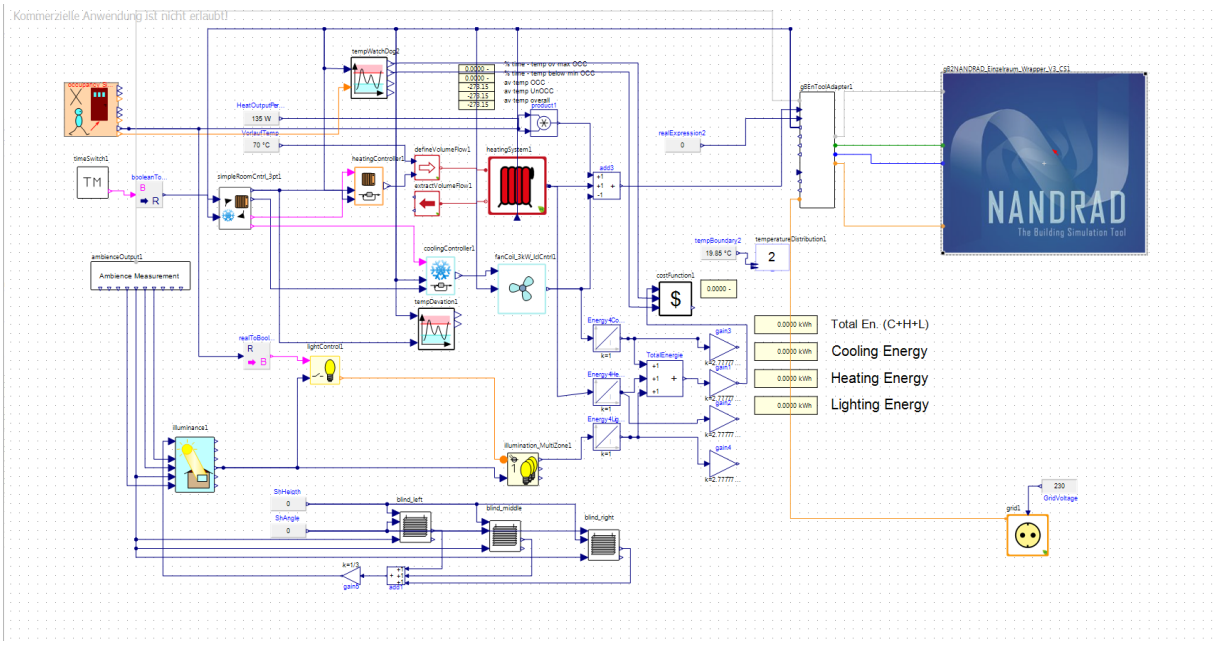

Figure 1. Usage Scenario 1: Building simulation FMU (NANDRAD) imported into Modelica environment (SimulationX)

heating/cooling loads from the HVAC FMUs. This interface can be considered a very universal interface, since any kind of heating/cooling loads can be modeled and imported as energy source to each thermal zone's energy balance. The interface defines for each thermal zone an export of mean air and operative temperature and input of convective and radiative thermal load.

The building simulation FMU includes databases for climatic loads and user behavior and related equipment schedules[5]. Hence, climatic data and schedules are additionally exported via the FMU interface. This allows consistent treatment of climatic input data in building and equipment models. Part of the scheduled user loads are also hot and cold water demand as well as user-related electric power consumption.

2.2 Convenience Adapters and Wrappers

The interface definition allows exporting and importing zonal quantities. Considering typical buildings of more than hundred conditioned zones, a large number of input/output variables need to be connected to the plant FMU. Even if the FMI standard would allow usage of vector variables, the manual connection of exported temperatures to the various input ports on the plant side would not be expedient and may lead to errors that are difficult to identify and track.

Also, when importing a building simulation FMU into a Modelica development environment the graphical representation of the inserted FMU with hundreds of ports is not suitable for practical use. Therefore, we utilize helper components that assist with mapping native FMU interface quantities to Modelica library ports and buses.

Different helper components are used depending on the usage scenario:

- When the building FMU is imported into the Modelica environment, the FMU is encapsulated into a Modelica wrapper model, which internally holds the FMU and connects to the native FMU interface. On the outside it provides port and bus connectors matching the corresponding library interfaces, in our case the GreenBuilding climate, electrical and HVAC buses (see Figure 2, we use the HVAC, HotWater and Electrical port of the GreenBuilding library). This wrapper is therefore specific to each building[6] and to the interfaced library.

