Diese Arbeit wurde vorgelegt am
Lehr- und Forschungsgebiet Theorie der hybriden Systeme

Automatisierte Dimensionierung und Effizienzsteigerung von Heizungssytemen

Automatic Dimensioning and Efficiency Optimization of Heating Systems

Masterarbeit
Informatik

März 2026

Vorgelegt von Presented by	Matteo Ludwig Matrikelnummer: 402440 matteo.ludwig@rwth-aachen.de
Erstprüfer First examiner	Prof. Dr. rer. nat. Erika Ábrahám Lehr- und Forschungsgebiet: Theorie der hybriden Systeme RWTH Aachen University
Zweitprüfer Second examiner	Prof. Dr. rer. nat. Jürgen Giesl Lehr- und Forschungsgebiet: Programmiersprachen und Verifikation RWTH Aachen University
Betreuer Supervisor	Dr. rer. nat. Pascal Richter Lehr- und Forschungsgebiet: Theorie der hybriden Systeme RWTH Aachen University

Contents

Acronyms

API application programming interface

ARC automatic reference counting

ATD air transfer device

BAFA Federal Office for Economic Affairs and Export Control

BFS breadth-first search

DFS depth-first search

DIN German Institute for Standardization

FPS frames per second

GEG Gebäudeenergiegesetz

KfW Kreditanstalt für Wiederaufbau

LP linear program

MILP mixed integer linear program

PDF portable document format

PICV pressure independent control valve

SOS special order sets

TRV thermostatic radiator valve

UUID universally unique identifier

VDI Association of German Engineers

ZVSHK Zentralverband Sanitär Heizung Klima

1. Introduction

Residential heating systems are among the largest consumers of energy in Germany, accounting for a substantial share of both total energy expenditure and national gas consumption [2]. In the context of rising energy prices [9, 10, 37] and an increasingly stringent regulatory framework [11], the improvement of energy efficiency in existing and newly installed heating systems has become a central requirement.

Established measures for improving efficiency include hydronic balancing and the optimization of supply water temperatures based on a heat load calculation, although these measures are not yet consistently implemented in practice [33]. These measures enable heating systems to be dimensioned and adjusted in accordance with the actual heating demand, thereby improving overall energy efficiency [18, 26, 29, 41].

Software tools that implement these procedures exist and are widely used by heating engineers and energy consultants [14, 28]. However, they are designed as standalone desktop or web-based applications and do not expose their functionality programmatically, making integration into third-party systems or automated workflows impractical. This work presents the design, implementation, and evaluation of an interface-independent library, written in Swift, that performs heat load calculations in accordance with DIN/TS 12831-1:2020-04, static and dynamic hydronic balancing for two-pipe central heating systems, and supply temperature optimization with cost-minimizing replacement recommendations for insufficiently dimensioned system components.

1.1. Motivation

The energy efficiency of residential heating systems has become a matter of increasing regulatory and economic relevance in Germany. Several paragraphs of the Gebäudeenergiegesetz (GEG) [11], a German law regulating mandatory energy-saving measures in buildings, directly mandate optimization procedures to be performed on residential heating systems. For newly installed systems, these include room-level heat load calculations, which form the basis for supply temperature optimization and hydronic balancing. As the GEG further stipulates that combustion-based heat generators must be replaced no later than thirty years after their commissioning [11], the demand for such optimization measures can be expected to increase in the coming years. Furthermore, depending on the type of heat generator installed and its year of commissioning, these measures may also be mandatory for existing systems regardless of any replacement [11].

Beyond regulatory compliance, hydronic balancing and supply temperature optimization offer empirically proven practical benefits. Depending on the heating system, efficiency improvements of more than 10% have been reported, with older and less efficient systems tending to benefit most [12, 29, 30]. For heat pump systems in particular, even modest reductions in supply water temperature can yield substantial improvements in efficiency, making supply temperature optimization economically significant [1]. Hydronic balancing can further reduce flow noise, improve the responsiveness and controllability of thermostatic radiator valves (TRVs), and help to ensure that each room receives the amount of thermal energy required to maintain its target temperature [1, 26, 41].

The economic relevance of heating system efficiency has been further amplified by the development of the energy markets in recent years. Political instability affecting the reliability and pricing of gas supply has led to substantial increases in gas prices in Germany [9, 10, 37]. This is particularly significant given that approximately 50% of residential buildings in Germany relied on gas-based heating systems as of 2023 [6]. According to an assessment by the Arbeitsgemeinschaft Energiebilanzen e. V., room heating accounted for approximately 78.3% of gas expenditure of private households in 2019 and 2020. Private households consumed 724.6 PJ of gas for room heating in 2020, representing 33.2% of total gas energy consumption in Germany [2]. Against this background, improving the efficiency of existing gas-based residential heating systems through measures such as hydronic balancing and supply temperature optimization represents not only a regulatory obligation but also a significant lever for reducing both energy costs and national gas consumption. Supply temperature optimization is further relevant in the context of replacing existing gas furnaces with heat pumps, as it is often a prerequisite for efficient heat pump operation.

Additionally, common modernization measures such as increased thermal insulation can introduce asymmetries in existing heating systems. Top and bottom floor rooms typically benefit more from additional insulation than middle floor rooms, potentially rendering their heat emitters substantially overdimensioned relative to the rest of the building. Correcting such imbalances may require replacement of system components in addition to hydronic balancing [1].

Financial incentives further reinforce the relevance of these measures. To qualify for a subsidised development loan granted by the Kreditanstalt für Wiederaufbau (KfW) for new construction or modernization of existing residential buildings, a so-called "Effizienzhausnachweis" is required, for which the execution of hydronic balancing is mandatory. The German Federal Office for Economic Affairs and Export Control (BAFA) additionally subsidises the optimization of existing heating systems through hydronic balancing, the replacement of fossil heat generators with heat pumps, and the installation of heat emitters enabling operation at reduced supply water temperatures [8].

1.2. Related Work

The calculation procedures relevant to room heat loads and hydronic balancing are defined by a set of standards and technical guidelines. As mandated by the GEG [11], heat load calculations are governed by DIN/TS 12831-1:2020-04 [23], a pre-standard published by the German Institute for Standardization (DIN). It represents a national addition to DIN EN 12831-1:2017-09 [21], which is the German adoption of the European standard EN 12831. Together, these standards define the normative basis for room-level and building-level heat load calculations in Germany. Hydronic balancing is addressed by VDI 2073 Part 2 [41], which describes the principles and methods of pipe network calculation and balancing for water-based heating systems, and by the draft standard E DIN 94679-1:2022-10 [26], which provides more prescriptive guidance on the balancing procedure. The standards DIN EN 442-2:2015-03 [20] and DIN 4703-3:2000-10 [17] further define methods for calculating the expected heat output of a radiator as a function of its

operating conditions, and are therefore relevant to the assessment and optimization of supply water temperatures.

Several established software tools exist that implement the aforementioned procedures and standards. ZVPlan [14], developed by ConSoft GmbH, is a desktop application for heating system planning that offers partial support for heat load calculations in accordance with DIN/TS 12831-1, pipe network analysis, and the determination of valve settings for hydronic balancing. EVEBI [28], developed by ENVISYS GmbH & Co. KG, is a certified energy consulting software that supports heat load calculations in accordance with DIN/TS 12831-1 and produces output documents following the normative report format defined by the standard. Dendrit STUDIO [16] by Dendrit Haustechnik-Software GmbH is a planning tool integrating heat load calculation and hydronic balancing, with output suitable for KfW and BAFA subsidy documentation. SOLAR-COMPUTER [36] by SOLAR-COMPUTER GmbH is a modular suite for building services engineering supporting standard-compliant calculations including DIN EN 12831-1 heat load calculation. ZUB Helena Heizlast [46] by ZUB Systems supports heat load calculations in accordance with DIN/TS 12831-1 as a module of the energy consulting software ZUB Helena, with a free extension for hydronic balancing of existing residential buildings. The company autarc energy [4] offers a cloud-based platform for hydraulic balancing, targeted at heating system installers, which features AI-assisted heat load and valve setting determination. Aside from software tools, there are also companies like wbs [43], which offer heat load calculations and hydronic balancing as a professional service.

1.3. Contribution

This work presents the design, implementation, and evaluation of a software library that covers the measures mandated by §60c of the GEG [11] for residential heating systems, namely room-level heat load calculations in accordance with DIN/TS 12831-1:2020-04, an assessment of whether the supply water temperature can be reduced, and hydronic balancing. Beyond the scope of §60c, the library additionally supports building-level heat load calculations in accordance with DIN/TS 12831-1, and extends the supply temperature assessment with cost-optimised, technically feasible replacement recommendations for insufficiently dimensioned radiators and pipes. The related normative calculation procedures defined in DIN/TS 12831-1:2020-04, DIN EN 12831-1:2017-09, E DIN 94679-1:2022-10, and VDI 2073 Part 2 are systematically analyzed, translated into computable formulations, and implemented as a unified, interface-independent Swift library exposing its functionality through a programmatic application programming interface (API).

More specifically, the library is developed to support both static and dynamic hydronic balancing for two-pipe central heating systems with and without branch balancing valves. To solve the static hydronic balancing problem, a novel three-stage lexicographic mixed integer linear program (MILP) formulation is employed, that simultaneously handles discretely and continuously adjustable valves, branch balancing valves acting across multiple circuits, and valve authority constraints, while guaranteeing optimal valve settings with respect to the defined balancing objectives. Furthermore, the supply

temperature optimisation is realized through a MILP-supported iterative algorithm that determines the feasibility of a lower supply temperature while minimizing replacement costs for radiators and pipes subject to hydraulic and thermal feasibility constraints.

Finally, the correctness of the implementation is assessed against six independent reference sources comprising established standards, certified software, and professional service providers, with all deviations systematically identified and attributed, including errors discovered in multiple reference documents. The results provide strong evidence that the developed library correctly implements the required calculation procedures and constitutes a reliable foundation for programmatic heating system analysis and optimization.

1.4. Outline

The remainder of this work is structured as follows. Section 2 introduces the technical background necessary to understand the implemented procedures, covering heat load calculations and hydronic balancing. Section 3 derives the functional and non-functional requirements for the library and presents the design of its two modules, including the data structures, calculation procedures, and optimization formulations employed by each. Section 4 describes the implementation of both modules in Swift, discussing the language conventions adopted and the implementation-level decisions made with respect to the design. Section 5 evaluates the correctness of the implementation by comparing the results of both modules against reference calculations drawn from the applicable standards and established software tools. Section 6 summarises the findings of the work and identifies directions for future development.

2. Background

Both heat load calculations and hydronic balancing, which will be introduced in this section, are established procedures for the design and optimization of heating systems. Heat load calculations follow normative procedures, that in Germany are defined by the pre-standard DIN/TS 12831-1:2020-04 [23], which in turn is the German national addition to the base standard DIN EN 12831-1:2017-09 [21]. The DIN standard for hydronic balancing, E DIN 94679-1:2022-10 [26], is still under development and currently available only in draft form. Nevertheless, hydronic balancing is a well-established engineering practice and is, in addition to the DIN draft, described in the technical rule document VDI 2073 Part 2 [41]. As room-level heat load calculations are a prerequisite to performing hydronic balancing, the following first introduces the principles of heat load calculations and subsequently addresses the methods and concepts of hydronic balancing, using the aforementioned standards and technical rules as references.

2.1. Heat Load Calculations

Even with modern insulation, buildings constantly lose thermal energy to the outside world, when their inside temperature exceeds the temperature of the outside. It is exactly this loss of thermal energy that must compensated by the heating system in order to maintain the inside temperature. Consequently, a large part of the heat load of a building is defined by how much heat it loses [21, 44].

