Automated modelling of building energy systems with mode-based control algorithms in Modelica

To cite this article: Xiaoye Cai et al 2021 J. Phys.: Conf. Ser. 2042 012067

View the article online for updates and enhancements.

You may also like

- Using Modelica to investigate the dynamic behaviour of the German national standard for high pressure natural gas flow metering
  M von der Heyde, G Schmitz and B Mickan

- Distributed model predictive control of building energy systems coupled to geothermal fields
  Marc Baranski, Phillip Stoffel, Alexander Kümpel et al.

- Modelica-based TCP simulation
  T R Velieva, E G Eferina, A Y Korolkova et al.
Automated modelling of building energy systems with mode-based control algorithms in Modelica

Xiaoye Cai, Junyi Xue, Alexander Kümpel and Dirk Müller
RWTH-Aachen University, E.ON Energy Research Center, Institute for Energy Efficient Buildings and Indoor Climate, Mathieustrasse 10, 52074 Aachen, Germany
E-mail: xiaoye.cai@eonerc.rwth-aachen.de

Abstract. The increasing use of renewable energy in building energy systems has brought considerable challenges for the traditional planning process to develop appropriate control strategies. In previous work, we have introduced the MODI method to support the structured development of mode-based control algorithms, in which operating modes are core elements. However, modeling of energy systems and control algorithms for control tests is time-consuming and error-prone. Identification of permissible operating modes is also unfeasible. The paper introduces a methodology to identify permissible operating modes and model energy systems with mode-based control algorithms in the modeling language Modelica automatically. In the case study, we apply the methodology for an energy supply network and verify the functionality of the methodology. In future work, automated optimization of control algorithms will be integrated into the methodology.

1. Introduction
In recent years, building consumption has made up approximately 20% of the total energy consumption worldwide [1], a figure that will probably continue to increase. The tendency of energy consumption raises the need for increasing utilization of renewable energy in building energy systems (BES) and improved system operations. However, the building energy systems are becoming more complex by integrating renewable forms of energy. It is correspondingly unfeasible for the classical planning process of building automation systems to design appropriate control strategies for such complex systems. To address this problem, we need new control algorithms in order to avoid suboptimal system behaviours and improve energy efficiency. In previous work, we have introduced the so-called MODI-method (MODI), which enables a structured development process of mode-based control algorithms for energy systems. Operating mode is a core element in mode-based control algorithms to define operating behaviours of energy systems [2]. We use signal-interpreted Petri nets (SIPN) as a formalized descriptor to model the mode-based control algorithms in a hierarchical structure. Based on the models, we can test the control algorithms in the early stages of the planning process [2]. Nevertheless, how to identify all permissible operating modes of a complex energy system correctly is a challenging issue in MODI application, especially if engineers have never used it. In order to test control algorithms, the desired energy systems and control algorithms should be modelled in a simulation phase, which is always time-consuming and a source of failures. Regarding this problem, we investigate a methodology for the automated identification of all permissible operating modes and the
automated modelling of energy systems with their control algorithms in the modelling language Modelica.

2. Method
Figure 1 illustrates the workflow of the methodology to identify permissible operating modes and generate a ready-to-use Modelica model of an energy system with a corresponding mode-based control algorithm conversed from a configuration file.

![Figure 1. Methodology overview to generate a ready-to-used Modelica model from a configuration file.](image1)

2.1. Structured description of energy systems
In order to gather all the relevant information, we use a basic configuration language to describe an energy system in a hierarchical way [3]. According to MODI, an energy system is primarily decomposed into a hierarchical structure of the four levels [4], as Figure 3 exhibits. As defined here, a network is a complete energy supply network consisting of energy supply, distribution and consumption systems. These three types of energy systems are defined as subsystems in the subsystem level. Each subsystem can be further divided into several components in the component level, such as heat pump and boiler. Within a component, actuators which support operating of a component are decomposed in the actuator level, such as pumps and valves. Correspondingly, we develop a configuration template to describe the hierarchical structure of the decomposition.

The configuration file will be mapped in a PYTHON data set holding a nesting dictionary. A nesting dictionary in PYTHON enables to put a dictionary with a unique key inside another dictionary. Based on the property, we map a hierarchical configuration file into a hierarchical nesting dictionary. The configuration of a lower level is stored in a dictionary inside a higher level. As Figure 2 shows, the information in a configuration file will be mapped into a key: value pair. A key can lead to a targeted parameter or a targeted dictionary, while a value may be a single value or a dictionary of a lower level.