- When the plant model is to be exported from Modelica into a stand-alone Co-Simulation FMU, the adapter (Figure 3) is used instead. It provides the same library-specific connectors as the wrapper, but does not connect to the building FMU. Instead, it exports and imports exactly the counterparts of the building FMU interface variables . When exporting the Modelica model, only these connectors become part of the FMU interface. Also, the connector counterparts are identically named to the building FMU interface quantities, which greatly simplifies automated connection between plant and building FMU connectors[7]. The graphical annotations of zonal con-

[5]Typically, such schedules and databases are part of the building model definition and will be generated/collected within the BIM process

[6]The native interface of the FMU changes with the number of thermal zones, or their IDs, and so does the wrapper component.

[7]Similarly, when importing a building FMU into a Modelica environment an automated matching of connectors between FMU and wrapper/adapter model would be possible. Unfortunately, none of the currently available modeling environments supports such a procedure.

nectors with same quantities but different zone references are arranged on top of each other, thus keeping the adapter symbol compact.

Figure 3 does not show the actual connector names, but rather a physical description and associated unit (see (Paepcke et al., 2016) for a complete specification).

2.2.1 Adapter/Wrapper Configurations

The use of wrappers/adapters is a compromise between flexibility of the building model interface and easy-of-use within Modelica environments. The current specification of our adapter/wrapper Modelica component is specialized of interfacing all building zones with exactly one HVAC system model in Modelica. For other situations, the adapter/wrapper models may look different. Yet, the principle approach to provide FMU-independent connectors for the remainder of the Modelica model appears to be a promising way to avoid connecting to individual FMU input/output variables directly.

3 Parametrization and Export of NANDRAD FMUs

3.1 Configuration for FMU Export

When NANDRAD is executed as stand-alone building energy simulation model, for example to compute annual energy demand and comfort criteria, it uses a set of input files with the building model (BIM) and database elements (material data, constructions, climatic data, etc.). The input data include definitions of all zones and their heating and cooling requirements, which enables an ideal heating and cooling load calculation.

When NANDRAD is used as building simulation FMU to simulate a realistic heating/cooling system, all conditioned zones need to be connected to the heating cycle or to the electrical grid of the plant model. All zones that are part of the interface and import/export variables are given different usage scenarios, for example, heating scenario or electrical usage scenario. This information is then used during export to generate required import/export quantities and also create the internal data structures that map FMU input/output variables to existing internal variables. Selecting the usage scenarios and selecting the corresponding zones is part of the FMU preprocessing.

3.2 Export procedure

The export procedure involves several steps:

- NANDRAD is run as stand-alone simulation to generate auxiliary information needed for parametrization of the HVAC/plant model, for example the heating/cooling design day calculation.
- The NANDRAD solver initialization is used to generate the variable dependency information, which is stored in the `modelDescription.xml` file.
- The `modelDescription.xml` is composed (included data for ModelExchange and Co-Simulation and the FMI v2 functionality).
- All referenced databases are collected. All input files, the pre-compiled NANDRAD dynamic library (with implemented FMI functionality), and additional dependent libraries[8] are copied. Finally, the FMU archive is created.
- Modelica wrapper and adapter models (.mo files) are generated individually for the current building project.
- A report including zone naming, dimensions, unique IDs and heating/cooling design loads is written to be used during configuration of the HVAC component model, and for automatic Modelica model generation scripts.

During export, compilation of source code is not necessary and the model initialization and the design day calculation are usually very fast, except for large buildings with several hundred of zones. The auxiliary files are provided seperately from the generated FMU.

[8]Depending on the target platform, different libraries are copied. Currently, one NANDAD FMU holds only binaries for one platform Win32, Win64, Linux64, Darwin64 at a time.

Figure 2. Modelica wrapper encapsulates NANDRAD FMU and provides collector ports for climate, HVAC and electrical quantities

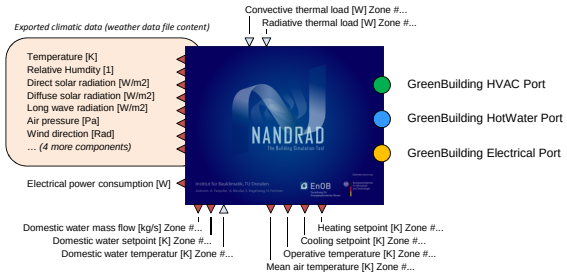

Figure 3. NANDRAD adapter provides Modelica collector ports as well as input and output variables identically named as the building FMU ports

Modeling modern complex integrated buildings will require much more data being commonly used by the building simulation model and HVAC system model. Hereby, the procedure of using BIM-data for consistent parametrization of all model components is desirable and an ongoing research issue.