The heat losses of a building increase with a growing temperature difference between the inside and outside, resulting in a higher load on the heating system. Accordingly, increasing the heating power to a building raises the inside temperature, which in turn leads to increased heat losses to the surroundings. Therefore, the inside temperature of the building will rise until it reaches a new equilibrium, where the building loses exactly as much thermal energy to the outside as it gains from the heating system. Given enough information on the building and its environment, one can calculate how much heating power is required for the building to reach that equilibrium at a desired temperature [3, 21, 44].

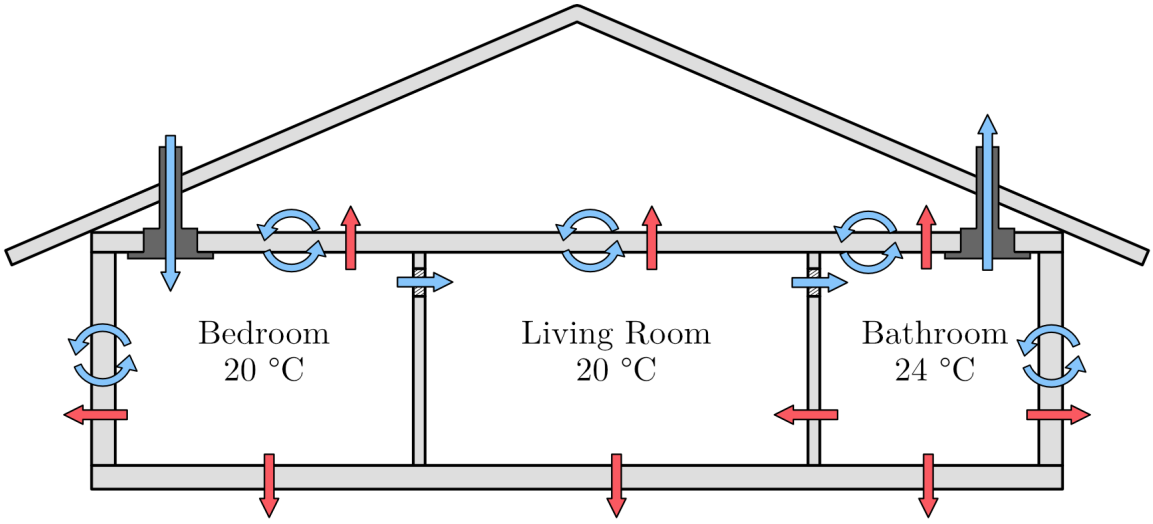

Figure 1: Schematic illustration of transmission (red) and ventilation (blue) heat losses in a single-family residential building. Infiltration through the building envelope is depicted by circular blue arrows, and a mechanical ventilation system supplying, transferring, and exhausting air through the rooms is shown by straight blue arrows and vents.

In buildings, heat is lost mainly through transmission across the building envelope and through ventilation, as depicted in Figure 1. Transmission losses occur whenever a part of the building, like a wall or the baseplate, are in contact with a colder environment such as the outside air or with the ground. The colder outside temperature cools that part of the building down, which in turn lowers the inside temperature. Buildings may also make contact other, potentially colder, adjacent buildings, to which they also lose heat via transmission [21, 44].

Ventilation losses occur when a building expels warm air to the outside world, replacing it with colder air. This energy loss of heated air needs to be compensated though heating up cold outside air in order to maintain the design inside temperature. Ventilation losses also occur in buildings without a dedicated ventilation system, since building envelopes are generally not designed to be airtight [21, 44].

These principles apply not only to the building as a whole, but also to the individual rooms within the building. When considering individual rooms, it must be taken into account that rooms not only lose energy to the outside environment, but may also transfer energy to adjacent, cooler rooms. This can occur both through transmission and ventilation. Conversely, some rooms may gain additional thermal energy from warmer, adjacent rooms [21].

The following subsections outline the basic principles of transmission and ventilation losses and examine how outside temperature, building thermal inertia, and additional heat loads affect the total heat load.

2.1.1. Transmission Losses

Transmission losses of a room are calculated by considering the components that define its enclosure and separate it from the exterior or neighboring rooms. Examples of such components include walls, floors, ceilings, windows, and doors. All of these components separate the room interior from the exterior and are therefore relevant to the calculation of transmission heat losses. If the environment on the outside face of the component is colder than the inside of the room, the component inevitably transmits energy from the inside to the outside, resulting in energy loss. Conversely, a component can make a room gain thermal energy when it borders on a warmer environment [3, 21, 44].

The amount of energy the component transmits over time can be calculated based on the components heat transfer coefficient. This coefficient, commonly referred to as the U-value, specifies the rate at which heat is transferred through the component depending on its surface area and the temperature difference between the two faces of the component. Components with better insulation exhibit lower U-values and therefore transmit heat at a lower rate. Equation 1 forms the fundamental basis for heat transmission [3]:

$$\Phi_T = A \cdot U \cdot \Delta\theta \tag{1}$$

where:

Φ_T	Heat transmission	W
A	Component area	m^2
U	Heat transfer coefficient	$W/(m^2 \cdot K)$
$\Delta\theta$	Temperature difference across the component	K

The equations used to calculate heat transfer in DIN/TS 12831-1 and DIN EN 12831-1 add some additional factors to Equation 1 in order to account for different environment factors. An example for such a factor defined by DIN EN 12831-1 is determined by the

type of environment which the component borders on, which can either be the outside air, the ground, or another heated or unheated room. On this basis, the heat transfer may be assumed to be higher or lower than the standard formulation would suggest [21, 23].

Having defined how to calculate the heat transfer losses caused by a single component, the transmission loss of a room is defined as sum of the losses of all components, including negative losses representing heat gains. The transmission losses of the entire building is the sum of all losses of components bordering the outside world. When considering the entire building, heat transmissions between rooms are neglected, as they do not affect the overall heat loss of the building [21, 23].

2.1.2. Ventilation Losses

When calculating the ventilation losses of a room, multiple pathways by which air at a different temperature enters the room, must be considered. DIN/TS 12831-1 and DIN EN 12831-1 define several equations for calculating the thermal energy losses caused by different forms of air exchange, all of which can be represented in the general form shown by Equation 2 [21, 23]:

$$\Phi_V = \rho \cdot c_p \cdot q_v \cdot \Delta\theta \tag{2}$$

where:

Φ_V	Ventilation heat loss	W
ρ	Density of the air	$\mathrm{kg/m^3}$
c_p	Specific heat capacity of the air	$\mathrm{Wh/(kg \cdot K)}$
q_v	Volumetric air flow rate	$\mathrm{m^3/h}$
$\Delta\theta$	Temperature difference between design room temperature and air stream	K

Both ρ and c_p are considered to be constants in DIN/TS 12831-1, leaving q_v and $\Delta\theta$ as the only variables influencing the ventilation heat losses. Since air can enter a room through multiple pathways and at different temperatures, each room is associated with several volumetric airflow rates q_v and their corresponding $\Delta\theta$ values. The temperature difference $\Delta\theta$ is the temperature of the air stream relative to the internal temperature of the room. Examples of such air streams include infiltration through leaks in the building envelope, outdoor air supplied by a mechanical ventilation system, and air transferred from adjacent rooms with different temperatures [21, 23].

The total ventilation heat loss of a room is obtained by summing the contributions of all air streams. DIN EN 12831-1 further defines the concept of ventilation zones, each comprising one or more rooms, which allows ventilation losses to be determined on a zone-by-zone basis and subsequently aggregated to building level [21, 23].

2.1.3. Outside Temperature

Both transmission and ventilation losses depend heavily on the temperature difference between the inside and outside of the building [3, 21, 23, 44]. While the indoor temperature can in principle be freely chosen, DIN/TS 12831-1 specifies standard values by room type, for example 20 °C for living rooms and 24 °C for bathrooms [23].

In contrast, the outdoor temperature is determined by natural conditions and therefore varies constantly. Since the goal of heat load calculations is not to simulate the total annual energy loss but to determine the maximum expected heat load, a design-relevant outdoor temperature must be defined as the basis for the calculation [3, 23]. For this purpose, DIN/TS 12831-1 specifies a design outside temperature and an average outside temperature for each German postcode. The design outside temperature is defined as the coldest two-day mean value occurring at least once every two years within the period from 1995 to 2012 [23].

2.1.4. Thermal Inertia

Although DIN EN 12831-1 standard is limited to steady state heat load calculations and generally do not model dynamic behavior, fluctuations in outdoor temperature can still affect the heat load. This can, partly, be attributed to the thermal inertia of the building. In this context, thermal inertia describes how quickly the building responds to changes in outside temperature [21, 23, 44].

Both DIN EN 12831-1 and DIN/TS 12831-1 allow this influence to be optionally included in the calculation by defining a synthetic adjusted design outside temperature. This adjusted temperature is derived from the building time constant τ, which is calculated based on the overall thermal storage capacity of the building and its overall heat transfer coefficient. Buildings with a higher time constant react more slowly to outside temperature changes and therefore retain heat longer following a temperature drop. Accordingly, a higher time constant is expected to reduce the building heat load, which is reflected in the calculation by increasing the adjusted design outside temperature relative to the original design value as a function of τ [21, 23].

The application of this adjusted temperature differs between the European base standard DIN EN 12831-1 and its national addition DIN/TS 12831-1. While the European norm uses the adjusted temperature for both transmission and ventilation losses, the German addition restricts its usage to transmission losses and uses the original design outdoor temperature for ventilation losses. As this work focusses on the national addition DIN/TS 12831-1, the adjusted temperature is therefore applied only to transmission calculations in the following [21, 23].

2.1.5. Additional Loads

Another potential influence on the buildings overall heat load are additional heating-up loads caused by rooms that are only heated intermittently. Such rooms regularly experience temperature setbacks from which they need to recover, thereby causing heat loads that exceed the regular load caused by maintaining a steady temperature. The

magnitude of such an additional heating-up load depends strongly on multiple factors including the duration for which the room will not be heated, its insulation, and its heat storage capacity [21, 23].

Finally, DIN/TS 12831-1 allows for not one, but two design temperatures to be specified for each room. The first is the standard temperature, which must be selected from a predefined range of indoor temperatures corresponding to the relevant room type. For example, according to the DIN/TS 12831-1 standard temperature table, a living rooms should be heated to 20 °C, while bathrooms should have a higher temperature of 24 °C. The other temperature is the comfort temperature, which may be higher or equal to, but never lower than the standard temperature. This allows heating system designers to account for potential preferences for elevated indoor temperatures. As a result, the heat load calculation under DIN/TS 12831-1 is carried out twice for each room, once based on the standard temperature and once based on the comfort temperature. Potential increases in heat load caused by intermittent heating and elevated comfort temperatures can be taken into account in the overall heat load of both the room and the building [23].

Having discussed the fundamental principles governing building heat losses and the resulting heating power requirements of individual rooms, the next step is to address the efficient adjustment of the heating system to these loads through hydronic balancing.

2.2. Hydronic Balancing

In German residential buildings, space heating is typically provided by central systems that distribute heat to individual rooms via water-based emitters such as radiators or panel heating [1, 6]. This section first introduces the core principles and components of such a central heating system before turning to hydronic balancing, a procedure aimed at improving system efficiency through the adjustment of hydraulic parameters.

Central heating systems use a single heat generator, such as a heat pump or a gas furnace, to heat water that is circulated via a network of pipes into heat emitters in the individual rooms. The thermal energy of the water is subsequently partially transferred to the indoor air. This results in an increase in room temperature and consequently a decrease in water temperature. The cooled water is returned to the heat generator, where it is reheated and recirculated [1]. The heat output of each emitter can be regulated by the temperature and flow rate of the water [20].

The objective of hydronic balancing is to adjust the water flow rate through each heat emitter in such a way that it delivers exactly the amount of heat required to heat the corresponding room. By matching the heat output of each emitter to the calculated heat load, hydronic balancing ensures that no heat emitter supplies substantially more or less heat than necessary [26, 41]. This prevents underheating in some rooms and overheating in others. Some studies and reports claim that hydronic balancing can improve heating system efficiency by more than 10% [12, 29, 30].