![Figure 2. Mapping a configuration file to a PYTHON nesting dictionary.](image2)

2.2. Rule-based identification approach of permissible operating modes
Based on the data set, all possible operating modes of a system are calculated using the following equation, where $M_a$ is operating modes of an actuator and $n$ is the amount of actuators:

$$N_{\text{operating mode}} = \prod_{a=1}^{n} M_a$$  (1)
A rule-based approach is proposed to identify permissible operating modes in these modes automatically. The algorithm of this approach in Figure 5 consists of two lists. One contains all possible operating modes and another list named Che is a list of Boolean value True reporting the permissibility of all modes. Each mode will be checked by all rules in the rule base for permissibility. If an operating mode conflicts with any one rule, the corresponding Boolean value of this mode in Che will be converted to False. After checking, the modes with True are the identified permissible modes and added into the list of permissible modes. The function named RuleCheck(mi, rj) in the algorithm test if there is conflict between mi and rj. Table 1 shows the performance of the function using an example. The rule in the first row can be implemented in an if-statement, where D = 0/1 means if there is a demand and m = 0/1 means if the two components (boiler and heat pump) is OFF or ON, respectively. The modes which do not satisfy the rule are marked with False in the list Mode check.
Algorithm  Rule-based identification approach

1. Possible operating mode \( m_{P_{\text{base}}} = \{m_0, m_1, \ldots, m_n\}_n \)
2. Mode check \( \text{Che} = \{\text{True}, \text{True} \ldots \} \)
3. Rule base \( R_{\text{base}} = \{r_1, \ldots, r_j\}_k \)
4. Checking permissibility RuleCheck\( (m_i, r_j) \)
5. Permissible operating mode \( m_{P_{\text{perm}}} = \{ \} \)
6. # Check for permissibility of all possible operating mode
7. for i in range (n) do
8. \( m = m_{P_{\text{perm}}} \)
9. for R in \( R_{\text{base}} \) do
10. if RuleCheck\( (m, R) = \text{False} \) then
11. \( \text{Che}[i] = \text{False} \)
12. end if
13. end for
14. end for
15. # Get permissible modes from all possible modes
16. for i in range (n) do
17. if \( \text{Che}[i] = \text{True} \) then
18. \( m_{P_{\text{perm}}} = m_{P_{\text{perm}}}, \text{append}(m_{P_{\text{perm}}}[i]) \)
19. end if
20. end for

Figure 5. Algorithm of the rule-based identification approach.

Table 1. An example to check the permissibility of modes with a rule. B: boiler; H: heat pump; D: demand.

| Mode(B,H,D) | (0,0,0) | (0,0,1) | (0,1,0) | (0,1,1) | (1,0,0) | (1,0,1) | (1,1,0) | (1,1,1) |
|-------------|---------|---------|---------|---------|---------|---------|---------|---------|
| Mode check  | True    | False   | True    | True    | True    | True    | True    | True    |

Based on available permissible operating modes, transitions of modes in the algorithm are implemented by integrating design requirements and expert knowledge. Designers can choose modes and organize mode transitions to achieve their mode-based control strategies. We also provide some design tips according to the constitution of the desired system. The proposed control strategy will be described in a configuration template in form of SIPN, which can be also mapped in the data set (see Figure 4 (a) and (b)).

2.3. Modelica code generation

The Modelica code generation in the methodology allows generating ready-to-use Modelica codes of a complex energy system containing several networks. The object-oriented character of the modelling language Modelica provides a basis to generate simulation models. Based on the Modelica libraries AixLib [5] and buildings [6], we have developed a library of the common energy components as objects, such as heat pump and boiler, in which a pump and a valve are
proposed to support the component operation. These objects are instantiated in the component level of MODI. The generation process begins also in the component level, where we only need to instantiate the objects and parameterize the instances by importing the PYTHON data set. The generated models are regarded as objects and further aggregated in the subsystem level. Based on the aggregation, the whole system model is constructed stepwise from the component level to the network level [2]. Figure 4 (c) illustrates a standard structure of a control model. All operating modes and transitions in the configuration are mapped to place and transition instances in Modelica, respectively. Transition conditions are implemented as parameters in the corresponding transition instances, while relevant state variables in the conditions are mapped to the inputs. Outputs of places are processed through a switch, which transfer the outputs of the places in the level to the corresponding modes in the next level.

We use the Mako template engine to convert the nesting dictionaries of the data set into the Modelica syntax [7, 8, 9, 10]. The template engine enables replaceable placeholders for corresponding values embedded in Modelica languages. Hence, fixed Modelica syntax in an object is compiled and flexible instances are parameterized according to the data set within a template. If we want to expend the functionality of the program, there is no need to change the program but enrich the libraries of the energy components and the corresponding templates. The code generation for the control model is similar but based on the library PNLib [11].