4 NANDRAD FMU Calculation Functionality

4.1 State-based model evaluation and time integration in NANDRAD

When the building simulation engine NANDRAD was developed at the IBK, its design was heavily influenced by the first version of the FMI for ModelExchange standard. The entire physics evaluation is encapsulated within a state-based model object, whose state changes only by modification of the time point or conservative quantites (solution variables). After spatial discretization of all partial differential equations within the building model, a large sparse system of coupled ordinary differential equations is assembled. The time integration is then performed using our own integration framework, which incorporates the SUNDIALS:CVODE solver (Hindmarsh et al., 2005). Internally, the CVODE integrator is called by the framework for each integration step at a time. It selects/predicts a suitable integration time step, performs a modified Newton iteration[9] and upon convergence or error test failure reduces integration step until an acceptable solution is found. Note, since integration step sizes are exclusively determined by the integrator engine, synchronization with communication intervals needs to be adressed.

Figure 4 illustrates the architecture of the stand-alone NANDRAD solver. The physical model implementation is encapsulated in a model object which has similar access functions as the ModelExchange specifications require. Therefore, the ModelExchange FMU interface implementation is only a thin layer around our physical model. Our integration framework calls one of the supported time integration methods in a step-wise manner. This core loop, which also signals successful steps (`stepCompleted()`) and tells the model to write interim outputs (`writeOutputs()`), is partially replaced by the Co-Simulation master.

4.2 Implementation of the doStep functionality

When NANDRAD runs as a simulation slave, the time integration is now interrupted at the end of communication interval and control is returned to the master. Since internal integration steps may not match interval end, we choose to limit the internal integration step size so that the communication interval is not exceeded. However, this may lead to situations, wherein the last integration step before end of communication interval is much shorter than previous integration steps[10].

An alternative to limiting the last integration step would be to allow the integrator to take its natural step size. In the case of CVODE, the solution at communication interval end could be easily obtained by backward interpolation. The CVODE integrator could now be re-started with that interpolated solution in the next communication interval. However, such a restart would destroy the history within the multi-step BDF method, effectively forcing CVODE to restart integration from first order with very small time steps. This approach leads to inacceptable simulation times and cannot be recommended.

4.3 Retrieving and restoring the FMU state

The aforementioned functionality is sufficient for executing NANDRAD as FMI for Co-Simulation version 1. However, as soon as the Co-Simulation master is using an iterative or error controling algorithm, the slaves must be repeatedly set back in time (see, for example (Clauß et al., 2017)). The master needs to retrieve and restore each FMU's state.

Within NANDRAD the internal state is stored in several solver components:

- State of the integrator (time point, state variables and Nordsieck history array, counters, control variables)
- State of linear equation system solver, in case of GMRES only control variables
- state of Jacobian, since with modified Newton algorithm it is only infrequently updated
- state of preconditioner (part of Jacobian matrix and in case of ILU preconditioner also the factorized representation)
- integral model states (integral outputs, state of hysteresis loops etc.)

The data structures are typically very fragmented. The serialization implementation within NANDRAD creates a continuous memory array and then copies all data members into the array, hereby advancing an insertion pointer after each copy operation. With the use of C macro definitions, the entire serialization, deserialization and size computation functionality is only coded once, thus ensuring binary compatibility and improving code maintenance (Nicolai and Paepcke, 2016).

Additionally, the ability to serialize the entire state of model and integrator into a continuous memory block enables implementation of the `fmi2Serialize()` and `fmi2Deserialize()` functions.

[9] Within each Newton iteration the large sparse equation system is solved using a Krylov-subspace method with NANDRAD-specific preconditioner.

[10] Drastic changes in time step sizes typically lead to invalidation of Jacobian matrix information, with the related overhead of re-composing and factorizing the Jacobian.