For water-based central heating systems, various piping configurations exist for connecting emitters within a heating system, which can be broadly classified into single-pipe and two-pipe systems [26, 40, 41].

2.2.1. Single and Two Pipe Systems

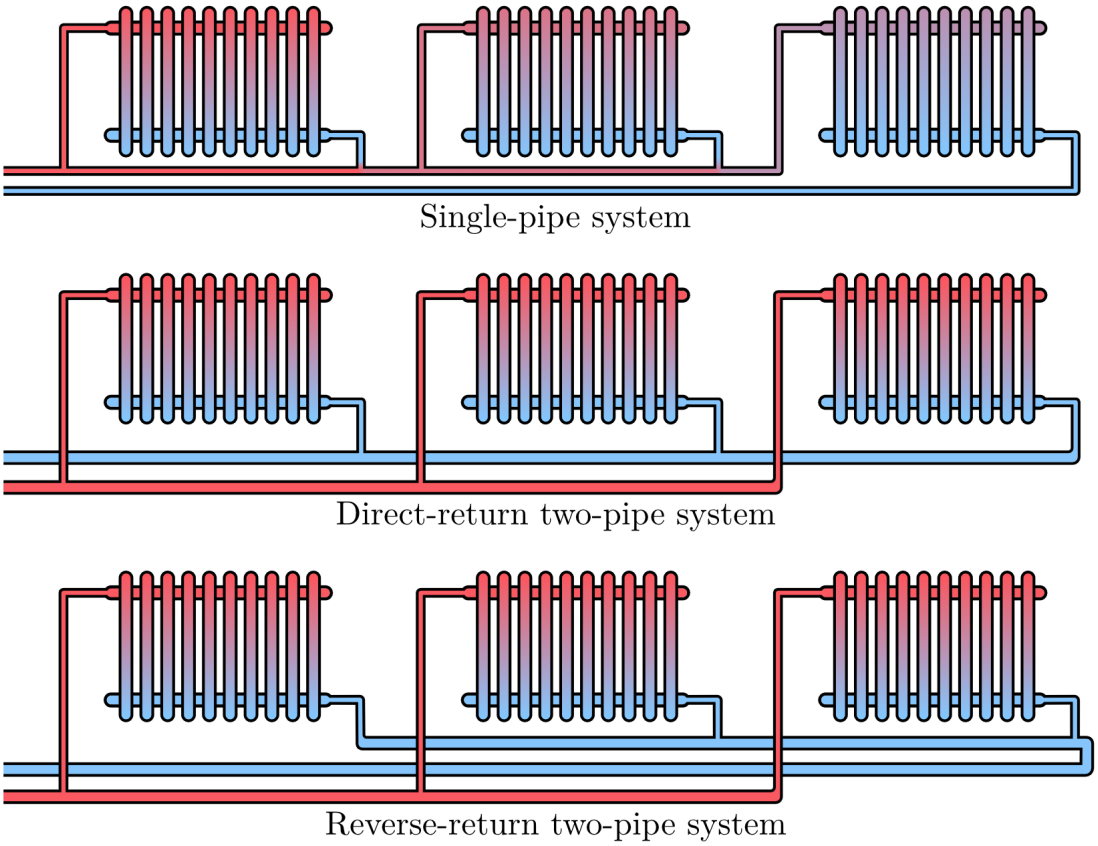

Figure 2: Schematic description of a single-pipe system (top), a direct-return two-pipe system (middle), and a reverse-return two-pipe system (bottom). In the single-pipe system, radiators are connected in series, causing a stepwise temperature drop. Both two-pipe systems connect radiators in parallel, ensuring approximately equal supply temperatures, with the reverse-return variant additionally equalising circuit lengths.

While multiple variants of single-pipe systems exist, they share the characteristic that all heat emitters are connected in series, as shown in Figure 2. From the heat generator, the heated supply water flows partially into each emitter and partially bypasses it, with the ratio determined by the valve setting of the emitter. Within the emitter, the supply water releases part of its thermal energy, reducing its temperature. The cooled return water then flows back into the pipe and mixes with the bypassed supply water, reducing the temperature of the combined flow. This process repeats sequentially through all emitters until the return water from the final emitter reaches the heat generator [26, 40].

In contrast, two-pipe systems connect all heat emitters in parallel, thus forming two networks of pipes, one for supply and one for return water. The supply network connects the outlet of the heat generator to the inlets of all emitters and the return network

connects the outlets of the emitters back to the inlet of the heat generator. By having a direct path for the heated water from the heat generator to the heat emitter through the supply pipes, and another direct path for the cooled water to return trough the return pipes, two-pipe systems effectively form one hydraulic circuit per heat emitter. However, this does not mean that each emitter is served by dedicated pipes throughout. Instead, common pipe sections are shared, and branches are formed using junctions to connect individual emitters [26, 40].

Two-pipe systems can further be classified into direct-return systems and reverse-return (Tichelmann) systems, both illustrated in Figure 2. In a direct-return system, the structure of the return pipes matches that of the supply network, leading through approximately equal lengths of supply and return pipes for each circuit. As a result, the individual hydraulic circuits differ in length, leading to varying pressure losses between emitters. In contrast, reverse-return systems are designed such that the shortest supply path corresponds to the longest return path and vice versa. By deliberately routing the return piping in this manner, the total length, and therefore pressure loss, of each circuit is made approximately equal [26, 40].

A two-pipe system has the advantage that each emitter is supplied with water of the same temperature, when not accounting for temperature losses within the pipes themselves. In single-pipe systems, by contrast, the water temperature decreases from one emitter to the next, which can result in insufficient heat output from emitter located later in the sequence. In radiator-based systems, this is typically compensated by oversizing the downstream radiators [1].

A further advantage of two-pipe systems is the ability to adjust the water flow rate independently for each emitter [26, 40]. In contrast, single-pipe systems impose a uniform flow rate on all emitters due to their single-circuit configuration, making individual hydraulic control impossible. This limitation can be partially mitigated by installing adjustable bypass valves, which allow some of the heating water to bypass an emitter, enabling limited regulation of heat output for following emitters [26, 41].

Due to their inherent advantages, two-pipe systems are the standard in newly constructed german residential buildings [1]. Hydronic balancing based on individual flow rate adjustment is not possible in single-pipe systems in the same way it is in two-pipe systems. While additional bypass valves and control elements can be used to influence emitter outputs [26, 41], these approaches are beyond the scope of this work. Accordingly, only two-pipe systems are considered in the following.

2.2.2. Radiators

Radiators are the predominant type of hydronic heat emitter in the German residential building stock [1]. As the objective is to adjust the flow rate of each heat emitter so that its actual heat output matches the requirements determined by the heat load calculations [26, 41], it is necessary to understand how the water flow rate influences radiator performance.

Each radiator is typically specified with a nominal heat output at a specified excess temperature. The excess temperature is defined as the temperature difference between

the average water temperature within the radiator and the air. In accordance with DIN EN 442-2, manufacturers are required to specify the nominal heat output Φ_N of each radiator at a nominal excess temperature ΔT_N. Additionally, an exponent n is specified, which describes how the heat output changes in response to variations in excess temperature. The relationship between excess temperature and heating power is characterized by Equation 3 [19, 20]:

$$\Phi = \Phi_N \cdot (\Delta T / \Delta T_N)^n \tag{3}$$

where:

Φ	Heat output	W
Φ_N	Nominal heat output at excess temperature ΔT_N	W
ΔT_N	Nominal excess temperature	K
ΔT	Excess temperature	K
n	Exponent of the radiator characteristic equation.	-

Since the heat output of a radiator must correspond to the heat load of the room it serves, the value of Φ in Equation 3 is determined by the previously calculated heat load. Furthermore, Φ_N, ΔT_N, and n are known parameters defined by the chosen radiator type. Consequently, ΔT is the only remaining variable governing the radiator heat output and can be calculated by rearranging Equation 3 [26].

As defined, the excess temperature ΔT is given as the difference between the average water temperature within the radiator and the room air temperature. Two definitions of this temperature exist within the applicable standards, an arithmetic definition given by DIN EN 442-2 and a logarithmic definition given by DIN 4703-3 [17, 20]. As both are used in practise [1, 14, 26], both definitions are presented in the following [17, 20]:

$$\Delta T_{ar} = \frac{\vartheta_s + \vartheta_r}{2} - \vartheta_i \tag{4}$$

$$\Delta T_{ln} = \frac{\vartheta_s - \vartheta_r}{\ln\left(\frac{\vartheta_s - \vartheta_i}{\vartheta_r - \vartheta_i}\right)} \tag{5}$$

where:

ΔT_{ar}	Arithmetic excess temperature	K
ΔT_{ln}	Logarithmic excess temperature	K
ϑ_s	Temperature of supply water	°C
ϑ_r	Temperature of return water	°C
ϑ_i	Air temperature	°C

The supply water temperature ϑ_s is dependent on the heat generator and overall system design. Modern heating systems, especially heat pumps, typically regulate the supply water temperature in response to outdoor temperature and the resulting heat load [1]. Hydronic balancing is generally performed for the design condition, assuming operation at heat loads as calculated by DIN/TS 12831-1. Accordingly, the supply water temperature is assumed to be fixed at the value specified for design conditions [26, 41].

Since the room temperature ϑ_i, supply water temperature ϑ_s, and excess temperature ΔT can all be determined from design parameters, the return water temperature ϑ_r can be determined by solving Equation (4) or (5), depending on the definition used for ΔT. Based on ϑ_s and ϑ_r, the needed water flow rate for the radiator to meet its required heat output can be determined using Equation 6 [20]:

$$q_m = \frac{\Phi_i}{c_s \cdot \vartheta_s - c_r \cdot \vartheta_r} \tag{6}$$

where:

q_m	Design mass flow rate	kg/s
Φ_i	Heat load of the room	W
ϑ_s	Temperature of supply water	°C
ϑ_r	Temperature of return water	°C
c_s	Specific heat capacity of water at supply temperature	J/(kg · K)
c_r	Specific heat capacity of water at return temperature	J/(kg · K)

Combining the equations presented above allows the required water flow rate for each radiator to be determined such that its heat output matches the room heat load calculated in accordance with DIN/TS 12831-1 [26, 41]. These calculations thereby form the basis for hydronic balancing.

2.2.3. Panel Heating

Panel heating systems, such as underfloor heating, present an alternative to radiators [23, 25, 26]. Their fundamental operating principle is comparable to that of radiators. Warm supply water is circulated through a heat exchanger, which in the case of underfloor heating consists of a long pipe loop embedded within the floor. As the water flows through the loop, it releases thermal energy to the room before returning to the heat generator for reheating [25, 26].

In contrast to radiators, panel heating systems have a substantially larger heat-emitting surface and operate at lower supply water temperatures. This is because the surface temperature of floors, ceilings, and walls should generally not exceed 33 °C for physiological reasons [24], whereas surface temperatures above 70 °C are considered standard for radiators [20]. While flow rate calculations for underfloor heating loops are not addressed in this work, the principles of hydronic balancing are equally applicable as for radiators [26, 41].

2.2.4. Hydraulic Resistance

To enable adjustment of heat emitter flow rates in a two-pipe heating system, a circulation pump must first be selected that is capable of delivering the total volumetric flow rate required by all emitters combined. The distribution of this total flow among the individual emitters is determined by hydraulic resistance. As water flows through the heating system, it encounters hydraulic resistance from pipes and other components, resulting in a pressure loss along each circuit. This pressure loss scales non-linearly with the flow rate through the circuits components. In a pipe network with parallel paths, such as a heating system, the flow rate through each path adjusts until the pressure loss across all paths is equal, as the pump imposes a fixed pressure difference across the network. Paths of lower hydraulic resistance therefore carry higher flow rates, since a higher flow rate is required to produce the same pressure loss as a high-resistance path. In a central heating system, pipe lengths and installed components typically differ between circuits, causing the hydraulic resistance of each path generally differs as well, resulting in an unequal flow distribution unless deliberate corrective measures are taken [26]. This behavior, analogous to the distribution of electric current among parallel resistors, is the reason why hydraulic resistance is an fundamental concept in hydronic balancing.