3. Case study
In our case study, we use the methodology to model an energy system dealing with a heat supply network, whose structure is shown in Table 2. Based on the rule-based identification approach, the permissible operating modes in all levels are available. We define the transition conditions in each level to complete a mode-based control strategy for the desired system. Table 2 presents this process. In the network level, if there is a demand, is the considered transition condition. With a demand, the network will work to satisfy the demand, which will drive subsystems in the subsystem level. Due to a heat pump as renewable energy, the subsystem 1 (Sub1) is preferred to supplying energy. If Sub1 is overload, the subsystem 2 (Sub2) will take a part of load. We use the load percentage of Sub 1 as indicator. With the mode On (1) of a subsystem, there are different modes of components in the component level. Within Sub1, the heat pump has the highest priority. The boiler runs only if maximal power of the heat pump is reached. The temperature of the heat storage in the heat pump reveals if the heat storage is empty and the heat pump runs at maximal power. In order to achieve similar operating times of the boilers, we use a time-based strategy to compensate the operating time difference. If only one boiler is needed, the boiler with low operating time is preferred to running. Maintenance cost will benefit from a compensated operating time difference between the two boilers.

As described in Section 2.3, the corresponding simulation model is also generated. Figure 6 depicts the simulation results for one day. The simulation results in the subsystem level illustrated in the left column present that Sub1 runs preferentially, while Sub2 will help supplying heat if the load percentage of Sub1 > 0.85 and stop if the load percentage of Sub1 < 0.5. Correspondingly, the model of Sub1 and Sub2 can provide heat in accordance with their operating modes. Furthermore, the boilers in Sub2 supply water with a higher flow temperature, hence Sub2 shows more visible oscillations than Sub1 with a low demand. In the right column, the heat storage in the heat pump is initially empty so that the heat pump and the boiler supply heat for demands and charging the storage. If the storage is not empty ($T_{storage} > 327.15K$), the boiler will stop. The heat supply of the heat pump and the boiler reveals the control model indeed organizes the operation of these two components correspondingly.
Table 2. Result of mode identification and control design in the case study. Sub: subsystem; HP: heat pump; B: boiler; mode 0: Off and 1: On.

| Energy system | Permissible operating modes | Transition condition |
|---------------|-----------------------------|----------------------|
| Network level: Heat supply network (Network 1) | (0), (1) | If there is a demand? |
| Subsystem level: Two subsystems (Sub1, Sub2, Demand) | (0,0,0), (1,0,1), (0,1,1), (1,1,1) | Priority: Sub1 > Sub2 If Sub1 is overload? |
| Component level Sub1 (HP, B) | (0,0),(1,0),(0,1),(1,1) | Sub1: If HP is overload? |
| Component level Sub2 (B1, B2) | (0,0),(1,0),(0,1),(1,1) | Sub2: Time-based control |

4. Conclusion
The results of the paper demonstrate the feasibility of the methodology to generate ready-to-run Modelica simulation models and benefits of the semi-automatic development of mode-based control algorithms. In the future, a complete tool-chain in accordance with the methodology is expected for further simplifying the application of MODI. The rule base for identification of permissible operating modes and the library of energy components will be extended aiming at the enrichment of the tool functionality. Furthermore, optimization algorithms can be integrated into the tool for optimizing mode-based control algorithms.

Acknowledgments
We gratefully acknowledge the financial support provided by the Federal Ministry for Economic Affairs and Energy (BMWi), promotional reference 03ET1485A.

References
[1] International Energy Outlook 2019 with projections to 2050 https://www.eia.gov/outlooks/ieo/pdf/ieo2019.pdf
[2] Cai X, Schild T, Schreiber T, Baranski M and Müller D 2019 Modeling of petri-net-based control algorithms for the simulation-based improvement of the planning process of building energy systems Journal of Physics: Conference Series vol 1343 (IOP Publishing) p 012123
[3] Configuration file parser https://docs.python.org/3/library/configparser.html
[4] Cai X, Schild T, Schreiber T, Kümpel A and Müller D 2021 Using simulation to evaluate mode-based control algorithms for energy systems
[5] Müller D, Lauster M, Constantin A, Fuchs M and Remmen P 2016 Aixlib an open-source modelica library within the iea-ebc annex 60 framework
[6] Wetter M, Zuo W, Nonidui T S and Pang X 2014 Modelica buildings library vol 7 (Taylor & Francis) pp 253–270
[7] Mako Templates for Python http://www.makotemplates.org accessed: 2016-03-07
[8] Thorade M, Rädler J, Remmen P, Maile T, Wimmer R, Cao J, Lauster M, Nytsch-Geusen C, Müller D and van Treeck C 2015 An open toolchain for generating modelica code from building information models Proceedings of the 11th International Modelica Conference, Versailles, France, September 21-23, 2015 118 (Linköping University Electronic Press) pp 383–391
[9] Remmen P, Lauster M, Mans M, Fuchs M, Osterhage T and Müller D 2018 Teaser: an open tool for urban energy modelling of building stocks vol 11 (Taylor & Francis) pp 84–98
[10] Remmen P, Lauster M, Mans M, Osterhage T and Müller D 2016 Citygml import and export for dynamic building performance simulation in modelica Proceedings of the Building Simulation and Optimization Conference (BSO16), Newcastle University, Newcastle upon Tyne, UK pp 12-14
[11] Proß S and Bachmann B 2012 Pnlib - an advanced petri net library for hybrid process modeling (Munich)