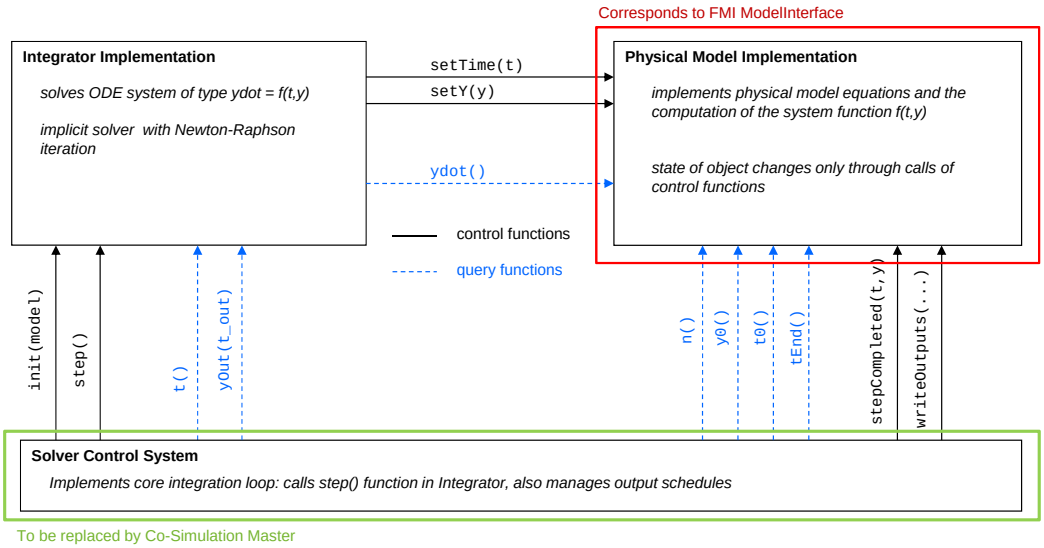

Figure 4. Core components of the NANDRAD stand-alone solver.

4.4 Integration of FMU inputs and outputs in the NANDRAD building model

The physical model of NANDRAD is internally implemented by means of interconnected state-based model objects. We allow a single model object calculation to depend on other model results. For example, room air balance is encapsulated in a single model object that requests heating and cooling load as input quantities. In turn, the power of controlled heating and cooling elements reacts on thermal response of the zone. In complete, NANDRAD owns several model objects with arbitrary interconnections that form an unstructured graph. Indeed, the states of all these models must be updated in the correct order whenever a solver state change is registered. For this purpose we cluster the model graph into nodes with cyclic and sequential connections first and order it afterwards during initialization process. As a result, all model objects appear stacked with respect to their evaluation chronology. This strategy guarantees all internal states to be current whenever an update is necessary because of changes of solver states or solver time.

This modeling concept can be easily extended to FMU inputs and outputs. In detail, we encapsulate all FMU quantities into an FMU import and an FMU export model object. The export model transfers all required output variables from the building model towards the FMI. The import model caches FMI input quantities, such as heating and cooling loads, and provides them just like calculation results to other internal model objects. This structure enables the model initialization to sort FMU inputs and outputs to the correct position inside the model object graph. For evaluation of all models depending on FMU input we store the position of the FMU import model object within the graph. In the case of update due to FMU input changes only the corresponding dependent nodes of the model graph are taken into account. This allows a model evaluation/update with only small computational effort.

To achieve good simulation performance we follow the concept of lazy evaluation: the call of `fmi2SetReal()` does not enforce an update of dependent building model objects but temporarily fills a data container. Only at the beginning of each communication step the container values are copied and the model evaluation is triggered. So, during each communication interval the model results as well as FMU outputs are consistent to the FMU inputs.

5 Application Cases

The procedure of creating and parametrizing building and equipment models and exporting FMUs has been tested with three application cases of different scales: an office room (1 conditioned zone, 383 ODEs in the building simulation part), a family row-house (2 heated zones, 502 ODEs in the building FMU) and a large appartment complex (178 conditioned zones, 23220 ODEs in the building FMU).

In this article we will only look at the first case and discuss the observed behavior with respect to the different Co-Simulation master algorithms employed. Note, it is generally possible to reduce the number of ODEs, which mostly result from spatial discretization of envelope/interior constructions, by adjusting the grid-generation parameters. As with most spatial discretization techniques, such a variation should be complemented by sensitivity studies which are beyond the scope of the article. In the test case we selected medium-fine discretization settings, leading to the reported number of elements.

5.1 Office Room Model Setup

In this model, the office is represented by four enclosing wall/floor constructions, where internal walls with same behavior are lumped into one. The only external wall

faces west and contains a large window. The old construction is made from massive lime sandstone (simplified as single layer construction) with poor thermal insulation properties. The HVAC system operates with ideal control for a heating and cooling demand calculation, dynamic daily schedules and low setback temperature on weekends. During the week, schedules distinguish between daytime and nighttime use (occupied/not-occupied). Corresponding thermal loads are computed by the plant model and imposed onto the room energy balance. Infiltration is considered with standard settings. Since the heating system is modeled in an idealistic way. The interaction between room response and heating system is very strong, leading to stiffly coupled system.