For a circuit, the total hydraulic resistance is the sum of all pressure losses along the path from the heat generator to the emitter and back [1, 26, 40]. Every pipe section causes such a pressure loss as a result of internal surface roughness and friction between the moving fluid and the pipe wall [1, 26, 32, 34]. Beyond straight pipe runs, each component the water passes through, such as bends, junctions, and emitters, further lower the pressure. In addition, valves may be installed to deliberately increase hydraulic resistance in an adjustable manner. It should be noted, however, that even fully opened valves introduce a non-negligible pressure loss. The pressure loss caused by pipes and hydraulic components depends nonlinearly on the flow rate, with higher flow rates resulting in disproportionately higher pressure losses [1, 26, 41].

2.2.5. Pipe Friction

The pressure loss caused by friction in a straight pipe section is commonly expressed using the Darcy friction factor λ. Based on this dimensionless coefficient, the pressure loss per meter of pipe can be calculated [1]:

$$R_L = \lambda \cdot d_i \cdot \frac{\rho \cdot v^2}{2} \tag{7}$$

where:

R_L	Pressure loss per meter of pipe due to friction	Pa/m
λ	Darcy friction coefficient	-
d_i	Inner pipe diameter	m
ρ	Fluid density	kg/m^3
v	Mean flow velocity	m/s

The value of λ depends on the flow regime, which is characterized by the dimensionless Reynolds number Re [1, 5]:

$$Re = \frac{v \cdot d_i}{\nu} \tag{8}$$

where:

Re	Reynolds number	-
v	Mean flow velocity	m/s
d_i	Inner pipe diameter	m
ν	Kinematic viscosity of the fluid	Pa · s

Three distinct regimes are defined based on the value of Re, namely laminar flow, turbulent flow, and a transition regime between the two [5]. In the laminar regime, which occurs at low Reynolds numbers ($Re < 2000$), flow is well-ordered and λ can be determined analytically as a function of Re alone [1, 38]. In the turbulent regime, which occurs at higher Reynolds numbers ($Re > 4000$), flow is disordered and λ depends on both Re and the relative roughness of the pipe wall [5, 32]. Several empirical correlations exist for this regime depending on the relative roughness of the pipe wall, including the Prandtl equation for hydraulically smooth pipes [1, 5, 31, 32], the Nikuradse equation for fully rough pipes [1, 5, 32], and the Colebrook-White equation for the intermediate case [13, 32]. It is worth noting that the threshold Re values describing the boundaries of the laminar and turbulent regimes are not universally agreed upon. The values presented here are a from Moody [32].

The transition regime, occurring approximately in the range $2000 \leq Re \leq 4000$, is characterized by intermittent switching between laminar and turbulent behavior. Due to this instability, λ cannot be predicted reliably by a single analytical expression in this regime [32]. A common engineering approaches to handling this regime is conservative estimation by treating the flow as turbulent [1, 31]. It is worth noting, however, that even within the fully turbulent regime, all available correlations provide estimates rather than exact values for the friction coefficient λ [32]. This is further underlined by the existence of multiple established correlations applicable to the same flow conditions, which can yield values differing by several percent or more from one another.

As the Reynolds number depends in part on the kinematic viscosity ν of the fluid, which varies substantially with temperature in water [31], pipe friction losses in water-based heating systems are sensitive to the operating water temperature.

2.2.6. Static Hydronic Balancing

Static hydronic balancing describes a method of hydronic balancing that is achieved by installing adjustable balancing valves in each circuit, which allow the hydraulic resistance of that path to be tuned. Increasing the hydraulic resistance of a specific circuit reduces the flow rate through that path while causing a redistribution of flow to the remaining

paths. By appropriately adjusting all balancing valves, the desired flow rate can be established for each individual circuit, and thus for each heat emitter [26, 41].

In radiator-based systems, balancing valves are often implemented as statically adjustable thermostatic radiator valves (TRVs), installed directly on the inlet of the radiator. These valves provide a preset resistance setting used for hydronic balancing, which is adjusted once during commissioning and remains unchanged during operation. In addition, they include a user-operated control function that allows occupants to regulate the heat output as required. A statically adjustable return valve may additionally be installed on the outlet of the radiator, either as a supplement to or a replacement for the TRV preset function [1, 15, 26, 41].

If not specified by the system design, the required water flow rate of each circuit must be calculated, as described for radiators in Section 2.2.2. Based on the flow rate, the pressure loss of each circuit is determined, assuming all balancing valves are fully open.

Having calculated the pressure losses of all circuits, the circuit with the highest pressure loss is determined to be the index circuit. Hydronic balancing is then performed by adjusting the balancing valves of all remaining circuits so that their hydraulic resistance approximates that of the index circuit. As balancing valves typically provide only discrete adjustment settings, an exact match in resistance is generally not possible and can only be approximated [26, 41].

When choosing the settings for the balancing valves, another key concept to consider is valve authority. Valve authority describes the ratio between the pressure drop across the balancing valves installed at a heat emitter, and the total pressure drop of the associated hydraulic circuit at the design flow rate [26]:

$$a_v = \frac{\Delta p_e^{\mathrm{bal}}}{\Delta p_e^{\mathrm{bal}} + \Delta p_e^{\mathrm{br}}} \tag{9}$$

where:

a_v	Valve authority	-
$\Delta p_e^{\mathrm{bal}}$	Combined pressure loss of the thermostatic and return valve of emitter e	Pa
Δp_e^{br}	Total pressure loss of all remaining components in the circuit of emitter e, excluding the thermostatic and return valve	Pa

Sufficient valve authority is important because it ensures that adjustments made at the valve have a meaningful and predictable effect on the flow rate. If the pressure drop across the valve is too small compared to the remaining circuit losses, changes in valve position will have little influence, making precise operation difficult or unstable. Conversely, excessive valve authority can make statically balanced system overly sensitive, such that small changes in valve position lead to large variations in flow rate [1, 26, 41]. The E DIN 94679-1 draft recommends ensuring valve authority is within a range of 30% to 70% for statically balanced systems [26].

Appropriate valve authority is ensured by choosing a static setting that introduces a sufficient but not excessive pressure loss [26, 41]. Usually, the valve of the index circuit is fully opened, but to ensure sufficient valve authority, it can be necessary to increase the pressure loss of even the index circuit. By the nature of the index circuit, this also results in all other circuits requiring more hydraulic resistance. In systems with large pressure discrepancies between circuits, it may be useful to install branch balancing valves in addition to adjustable TRVs [41]. Such valves allow part of the required pressure drop to be shifted away from the control valves, thereby avoiding excessive valve authority.

Since static hydronic balancing identifies the maximum pressure loss expected within the system, namely that of the index circuit, the resulting information can also be used to select an appropriate circulation pump for the heating system. This is crucial for efficient operation, as an undersized pump may fail to provide sufficient flow under conditions of high hydraulic resistance, while an oversized pump may operate inefficiently [26].

2.2.7. Dynamic Hydronic Balancing

While static hydronic balancing establishes the desired flow distribution under design conditions by means of fixed valve settings, dynamic hydronic balancing extends this concept to varying operating conditions. In practise, heating systems do not constantly operate at design conditions, due to flow demands change continuously as a result of user interaction, partial load operation, occupancy levels, and fluctuating supply temperatures and outdoor temperatures. Under such conditions, a statically balanced system may deviate from its intended flow distribution [26, 41].

Dynamic hydronic balancing addresses this limitation by employing components that automatically adapt to changing pressure conditions. This can be achieved through pressure independent control valves (PICVs), which maintain a predefined flow rate across a range of differential pressures. By internally compensating for pressure fluctuations in the system, these devices ensure that each heat emitter receives its intended design flow regardless of the operating state of neighboring circuits [26, 41].

Dynamic hydronic balancing is performed by first determining the design flow rate required for each circuit and then setting this value directly at the valve. Hydraulic pressure calculations are only required to verify that the resulting pressure conditions lie within the operating limits of the selected valves and to determine the pressure head that must be provided by the system pump. Due to the usage of PICVs, valve authority does not need to be verified, as it is inherently equal to 100% [26].

3. Design

To implement the heat load calculations and the subsequent processes of hydronic balancing and heating system optimization in the form of a software library, requirements must be formulated to inform subsequent design and implementation decisions. The requirements are divided into functional and non-functional. The functional requirements are derived from the obligations stated in §60c of the GEG [11]. The measures mandated by paragraph §60c are room-level heat load calculations in accordance with DIN/TS 12831-1 [23], an assessment of whether the heating system can be optimised by reducing the supply water temperature, and the execution of hydronic balancing. The non-functional requirements in contrast define quality criteria for the implementation itself.

3.1. Functional Requirements

The functional requirements stated in the following formulate the capabilities the library shall provide. These requirements are formulated to be verifiable and traceable to the underlying engineering procedures and standards discussed previously.

F1: Standard-compliant heat load calculation The library shall calculate room-level and building-level heat loads in accordance with DIN/TS 12831-1:2020-04 for both existing residential buildings and new constructions.

F2: Hydronic balancing The library shall determine the design mass flow rates, hydraulic pressure losses, and required valve settings for hydronic balancing of two-pipe heating systems. These calculations shall be based on the previously determined heat loads, following the principles described in VDI 2073 Part 2 and the draft standard E DIN 94679-1:2022-10.

F3: Radiator performance assessment For radiator-based systems, the library shall assess whether existing radiators are capable of covering the calculated room heat loads under the specified hydraulic operating conditions.

F4: Hydraulic constraint verification The library shall verify whether the required mass flow rates and valve settings can be realized without violating hydraulic constraints, in particular with respect to valve authority and maximum permissible water velocities.

F5: Supply temperature optimization As mandated by the GEG [11], the library shall attempt to improve the efficiency of the heating system by reducing the supply water temperature. As the radiators of an existing radiator-based heating system may be insufficiently dimensioned for operation at lower supply temperatures, the library shall generate technically feasible replacement recommendations aimed at minimizing cost while maximizing the reduction in required supply temperature.

Together, these requirements ensure that all measures mandated by the GEG [11] are implemented in a compliant manner.

3.2. Non-Functional Requirements

In addition to the functional capabilities, the library must satisfy a set of non-functional requirements. These requirements address aspects such as performance, extensibility, portability, and interface independence, and ensure that the library can be integrated into different application contexts while maintaining correctness and reliability.

N1: Performance and responsiveness All calculations shall complete within a time frame compatible with interactive use, ensuring that users do not experience noticeable delays during typical application workflows. The library shall avoid computational approaches that impose excessive memory usage or computational load, ensuring suitability for deployment across a range of hardware configurations.

N2: Maintainability and readability The codebase shall prioritize clarity, consistent structure, and variable naming aligned with relevant technical standards in order to facilitate verification by third parties, maintenance, and future development.

N3: Testability and verifiability The library shall be designed to enable testing of individual components and verification of results against reference calculations and normative examples.

N4: Programmatic and interface-independent access The library shall provide its functionality through a programmatic API that is independent of any specific user interface, enabling integration into different platforms and ecosystems.

N5: Idiomatic Swift design The implementation shall follow established Swift development conventions, making appropriate use of strong typing, value semantics, and structured concurrency where applicable.

3.3. Module Scopes

The library is structured into two separate modules, one for heat load calculation (**F1**) and one for hydronic balancing (**F2-F5**). This separation reflects the limited coupling between the two processes, which is visualized by Figure 3. Both modules share only a single common set of input variables, namely the target indoor room temperatures of each heated room. Beyond this shared input, the hydronic balancing module does not require the full set of intermediate results produced by the heat load module. Instead, it relies solely on the resulting total room heat load as its relevant input. By decoupling the modules, each process can be implemented, validated, and reused independently, while maintaining a clear and minimal data dependency between the two processes.

While the room heat load is the only output of the heat load module required for hydronic balancing, the module nonetheless computes all results necessary to generate a complete heat load report in accordance with DIN/TS 12831-1 [23]. However, as this work focuses exclusively on the underlying calculations, the development of user interfaces or export functionalities, such as PDF report generation, are outside the scope of this work.

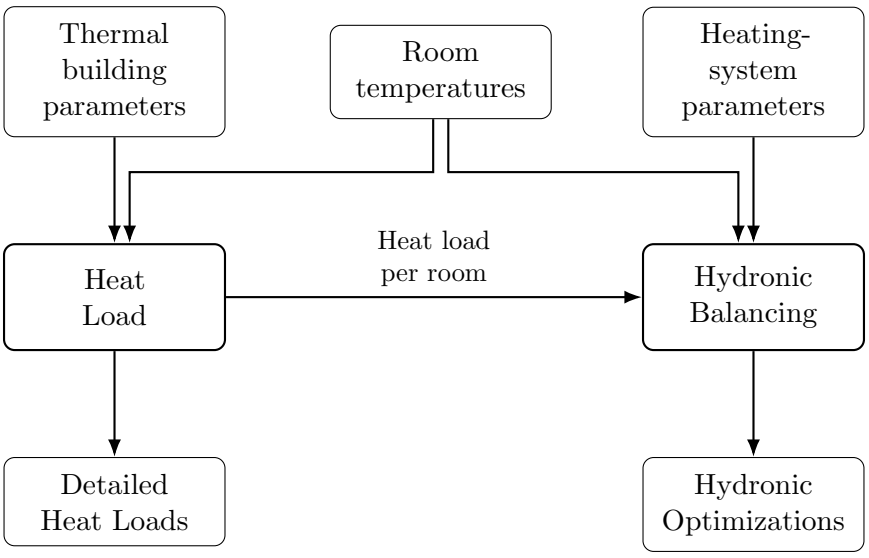

Figure 3: Modular architecture of the library, consisting of a Heat Load module and a Hydronic Balancing module. The heat load calculation processes thermal building parameters and room temperatures to determine the detailed room heat loads. Hydronic balancing builds upon this result, using only the total room heat load combined with heating-system parameters and the shared room temperature input to compute the corresponding valve settings.

3.4. Heat Load Module

The heat load module implements the calculation of room-level and building-level heat loads in accordance with DIN/TS 12831-1 [23]. The standard provides three calculation approaches, a standard procedure and two simplified procedures. The standard procedure is characterized by a high level of input detail and is primarily intended for application in new constructions or in extensive reconstruction measures. It provides both room-level and building-level heat load results within a single calculation framework [21, 23].

In contrast, both simplified procedure are limited to one of the two levels. The simplified procedure for room-level heat loads is intended for projects involving the replacement of heat emitters or the performance of hydronic balancing. The simplified procedure for building-level heat loads is intended for cases in which only the heat generator is to be replaced. The two simplified procedures are mutually exclusive. If the heat generator is to be replaced in addition to changes to emitters or hydronic balancing, this constitutes an extensive reconstruction measure and the standard procedure must be applied. Furthermore, the simplified procedures may only be applied to existing residential buildings, or buildings of similar use, without a mechanical ventilation system. For buildings with a mechanical ventilation system, the standard procedure must always be applied [21, 23].

3.4.1. Standard Procedure

For the standard procedure, building data are structured across four hierarchical levels, illustrated by Figure 4. At the building level, global properties such as the heat storage capacity and air tightness of the building are defined. Since climate data is also a global property, information such as the design outdoor temperature is also stored at the building level. The next level is the ventilation zone, where information related to the ventilation system, if present, and other zone-specific parameters, such as the presence and design of air transfer devices (ATDs), are specified. Subsequently, at the room level, parameters such as the target indoor temperature, floor area, and internal volume are provided. Finally, at the component level, detailed information on individual building elements is defined, including U-values, surface areas, and the type of adjacent environment (e.g., outdoor air or unheated spaces). This structure follows the hierarchical organization defined in DIN EN 12831-1 [21, 23].

A building may contain multiple ventilation zones, and each ventilation zone may contain multiple rooms. Within the DIN EN 12831-1 standard, buildings and ventilation zones without at least one assigned room, as well as rooms not associated with a ventilation zone, are conceptually invalid [21, 23]. Consequently, every building must contain at least one ventilation zone, and each ventilation zone must include at least one room.

Rooms typically consist of multiple components that define their thermal boundaries to adjacent environments. However, a room may be valid even if no components are specified. This situation occurs in interior rooms where all bounding surfaces are adjacent to spaces maintained at the same temperature. Since components with identical temperatures on both sides do not contribute to heat transmission, they may be omitted, allowing the existence of rooms without explicitly defined components [21, 23].

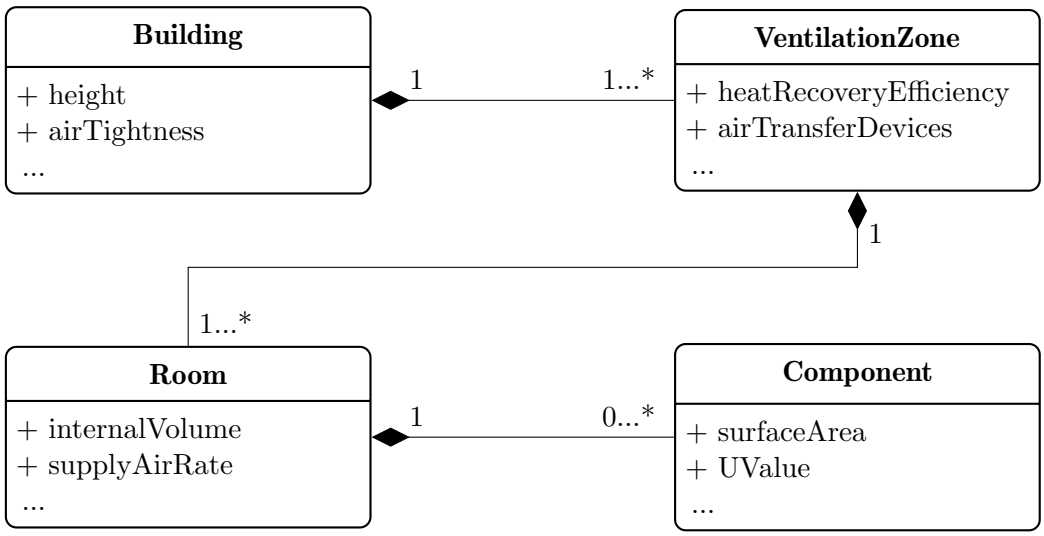

Figure 4: Class diagram illustrating the hierarchical data structure used for heat load calculation. The listed properties (e.g., `height`, `airTightness`, `internalVolume`, `UValue`) are exemplary.

The overall calculation process can be structured into four sequential phases. The entire process is coordinated by the building instance, as it is at the top-level of data structure hierarchy. The following paragraphs describe the data flow within each phase and the transformation of input data into intermediate and final results.

Phase 1 In the first phase, the airflow rates within each ventilation zone are determined, with the data flow illustrated in Figure 5a. The building instance passes climate information, such as the design outdoor temperature, and building parameters such as air tightness, to each ventilation zone. The ventilation zones subsequently collect room-level data, including envelope area and internal volume, where the envelope area of a room is derived from the surface areas of its components. Room-specific ventilation parameters, such as the supply and exhaust air flow rates, are likewise stored at the room level and retrieved by the corresponding ventilation zone.

The resulting zone-level airflow rates are passed back to the building instance for use in the subsequent phases. The data flow within this phase is strictly hierarchical, as data are exchanged only between directly related levels, without horizontal communication between entities of the same hierarchy level.

Phase 2 The second phase uses the airflow rates determined in phase 1 together with thermal building properties to calculate the time constant of the building. The time constant is derived from the relationship between the thermal mass of the building and an approximation of its overall heat loss rate. As described in Section 2.1.4, the time constant may influence the adjusted design outdoor temperature, which in turn influences the transmission heat loss calculations of the next phase. As this phase primarily involves the transformation of already aggregated airflow rates and thermal parameters of the building itself, this phase is handled internally by the building instance and requires minimal dataflow between the types.

Phase 3 In the third phase, room-level heat loads are calculated by combining transmission and ventilation heat losses. Here, zone-level parameters are no longer required, as the relevant information is already included in the zone-level airflow rates calculated in phase 1. Consequently, the ventilation zone level is bypassed. Instead, the building iterates directly over all rooms aggregated from the zones.

An overview of the data flow within this phase is provided in Figure 5b. Transmission losses are determined at the component level using component-specific data such as U-values, surface areas, and boundary conditions, together with the adjusted design outdoor temperature obtained in phase 2. These component-level results are subsequently aggregated to determine the transmission loss at room level. Ventilation losses are calculated at the room level using room-level parameters such as internal volume and supply or exhaust airflow rates, in combination with the zone-level airflow rates determined in phase 1, which are passed to the rooms by the building instance.

The outputs of this phase are the total heat loads for each room, including intermediate results. These results are passed to the building level for aggregation in the subsequent

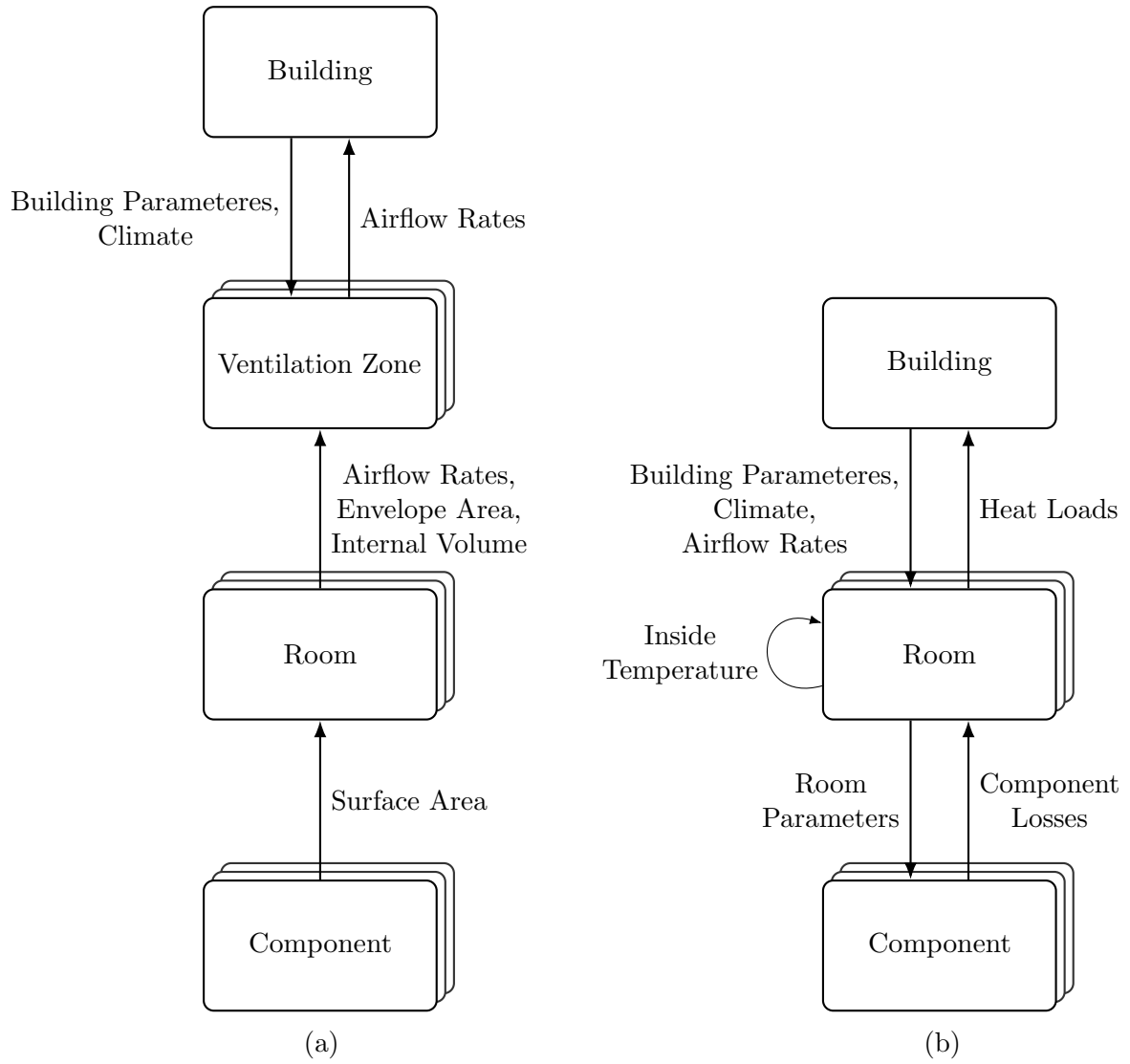

Figure 5: Data flow diagrams for phases 1 and 3 of the heat load calculation. Subfigure (a) illustrates the data flow across all four levels of the hierarchy during phase 1. Subfigure (b) illustrates the data flow during phase 3, in which the ventilation zone level is bypassed, and shows the horizontal data exchange between rooms required to resolve adjacent indoor temperatures, represented as a self-loop.

phase and constitute the primary input required for hydronic balancing. As shown in Figure 5b, the data flow in this phase remains largely vertical. An exception arises when calculating heat transfer between rooms at different temperatures. For transmission across internal components or inter-room airflow, rooms must reference the temperature of adjacent rooms, introducing limited horizontal data exchange, which is illustrated by a self-loop in Figure 5b.