5.2 Reference Solution and Verification Procedure

Initially, for this problem a Modelica-only solution existed, yet with simplified building representation. The results of this calculation can be used for plausibility tests (Figure 5).

A correct reference solution with detailed building physics can be obtained using the ModelExchange-functionality of the building FMU and the Modelica-based equipment model. Generation of a correct reference solution depends on the following assumptions:

- building FMU correctly implements the ModelExchange interface and internal room physics,
- ModelExchange master correctly implements time integration with error checking,
- Modelica model is correctly solved within the environment.

As Modelica simulation environment and ModelExchange master we use the SimulationX[11] software, which has a comprehensive quality testing procedure to ensure correctness of Modelica and FMI master implementation. Our own NANDRAD implementation is tested against standard and customized dynamic test scenarios, and also compared to the thermal room model THERAKLES[12].

Given these testing procedures, we are confident that the results of the ModelExchange calculation will be correct and can be used to evaluate the quality of the Co-Simulation runs.

5.2.1 ModelExchange Reference Simulation

A first step in generating this reference solution was to export the NANDRAD model as FMU for ModelExchange. Since NANDRAD exports a single FMU with both Co-Simulation and ModelExchange specification, the same

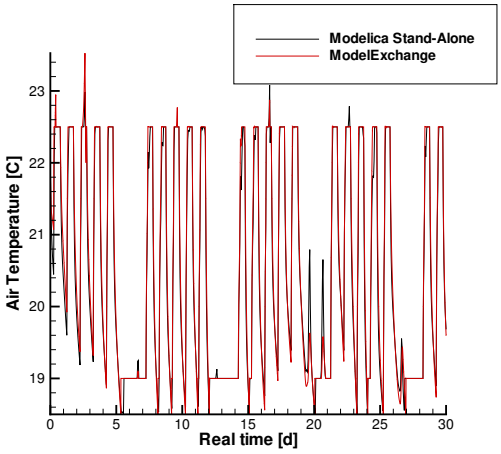

Figure 5. Comparison of reference ModelExchange solution with Modelica-only variant, using SimulationX for both simulations

FMU is used for later Co-Simulation testing. The FMU was imported into SimulationX (version 3.7), connected manually to the plant model and simulated with the SimulationX internal solver (CVODE solver, sparse Jacobian).

The fully coupled ModelExchange simulation is a fairly small problem and was simulated in acceptable time. The results were then compared to the stand-alone simplified Modelica variant and showed good agreement (Figure 5). We also did not expect much difference, since the single-layer constructions with high thermal conductance will be reasonably well approximated by the mean thermal resistance approach used by the Modelica model.

For the office room model, we use the ModelExchange reference solution for the subsequent Co-Simulation tests. However, for larger buildings (more than one hundred zones) the procedure of using ModelExchange for simulation fails, because already the symbolic analysis takes excessive time. For example, in the case of the appartment complex the symbolic analysis was not yet finished after three days.

5.3 Co-Simulation Variants

For the Co-Simulation approach, we exported the Modelica plant model from SimulationX into an FMU for Co-Simulation, version 2, hereby using the CVODE integrator option and numerical Jacobian generation method. Then, we ran the coupled simulation between the plant and building simulation FMUs with MASTERSIM[13]. We developed this open-source Co-Simulation master implementation specifically for testing and evaluation of building simulation applications.

5.3.1 Non-Iterating Gauss-Jacobi with Fixed Step-Size (only FMI v1)

The most trivial approach to Co-Simulation is the use of the Gauss-Jacobi algorithm without iteration and fixed

[11] https://www.simulationx.de

[12] See http://bauklimatik-dresden.de/therakles. THERAKLES was used in the test case as plugin alternative to NANDRAD and gave the same results for this single-zone model problem.