Phase 4 In the final phase, the room-level heat loads are aggregated to determine ventilation zone-level and overall building heat loads. Since the room-level heat loads were already passed to the building in the previous phase, this phase consists of structured aggregation without additional heat loss calculation. Therefore, only minimal, strictly vertical dataflow occurs between the ventilation zones and the building in order to determine the share of the total heat load caused by each ventilation zone.

While the data flow in phases 1, 2, and 4 is strictly vertical, phase 3 requires limited horizontal data exchange between rooms. This introduces a design challenge, as the hierarchical data structures shown in Figure 4 needs to be extended to allow rooms to refer to each other. The resolution of this challenge is discussed in Section 4.2.1.

3.4.2. Simplified Procedures

In contrast to the standard procedure, the simplified procedures require substantially less computational effort and smaller data structures. Both simplified approaches omit detailed airflow calculations [21, 23], bypassing most of the calculations performed in the first two phases of the standard procedure. Furthermore, the concept of ventilation zones is not applied. In the room-level simplified procedure, room heat loads are not aggregated to building level, eliminating the fourth phase of the standard procedure.

Similar to the standard procedure, the simplified room-level procedure uses a hierarchical data structure consisting of a variable number of rooms, which in turn contain components. The key differences are the drastic reduction in per-entity parameters and the simplification of the overall computational process. For instance, while a single room requires at least 15 parameters in the standard procedure, only 4 parameters are required in the simplified procedure, with component-level parameters considered separately. Furthermore, for the simplified calculation, DIN EN 12831-1 defines only one equation each for transmission losses and ventilation losses. The total room heat load is obtained as the sum of these two contributions, optionally including additional loads as described in Section 2.1.5 [21].

For the simplified building level procedure, just a building and component type are defined. All components that define the buildings envelope are added to the building instance. The calculation of the building heat load is then carried out analogous to that of a single room in the simplified room-level procedure [21].

In contrast to the standard procedure, the data structures for both simplified procedures introduce no horizontal dependencies. As a result, the strictly hierarchical data structure can be preserved without the need for architectural adaptations.

3.5. Hydronic Balancing Module

The hydronic balancing module further processes the room-level heat loads determined by the heat load module, and uses them to optimize the heating system of the building in accordance with the GEG [11] and applicable technical rules and standards such as E DIN 94679-1 [26] and VDI 2073 Part 2 [41]. It fulfils functional requirement **F2** by performing the hydronic balancing calculations for two-pipe systems, and requirements **F3**

and **F4** by verifying the heating systems compliance with constraints related to radiator heat output and hydraulic parameters such as maximal flow velocities and valve authority. Finally, the module determines the lowest achievable supply water temperature for the system and generates cost-optimised, technically feasible recommendations for radiator and pipe replacement in cases where insufficient dimensioning prevents operation at reduced supply temperatures, thereby satisfying requirement **F5**.

To define how hydronic balancing must be performed, the GEG refers to a procedure specified in a technical code of practice published by the Zentralverband Sanitär Heizung Klima (ZVSHK), the German national association of the sanitation, heating and air-conditioning trade [11, 45, 47]. Besides technical suggestions, such as ensuring that the heating system has an appropriate amount of system shutoff valves for maintenance, the procedure states that a pipe network calculation is required for hydronic balancing. In the context of water-based heating systems, a pipe network calculation involves modeling the hydraulic behavior of supply and return lines, fittings, valves, and heat emitters. The calculation accounts for frictional pressure losses in pipe sections, additional local losses caused by components such as bends and valves, and the interaction between different hydraulic circuits within the same heating system.

3.5.1. Heating System Graph

To support the pipe network calculation, the module models the two-pipe heating system as a graph consisting of pipes and system elements. System elements include heat emitters, branch valves, connectors, and a single heat generator. As the module focusses solely on central heating, the graph is constrained to a single heat generator. Connectivity within the graph is modeled at port level, as shown in the example graph of Figure 6. Instead of linking elements directly, each pipe forms an edge between two ports belonging to different elements. Examples of ports include the inlet and outlet ports of a heat emitter and the three ports of a tee fitting.

Defining edges at port level rather than at element level is particularly important to calculate the pressure losses caused by connectors with more than two ports, such as tee fittings. Due to the inherent asymmetry of some connectors, each port may exhibit different hydraulic characteristics depending on flow direction. Accurate resistance calculation along a hydraulic path therefore requires explicit knowledge of which pipe is connected to which specific port.

The graph is required to be connected, since elements not indirectly linked to the heat generator would serve no functional purpose in the heating system. Each port must be connected to exactly one pipe, as a port with no connection or multiple connections has no physical equivalent in a real heating system. If multiple connections to a single port are required, an explicit connector element, such as a tee fitting, must be inserted. Furthermore, connections may only be established between ports belonging to different elements, as the ports of a single element are assumed to be internally connected within that element. Consequently, flow conservation cannot be formulated at the individual port level, as a port does not represent a junction where multiple flows meet. Instead, flow conservation is defined over the sum of flows across all ports of a single element.

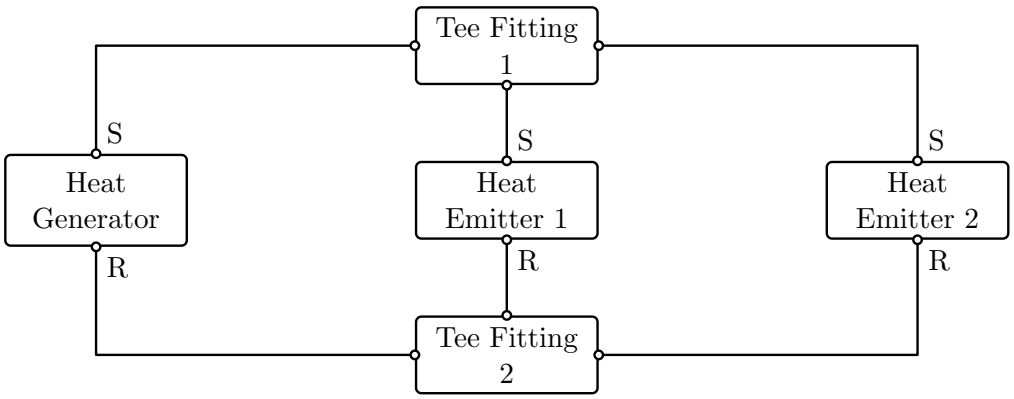

Figure 6: An exemplary heating system graph illustrating port level modeling. System elements are shown as rectangles, with attached circles visualizing ports. Pipes are shown as lines between ports. The first tee fitting splits the flow from the heat generator into two parallel emitter circuits, while the second tee fitting merges the two return flows before they re-enter the heat generator.

Since the graph models a two-pipe heating system, it can be decomposed into two directed subgraphs representing the supply and return networks. The only contact points between the supply and the return subgraphs are the heat generator and the emitters. Direct connections via a pipe between the two subgraphs are not permitted, as such a connection would create a hydraulic shortcut between supply and return. The hydronic balancing procedures described in E DIN 94679-1 and VDI 2073 Part 2 assume a single, unique hydraulic circuit between each emitter and the heat generator [26, 40]. Under this assumption, both subgraphs can be considered acyclic, as otherwise multiple path including the same emitter would exist. Due to the combined constraints on the graph, both the supply and return subgraph exhibit a tree structure, with the heat generator at the root. This is further supported by engineering literature stating that in the supply network of a two-pipe system flow is exclusively divided, whereas in the return network flow is exclusively merged [1].

The system element and pipe instances contained in the graph collectively provide the input parameters required for pipe network calculations and hydronic balancing. Pipe instances contain properties related to the calculation of their friction losses, such as length, diameter, and surface roughness. The module further defines four types of system elements, which are listed in Table 1 along with a selection of their most notable properties.

3.5.2. Pipe Network Calculation

Before performing hydronic balancing, the module verifies whether each radiator can deliver the heat output required to cover the heat load of the corresponding room. As defined in requirement **F3**, this assessment applies to radiators only, underfloor heating emitters are assumed to provide sufficient heat output by design. As discussed in Section 2.2.2, the excess temperature required for a radiator to meet its required power

Table 1: System element types and an exemplary selection of their stored properties.

Element Type	**Exemplary Properties**
Heat Emitter	Nominal and required heat output, available TRV and (optional) return valve settings, resistance coefficient
Heat Generator	Preferred supply temperature, preferred return temperature, resistance coefficient
Connector	Number of ports, direction-dependent resistance coefficients
Branch Balancing Valve	Available settings and their corresponding resistance values

output can be calculated based on the input data stored in the graph. If the calculated excess temperature exceeds what is achievable under the specified supply conditions, the radiator is deemed insufficient and the process is aborted, with the reason for the failure returned to the caller. This validation step implements requirement **F3**.

The balancing procedure begins by identifying all emitter circuits within the graph. Assuming a well-formed graph, there is exactly one circuit per heat emitter, each defined by a supply path from the heat generator to the emitter and a return path back. Due to the port-level modeling of the heating system graph, each path is expressed as an ordered list of connections, where each connection is an ordered tuple of two ports belonging to different elements.

After identifying all emitter circuits, the required mass flow rate for each radiator, and therefore for each corresponding circuit, is calculated as described in Section 2.2.2. For circuits using underfloor heating, the mass flow rate is given as an input parameter. By aggregating the flow rates of all circuits, the flow rate through each pipe and each system element can be determined. In accordance with requirement **F4**, water velocities in each pipe is be verified against the maximum velocity defined for the pipe at this point. Maximum flow velocities are imposed on pipes both to protect pipe integrity and, in some cases, to limit noise levels in occupied spaces [26].

Based on these flow rates and hydraulic input parameters such as pipe diameter and roughness, the pipe network calculation can be performed. For this, the pressure loss caused by every pipe and system element is calculated individually and stored alongside intermediate results such as water velocity or dynamic pressure. This initial calculation ignores resistances caused by branch or emitter balancing valves, as no balancing has been performed yet. The total pressure loss along a given path is obtained by summing the individual pressure losses along that path. The total pressure loss of an emitter circuit is determined through accumulating of the pressure losses along its supply and return paths.

For dynamic hydronic balancing, the procedure is complete at this stage, as valves such as PICVs are adjusted directly using the previously calculated mass flow rates [26, 41]. The computed pressure losses are only required for the installer to verify that the selected valve model operates within its specified pressure range [26].

In contrast, static hydronic balancing does not rely on self-regulating components. Instead, the required flow distribution must be achieved indirectly through the adjustment of hydraulic resistances. Consequently, the calculated pressure losses form the basis for determining appropriate valve settings [26, 41].

3.5.3. Static Hydronic Balancing

To complete the static hydronic balancing procedure, a setting must be assigned to each adjustable valve such that the difference between the pressure losses between all emitter circuits are minimized. This adjustment must not violate valve authority constraints and should avoid unnecessary increases in the overall hydraulic resistance of the system [26, 41].

At least one statically adjustable balancing valve must be installed at each heat emitter, typically a TRV at the inlet, optionally supplemented by a return valve at the outlet [14, 26, 41]. The following description assumes a TRV is present at every emitter, though configurations using only a return valve are equally supported, as both are functionally equivalent in this context. In larger systems, additional branch balancing valves may also be installed [41].

Statically adjustable valves typically specify their resistance values for discrete setting steps. Some valve models allow intermediate positions between these settings, however, this capability is not universally guaranteed [15, 26]. Therefore, the balancing algorithm offers support for both continuously and discretely adjustable valves.

Both E DIN 94679-1 and VDI 2073 Part 2 initiate the procedure by identifying the index circuit, defined as the emitter circuit exhibiting the highest pressure loss under the assumption of fully opened valves. Subsequently, the remaining circuits are to be adjusted such that their pressure losses approximate that of the index circuit. However, neither reference specifies an explicit method for selecting valve settings in systems in which multiple adjustable valves act in series on the same emitter circuit, as is the case when branch balancing valves are installed [26, 41].

In such configurations, the interaction between serially connected valves renders the adjustment problem non-trivial, as the pressure loss of a circuit is influenced by several coupled decision variables. Branch balancing valves may additionally influence multiple circuits, further increasing the coupling between variables. Moreover, balancing valves may only provide discrete resistance values based on adjustment positions, and additional constraints such as valve authority must be satisfied.

The hydraulic relationship between valve position and pressure loss is inherently nonlinear. However, the total pressure loss of a circuit is a linear combination of the individual pressure losses contributed by each valve, in addition to a fixed pressure loss caused by pipes and other system elements. To find a solution for hydronic balancing, it is therefore sufficient to represent each adjustable valve by a decision variable encoding its target pressure loss, from which the corresponding valve setting can be inferred once a solution is obtained. For valves offering only discrete settings, the target pressure loss is modeled using one binary decision variable per admissible setting, with the pressure loss at that setting as its coefficient. A constraint enforcing that exactly one of these

binary variables equals one ensures that their weighted sum equals the pressure loss of the selected setting. The total circuit pressure loss can therefore be expressed as a linear combination of these decision variables and a constant. Static hydronic balancing can consequently be formulated as a mixed-integer optimization problem with coupled decision variables and linear constraints, which is solved in the hydronic balancing module using a MILP formulation.

In the following, the total pressure loss of a emitter circuit c is denoted by Δp_e. At any stage of the procedure, one circuit exhibits the highest pressure loss and another the lowest. The corresponding pressure losses are denoted by $\Delta p_{\max}$ and $\Delta p_{\min}$ respectively. The hydronic balancing problem is formulated as a three-level lexicographic optimization problem with the following objectives:

Primary Objective The highest priority objective is minimizing $\Delta p_{\max}$. Although the valves on the circuit with the maximal pressure loss can be expected to be fully open, partial closing may be required to satisfy minimum valve authority constraints. In such cases, this objective ensures that valves are throttled only as much as strictly necessary to achieve the required authority. By minimizing $\Delta p_{\max}$, an upper bound on the circuit pressure losses is established, to which all circuits are subsequently adjusted in the next phases. This phase therefore reproduces the identification of the index circuit as described in E DIN 94679-1 and VDI 2073 Part 2 [26, 41]. This minimization being the primary objective reflects the fact that $\Delta p_{\max}$ directly determines the required pump head and any increase may negatively impact system efficiency.

Secondary Objective The secondary objective is the maximization of $\Delta p_{\min}$, thereby reducing the spread between $\Delta p_{\max}$ and $\Delta p_{\min}$. By raising the lower bound on circuit pressure losses, this objective contributes to the balancing goal implied by E DIN 94679-1 and VDI 2073 Part 2, namely that all circuits should approximate the pressure loss of the index circuit [26, 41]. However, this objective alone is not sufficient to ensure optimal balancing. While the first two phases together enforce $\Delta p_{\min} \leq \Delta p_e \leq \Delta p_{\max}$ for all emitter circuits, this does not guarantee that each individual circuit is adjusted as closely as possible to the pressure loss of the index circuit, i.e., to $\Delta p_{\max}$.

Tertiary Objective The tertiary objective is the maximization of $\sum_{e \in E} \Delta p_e$, subject to the constraints established by the preceding two phases. By maximizing the total pressure loss across all emitter circuits within the bounds imposed by $\Delta p_{\min}$ and $\Delta p_{\max}$, this objective drives each individual circuit pressure loss towards $\Delta p_{\max}$. Without the secondary objective, maximizing the total pressure loss alone could still yield solutions in which one circuit exhibits a disproportionately low pressure loss while the remaining circuits are closely aligned with the index circuit. The hierarchical combination of the secondary and tertiary objectives prevents such imbalanced solutions from being generated if alternatives exist.

Based on these objectives and constraints, the following MILP model is defined:

$$\text{lex min} \quad \Big(\Delta p_{\max},\ -\Delta p_{\min},\ -\textstyle\sum_{e\in E}\big(\Delta p_e^{\text{bal}} + \Delta p_e^{\text{br}} + \Delta p_{e,\text{th}}\big)\Big) \tag{10}$$

$$\text{s.t.} \quad \Delta p_e^{\text{bal}} + \Delta p_e^{\text{br}} \le \Delta p_{\max} \qquad \forall e \in E \tag{11}$$

$$\Delta p_e^{\text{bal}} + \Delta p_e^{\text{br}} \ge \Delta p_{\min} \qquad \forall e \in E \tag{12}$$

$$(1 - a_v^{\max}) \cdot \Delta p_e^{\text{bal}} \le a_v^{\max} \cdot \Delta p_e^{\text{br}} \qquad \forall e \in E \tag{13}$$

$$(1 - a_v^{\min}) \cdot \Delta p_e^{\text{bal}} \ge a_v^{\min} \cdot \Delta p_e^{\text{br}} \qquad \forall e \in E \tag{14}$$

$$\sum_{s\in S_{e,r}} x_{e,r,s} = 1 \qquad \forall (e,r) \in (E \times R) \cap \mathcal{D} \tag{15}$$

$$\sum_{s\in S_b} y_{b,s} = 1 \qquad \forall b \in B \cap \mathcal{D} \tag{16}$$

$$x_{e,r,s} \in \{0,1\} \qquad \forall (e,r) \in (E \times R) \cap \mathcal{D},\ \forall s \in S_{e,r} \tag{17}$$

$$y_{b,s} \in \{0,1\} \qquad \forall b \in B \cap \mathcal{D},\ \forall s \in S_b \tag{18}$$

$$\tilde{p}_{e,r} \in [\Delta p_{e,r}^{\min},\ \Delta p_{e,r}^{\max}] \qquad \forall (e,r) \in (E \times R) \cap \mathcal{K} \tag{19}$$

$$\tilde{p}_b \in [\Delta p_b^{\min},\ \Delta p_b^{\max}] \qquad \forall b \in B \cap \mathcal{K} \tag{20}$$

$$\Delta p_{\max} \in \mathbb{R}_{\ge 0} \tag{21}$$

$$\Delta p_{\min} \in \mathbb{R}_{\ge 0} \tag{22}$$

where $R = \{\text{th}, \text{re}\}$ is the set of valve roles (thermostatic and return), $R_e \subseteq R$ is the set of valves present for emitter e (always $\text{th} \in R_e$, optionally $\text{re} \in V_e$). The set of all heat emitters is denoted by E and the set of all branch valves by B. The set of all valves in the system, expressed as $\{(e,v) \mid e \in E, v \in V_e\} \cup B$ is partitioned by $\mathcal{D}$ and $\mathcal{K}$ into discrete and continuous valves respectively. The pressure loss of emitter and branch valves are defined as:

$$\Delta p_{e,r} = \begin{cases} \displaystyle\sum_{s\in S_{e,r}} \Delta p_{e,r,s} \cdot x_{e,r,s} & (e,r) \in \mathcal{D} \\ \tilde{p}_{e,r} & (e,r) \in \mathcal{K} \end{cases} \tag{23}$$

$$\Delta p_b = \begin{cases} \displaystyle\sum_{s\in S_b} \Delta p_{b,s} \cdot y_{b,s} & b \in \mathcal{D} \\ \tilde{p}_b & b \in \mathcal{K} \end{cases} \tag{24}$$

The pressure loss caused by emitter side balancing valves (th and re) is defined as:

$$\Delta p_e^{\text{bal}} = \sum_{r\in R_e} \Delta p_{e,r}, \qquad \forall e \in E \tag{25}$$

and the branch-side pressure of the circuit of an emitter as:

$$\Delta p_e^{\mathrm{br}} = \Delta p_e^{\mathrm{const}} + \sum_{b \in B_e} \Delta p_b \qquad \forall e \in E \tag{26}$$

E	Set of emitters (i.e., radiators and underfloor heating)
B	Set of branch valves
B_e	Set of branch valves on the hydraulic circuit of emitter e
R	Set of emitter valve roles $\{\mathrm{th}, \mathrm{re}\}$ (thermostatic and return)
R_e	Set of valve roles present for emitter e
$\mathcal{D}$	Set of (e, r) or b pairs with discrete valves
$\mathcal{K}$	Set of (e, r) or b pairs with continuous valves
$S_{e,r}$	Set of admissible settings of discrete emitter valve (e, r)
S_b	Set of admissible settings of discrete branch valve b
$\Delta p_{e,r,s}$	Pressure loss of emitter valve (e, r) at setting s
$\Delta p_{b,s}$	Pressure loss of branch valve b at setting s
$\Delta p_e^{\mathrm{const}}$	Fixed pressure losses in circuit of emitter e
$\Delta p_{e,r}^{\mathrm{min}}$	Minimum pressure loss of continuous emitter valve (e, r)
$\Delta p_{e,r}^{\mathrm{max}}$	Maximum pressure loss of continuous emitter valve (e, r)
$\Delta p_b^{\mathrm{min}}$	Minimum pressure loss of continuous branch valve b
$\Delta p_b^{\mathrm{max}}$	Maximum pressure loss of continuous branch valve b
a_v^{min}	Minimum valve authority
a_v^{max}	Maximum valve authority
$x_{e,r,s}$	Binary variable equal to 1 iff emitter valve (e, r) is at setting s
$y_{b,s}$	Binary variable equal to 1 iff branch valve b is at setting s
$\tilde{p}_{e,r}$	Continuous pressure loss variable for emitter valve (e, r)
$\tilde{p}_b$	Continuous pressure loss variable for branch valve b
$\Delta p_{e,r}$	Pressure loss of emitter valve (e, r) (discrete or continuous)
$\Delta p_e^{\mathrm{bal}}$	Combined pressure loss of all emitter valves of circuit e
Δp_e^{br}	Sum of fixed and branch valve pressure losses in circuit e
Δp_{max}	Objective variable, maximum circuit pressure loss across all emitters
Δp_{min}	Objective variable, minimum circuit pressure loss across all emitter circuits

The model is formulated as a lexicographic optimization problem with three priority levels. At the first level, the objective function (10) minimizes $\Delta p_{\max}$, i.e. the maximum total pressure loss among all emitter circuits. At the second level, among all solutions optimal with respect to $\Delta p_{\max}$, the minimum circuit pressure loss $\Delta p_{\min}$ is maximized. At the third level, among all solutions that are optimal with respect to the first two objectives, the sum of circuit pressure losses is maximized. The additional term $\Delta p_{e,\text{th}}$ gives a secondary preference to throttling the thermostatic valve before the return valve in cases where both are adjustable, thereby reducing symmetry in the solution space and improving solution stability. This preference of closing the thermostatic valves first is consistent with the behavior observed in ZVPlan software [14].