[13] http://mastersim.sourceforge.net

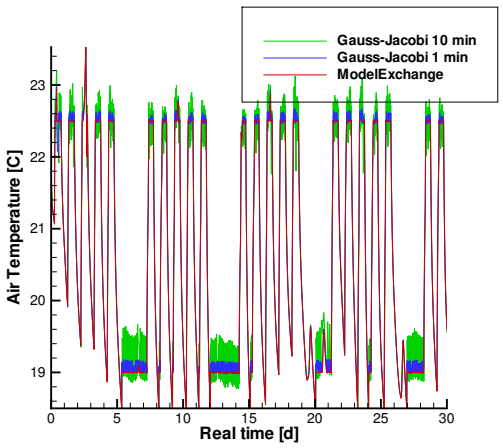

Figure 6. Gauss-Jacobi, non-iterating, fixed communication step sizes (ModelExchange results from SimulationX, Co-Simulation results calculated with MASTERSIM)

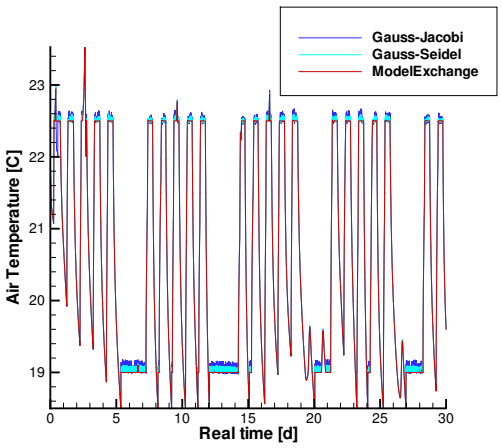

Figure 7. Comparison of non-iterating Gauss-Jacobi and Gauss-Seidel calculation for a fixed communication step of 1 minute (Co-Simulation cases done with MASTERSIM)

time step. This algorithm does not require any FMI v2 features and is thus the most compatible.

For the office room case the simulation was done with a fixed communication step size of 10 and 1 minutes. For both variants, stability problems appear. Figure 6 shows computed room mean air temperatures for the first weeks of the annual simulation.

The two Co-Simulation variants are plotted vs. the reference solution and clearly show unphysical oscillations, even at times when heating setpoints are constant. Source of the problem is the delayed reaction of the plant FMU on changes in room air temperature. Whenever the room temperature crosses the setpoint temperature during the course of the communication interval, the plant loop continues calculating based on outdated information. Specifically, when room temperature increases above setpoint temperature, the heating system still provides heat to the room based on previous room air temperature information. During cooling, the heating system remains off for too long, allowing the room air temperature to drop below the setpoint temperature. As expected, reducing the communcation step size also reduces magnitude of observed oscillations.

Using SimulationX as Co-Simulation master with same time steppings gave nearly identical results compared to MASTERSIM. Thus, we have confidence in correct behavior of the building and HVAC system FMUs.

5.3.2 Non-Iterating Gauss-Seidel with Fixed StepSize (only FMI v1)

An attempt at improving the solution was made by using the Gauss-Seidel algorithm, again with fixed step size and no iteration. Hereby, the building FMU is been given updated plant FMU results when integrated in the same step. Figure 7 shows a comparison between a Gauss-Seidel and Gauss-Jacobi simulation using the same communication step size.

Despite the notable improvement, even Gauss-Seidel does not provide sufficiently accurate results. Lowering the time step will of course improve results, but at the cost of reduced simulation performance. In practical applications the user would have to guess the communication interval and refine it in the case of stability/accuracy problems. Recognizing incorrect results may not always be easy, especially since for realistic application scenarios a reference solution does not exist. Therefore, it would be desirable to automatically adjust the time step such that results are within acceptable tolerances.

5.3.3 Adaptive Communication Step Size

We implemented the step-doubling technique in MASTERSIM as adaptive communication step method (Clauß et al., 2017). Clauß discusses such an approach within in context of FMI Co-Simulation. The error tests uses the weighted root mean square norm of all communicated real variables. When this algorithm is used, all FMUs must have the capability `canGetAndSetFMUstate` and formally implement version 2 of the FMI standard. The algorithm begins with storing the current FMU states, followed by a full communication step calculation. The results are cached, FMUs are set back and two subsequent communication steps of half length are computed. The result of the single step and the double-step calculation are used as a measure for the local truncation error. With this approach, each step, even if successful, requires 3 FMU evaluations compared to one FMU evaluation without error test.

Three simulations cases were run: Gauss-Jacobi and Gauss-Seidel methods, each with only one evaluation (no iteration), and the last with Gauss-Seidel allowing three iterations. All tests were done with a relative tolerance and an absolute tolerance of 10^{-5}. The latter may be important since thermal loads, the output variables of the plant FMU, can go down to zero.

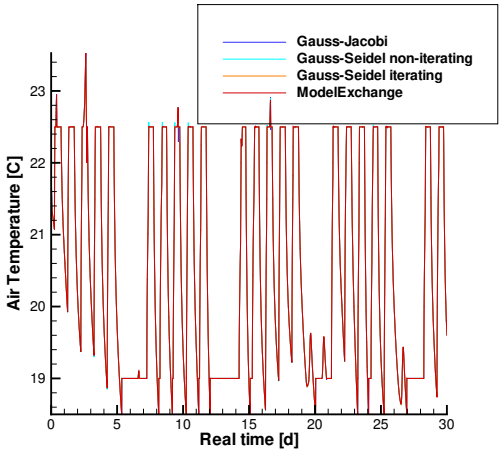

Figure 8. Comparison of non-iterating Gauss-Jacobi and Gauss-Seidel calculation variants with adaptive communcation step sizes (Co-Simulation cases done with MASTERSIM)

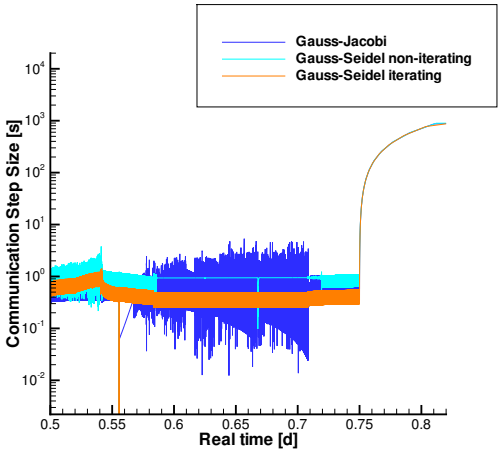

Figure 9. Illustration of step-size variation with adaptive time step methods within the first day of simulation.

When time step sizes fall below 1 s, iteration is disabled. This is a fallback criterion in MASTERSIM in order to avoid useless iterations in case of encountered discontinuities. Changing this value may also change performance of the simulation, but not impact accuracy of results.

Figure 8 shows the results obtained with variable step sizes for non-iterating cases. The results are now within the requested tolerance limit and are, with very few exceptions, nearly identical to the ModelExchange variant. The simulation time, however, has increased substantially compared to the incorrect fixed step variants.

Table 1 shows the statistics obtained from the three cases. For the first two cases iteration is not used, hence no convergence failures were recorded. Error test failures occurred about three times more frequent for the Gauss-Jacobi variant, which resulted in a drastic reduction of average time step sizes and similar increase of simulation time. The step sizes were sometimes reduced drastically to 10^{-7}s. Figure 9 illustrates the strong variations in time step. For the Gauss-Jacobi simulation, the time step varies permanently over several orders of magnitude. For all variants, when the heating system has been turned off at 0.75 d (6:00 pm), the time steps increase again up to the allowed maximum of 15 minutes. This is important for increasing overall simulation performance.

Interesting is the comparison between iterating and non-iterating Gauss-Seidel. Apparently, even with three iterations often a situation is encountered, that Gauss-Seidel cannot resolve. In these cases time step sizes were reduced due to convergence errors, which in turn reduced the number of error test failures. With this stability-dominated simulation case, use of the iterating Gauss-Seidel approach is not meaningful.

6 Summary and Conclusion

We presented the tasks necessary to successfully run a coupled building energy performance simulation using the FMI standard. We discussed the physical interface between plant and building FMU, the process of generating the building FMU itself and its internal interface implementation. In realistic cases buildings may have a large number of conditioned zones, resulting in many input and output variables. Therefore, we presented an approach for improving usability by automatically generating Modelica helper components. Further, we showed one example application for a single zone model and tested different Co-Simulation algorithms for accuracy and simulation performance.

In the test case we used an ideal heating system. The strong coupling between building and plant FMU caused stability problems for fixed-step solvers. These could be controlled by use of an adaptive communcation time step, based on local error estimates. The non-iterating Gauss-Jacobi method performed poorly compared to the non-iterating Gauss-Seidel method. Iteration, tested with the case of Gauss-Seidel, did not improve simulation performance. Without iteration, stability problem were detected by the error test, with iteration these stability problems often caused conversions failures. In either case communcation step sizes were reduced. However, in all variants the error test and communcation time step adjustment method yielded results of acceptable quality.

In our test case, the iterative Gauss-Seidel method failed frequently due to stability problems. Therefore, using Gauss-Seidel or Gauss-Jacobi iteration is not meaningful for such strongly coupled cases.

The observed behavior and conclusions drawn from the simulations are of course only an indication of general behavior. In particular, the ideal plant model and control method in conjunction with a strong thermal response of the building are definitely an extreme case. Still, success-

Table 1. Simulation statistics obtained with adaptive step simulations

Method	*Comm. Steps*	*Error Test Fails*	*Convergence Fails*	*Simulation Time*
ModelExchange	—	—	—	54 min
Gauss Jacobi non-iterating	1984803	590539	—	25 min
Gauss Seidel non-iterating	827616	146637	—	10 min
Gauss Seidel iterating	1595677	4003	809681	28 min

ful simulation was possible by use of time step adjustment, and the method and approach itself is suitable for general application.

To achieve this, the following requirements on FMU and master simulator must be fulfilled:

- the solvers within the building and plant FMU must implement an error test procedure to give consistent results,
- the FMUs must implement FMI standard version 2 with capability to set and get their states, and
- the Co-Simulation master must support communication time step adjustment based on local error estimates.

It has to be noted, though, that our conclusions are specific to the idealistic HVAC system used and observations may be different when dealing with detailed HVAC system models for modern integrated buildings.

For practical applications, overall simulation performance remains a crucial criterion. Considering the still long simulation times when applying Co-Simulation, further work is required with regard to finding suitable physical interfaces, choice of master algorithms and algorithmic parameters.

Acknowledgements

We gratefully acknowledge the support and funding received from the German Federal Ministery for Economic Affairs and Energy in the research project "EnTool:CoSim" #03ET1215A F-002792.

References

FMI 2.0 Standard, 2014. Functional Mock-up Interface for Model Exchange and Co-Simulation.

Christoph Clauß, Kristin Majetta, and Richard Meyer. Application of Richardson Extrapolation to the Co-Simulation of FMUs from Building Simulation. In *Proceedings of the 12th international Modelica Conference*, 2017.

William.S. Dols, Wang Liangzhu, Steven J. Emmerich, and Brian J. Polidoro. Development and Application of an Updated Whole-Building Coupled Thermal, Airflow, and Contaminant Transport Simulation Program (TRNSYS/CONTAM). *Journal of Building Performance Simulation*, 8(5): 326–337, 2014.

Alan C. Hindmarsh, Peter N. Brown, Keith E. Grant, Steven L. Lee, Radu Serban, Dan E. Shumaker, and Carol S. Woodward. SUNDIALS: Suite of nonlinear and differential/algebraic equation solvers. *ACM Transactions on Mathematical Software*, 31(3):363–396, 2005.

Sanford A. Klein, William A. Beckman, and John A. Duffie. TRNSYS - A Transient Simulation Program. *ASHRAE Transactions*, 82(1):623–633, 1976.

Andreas Nicolai. Physikalische Grundlagen des thermischen Raummodells THERAKLES, 2013. URL `http://nbn-resolving.de/urn:nbn:de:bsz:14-qucosa-102112`.

Andreas Nicolai and Anne Paepcke. Die Gebäudesimulationsplattform NANDRAD - Physikalisches Modell, Umsetzungskonzept und Technologien im Überblick. In *Proceedings of the BauSIM 2012*, 2012.

Andreas Nicolai and Anne Paepcke. Transformation der Gebäudeenergiesimulation NANDRAD mit variablem Zeitschrittlöser in eine FMU für Co-Simulation. In *Proceedings of the BauSIM 2016*, 2016.

Christoph Nytsch-Geusen, Jörg Huber, Manuel Ljubijankic, and Jörg Rädler. Modelica BuildingSystems – eine Modellbibliothek zur Simulation komplexer energietechnischer Gebäudesysteme. *Bauphysik*, 35(1):21–29, 2013.

Anne Paepcke and Andreas Nicolai. Anlagenregelung in ODE-Systemen am Beispiel der thermischen Raum- und Gebäudesimulation. In *Proceedings of the BauSIM 2014*, 2014.

Anne Paepcke, Torsten Schwan, and Andreas Nicolai. Schnittstellen für die Co-Simulationskopplung zwischen Gebäude- und Heizungsanlagensimulation. In *Proceedings of the BauSIM 2016*, 2016.

Per Sahlin, Lars Eriksson, Pavel Grozman, Hans Johnsson, Alexander Shapovalov, and Mika Vuolle. Whole-building simulation with symbolic DAE equations and general purpose solvers. *Building and Environment*, 39:949–958, 2004.

Michael Wetter. A Modelica-based model library for building energy and control systems. In *Proceedings of the 11th IBPSA Conference*, 2009.

Michael Wetter, Christoph van Treeck, and Jan Hensen. IEA EBC Annex 60: New generation computational tools for building and community energy systems. Technical report, Lawrence Berkeley National Laboratory, 2013.