Thereby, the largest pressure loss of any circuit in the system is minimized. Constraints (13) and (14) are derived from Equation (9) and ensure that the valve authority of each emitter valve is within the desired range. Finally, constraints (15) and (16) ensure that exactly one valve setting is selected for each emitter and branch valve respectively.

From the optimal solution obtained by solving the model, the valve settings for each adjustable valve can be determined. For discretely adjustable valves, the optimal setting is encoded in the binary variables $x_{e,r,s}$ and $y_{b,s}$, of which exactly one per valve is equal to one, identifying the selected setting. For continuously adjustable valves, the optimal solution yields a pressure loss value stored in variables $\tilde{p}_{e,r}$ and $\tilde{p}_b$, from which the corresponding valve setting can be calculated through linear interpolation of the setting position specified in the data sheet of the valve. With these valve settings applied, the pipe network calculation can be repeated to verify the results obtained from the model, concluding the static hydronic balancing procedure.

3.5.4. Supply Temperature Optimization

In addition to hydronic balancing, the module implements requirement **F5** by determining the lowest feasible supply water temperature for radiator-based systems. Lowering this temperature reduces the heat output of each radiator, which must be compensated by increasing the volumetric flow rate. However, this compensation is subject to a upper bound imposed by physical and technical constraints.

The excess temperature ΔT required for a radiator to meet its heat output is determined by the ratio of its required to its nominal heat output. As ΔT is related to both supply and return water temperature (see Section 2.2.2), a reduction in supply temperature at constant heat output results in an increase in return temperature. Depending on the installed heat generator, a minimum and maximum spread between supply and return temperature must be maintained. For example, condensing boilers generally favour a larger temperature spread, whereas heat pumps favour a smaller one [1]. Increasing flow rates further raises the water velocity within the pipe sections of the system. This flow velocity is subject to upper limits imposed both to protect pipe integrity and, in some cases, to limit noise levels in occupied spaces [26]. As a result, achieving a lower supply temperature may require replacing existing radiators with larger models that deliver the required heat output at reduced flow rates, or replacing pipe sections that would otherwise exceed the maximum permissible water velocity.

The approach presented here, outlined in Algorithm 1, iterates over supply temperatures starting from the current design value and reducing it in steps of 0.5 K by default. At each iteration, a pipe network calculation is performed to verify whether the current system configuration satisfies the pipe velocity and temperature spread constraints at the iteration supply temperature. If all constraints are satisfied, the current configuration is recorded as a feasible solution. Otherwise, a MILP model is solved to determine the minimum-cost combination of component replacements that restores feasibility.

Algorithm 1: Supply temperature optimization

Require: Heating system graph G, minimum supply temperature ϑ_s^{min}, maximum cost c_{max}, step size δ
Ensure: List of feasible solutions at decreasing supply temperatures
$\vartheta_s \leftarrow$ design supply temperature of G
solutions $\leftarrow []$
$(G', c) \leftarrow (G, 0)$
while $\vartheta_s \geq \vartheta_s^{min}$ **do**
    $\vartheta_s \leftarrow \vartheta_s - \delta$
    if G' satisfies pipe velocity constraints at ϑ_s **then**
        Append (G', ϑ_s, c) to solutions
    else
        $(G', c) \leftarrow \text{SolveMILP}(G, \vartheta_s)$
        if MILP infeasible **then return** solutions
        end if
        Append (G', ϑ_s, c) to solutions
        if $c \geq c_{max}$ **then return** solutions
        end if
    end if
end while
return solutions

For each radiator, alternative higher-output models are selected from a provided catalog of available radiator types. Only models that exhibit the required temperature spread at the iteration supply temperature are considered. For each pipe section, upsizing is likewise considered. As heating system pipes in residential buildings are commonly concealed within walls [1, 18], pipe sections must be explicitly marked as replaceable in order to be considered by the algorithm.

If a feasible solution is found, the modified graph is stored and the algorithm proceeds to the next iteration, continuing until a specified minimum supply temperature or maximum replacement cost is reached. If the MILP model becomes infeasible, the algorithm terminates and returns the solutions found up to that point, as infeasibility at a given temperature implies that no valid replacement combination exists at any lower temperature. The full MILP formulation is presented in the following:

$$\min \quad \sum_{h \in H} \sum_{m \in M_h} c_{h,m} \cdot x_{h,m} + \sum_{p \in P^*} c_p \cdot y_{p,d} \tag{27}$$

$$\text{s.t.} \quad \sum_{h \in H_p} \sum_{m \in M_h} q_{h,m} \cdot x_{h,m} \leq q_{p,\text{base}} + \Delta q_p \cdot y_p \qquad \forall p \in P \tag{28}$$

$$\sum_{m \in M_h} x_{h,m} = 1 \qquad \forall h \in H \tag{29}$$

$$x_{h,m} \in \{0,1\} \qquad \forall h \in H,\ \forall m \in M_h \tag{30}$$

$$y_p \in \{0,1\} \qquad \forall p \in P^* \tag{31}$$

The required mass flow of emitter h under model m is given by $q_{h,m}$, and the flow capacity of pipe p and diameter d is defined as [1, 26]:

$$q_{p,d} = \frac{1}{4} \cdot v_p^{max} \cdot \rho \cdot d^2 \cdot \pi \tag{32}$$

The flow capacity of a pipe at its current, unchanged diameter is denoted as $q_{p,base}$. The incremental flow capacity gained by upsizing pipe p to inner diameter d is:

$$\Delta q_p = \max_{d \in D_p}(q_{p,d}) - q_{p,\text{base}} \tag{33}$$

E	Set of radiators
P	Set of pipes
P^*	Set of pipes eligible for upsizing
E_p	Set of emitters whose hydraulic circuit contains pipe p
M_h	Set of admissible radiator options for emitter h (includes the option to keep existing model $m = \varnothing$)
D_p	Set of admissible inner diameters for pipe p, restricted to $d > d_{p,\text{base}}$
$q_{h,m}$	Required mass flow of emitter h under radiator option m
$q_{p,\text{base}}$	Baseline mass flow capacity of pipe p at its current inner diameter d_p^{base}
Δq_p	Maximal mass flow capacity gained by upsizing pipe p
ρ	Fluid density
$c_{h,m}$	Replacement cost of installing radiator model m at emitter h (0 for $m = \varnothing$)
c_p	Replacement cost of upsizing pipe p
$x_{h,m}$	Binary variable equal to 1 iff radiator option m is selected for emitter h
y_p	Binary variable equal to 1 iff pipe p is upsized

The objective function (27) represents the total cost incurred by the necessary replacements of radiators and pipes. Constraint (28) is defined for every pipe in the system, including both replaceable and non-replaceable pipes. The left-hand side of constraint (28) is the sum of flows of all selected radiator models whose hydraulic circuit includes the respective pipe. This value is constrained to not exceed the right-hand side, which represent the flow capacity of the pipe, including additional capacity gained due to upsizing. The constraint therefore enforces the maximal flow rates and therefore maximal water velocity in the pipes.

It is worth noting that the cost model distinguishes between radiator and pipe replacements. For radiators, a replacement cost can be specified individually for each selectable model. For pipes, a single replacement cost is assigned per pipe section, regardless of the selected diameter. This simplification is justified by the assumption that for pipe replacements in residential buildings, the dominant cost is the installation effort rather than the material cost. The exact replacement diameter therefore has limited influence on the cost model and is selected based on the required flow rate after the optimization has determined which pipe sections are to be replaced.

Constraint (29) ensures that exactly one admissible radiator model is selected for each radiator circuit. The set of admissible models includes the currently installed radiator, provided that it is capable of operating at the current iteration supply temperature.

If a feasible solution is found, the selected radiator models and pipes to upsize are given by variables $x_{h,m}$ and y_p respectively. A modified heating system graph is constructed accordingly, and a pipe network calculation is performed to validate the solution produced by the MILP solver. Subsequently, the algorithm attempts to further reduce the supply temperature further without any additional component replacements, until the system can no longer operate at the current iteration temperature. At that point, the MILP model is invoked again to identify further replacements. To prevent previously selected replacements from restricting subsequent solution spaces, each MILP model is constructed from the original, unmodified heating system graph. This ensures that the globally optimal replacement combination is identified independently at each iteration supply temperature.

4. Implementation

The implementation of the two modules comprising the library is informed by the design decisions presented in the previous section. While these decisions dictate the overall architecture, implementation-level decisions remain and are made in consideration of the non-functional requirements **N1**–**N5** defined in Section 3.2 and the conventions of the chosen programming language. These decisions concern aspects such as data structure realization, the handling of inter-entity references, the choice of numerical methods, and the integration of third-party dependencies. In cases where multiple valid implementation strategies exist, the chosen approach is motivated with respect to the relevant requirements and language conventions.

4.1. Language Considerations and Conventions

Swift [39] was chosen as the implementation language for the library, primarily due to its strong typing, value semantics, and native support for multiple target platforms such as iOS, macOS, and Linux. For maximum compatibility with existing codebases, the library targets Swift 5.9, while remaining compilable under Swift 6 to ensure forward compatibility.

Swift encourages the use of value types, realized through `struct` and `enum` definitions, over reference types realized through `class` definitions [39]. This approach is adopted throughout the library. Value types use copy-on-write semantics, eliminating shared mutable state and making all data transformations explicit [39]. A further benefit is thread safety by default through the implicit conformance to the `Sendable` protocol, as no two execution contexts can hold a reference to the same instance [39]. In the implementation, types not automatically conforming to the `Sendable` protocol, such as `enum`types, are explicitly declared as `Sendable` where possible. This helps to ensure maximal compatibility with structured concurrency and Swift 6. Following the value type conventions of Swift, shared behavior is expressed through conforming to a `protocol` rather than inheritance. Outside of type definitions, immutable `let` declarations are preferred over mutable `var` declarations wherever possible, in accordance with the general Swift convention of favoring immutability [39]. The adoption of these idiomatic Swift design guidelines satisfies requirement **N5**.

Furthermore, calculation functions are required to be non-mutating and idempotent, i.e., the calculation must not impact the state of the system. This ensures that individual calculations remain testable in isolation, satisfying requirement **N3**. To reduce the potential for errors, type-based constraints on input data are preferred over run-time validation. This is realized by defining an `enum` types for input parameters where the set of valid inputs is finite and enumerable. This makes invalid states unrepresentable at the compile level rather than requiring detection at runtime. In cases where input parameters must be validated at runtime, explicit and descriptive errors are thrown instead of silently returning `nil` or another sentinel value. This aids in debugging and allows frontends to convey the reason for the error to the user in a comprehensible manner.

The modules expose a comprehensive `public` API, satisfying non-functional requirement **N4**. Calculation functions are to be annotated with their source, i.e., the standard, page and equation or section number from which they originate, facilitating further development and validation by third parties. All input and output parameters must be annotated with their units. Should a calculation function require a large set of inputs, or produce a large set of output values, dedicated input and output structs are created for it, thereby improving readability. Together, these conventions contribute to the readability and maintainability goals of requirement **N2** and facilitate verification by third parties as required by **N3**.

Numeric calculations use the `Double` type to ensure full 64-bit floating point precision. For the heat load module, this is not a convention but an interpretation of the requirement stated in DIN/TS 12831-1 to use the full precision offered by the computation platform [23].

4.2. Heat Load Module

As a result of preferring compile-time over runtime validation, the implementation defines a total of 50 enumeration types based on tables specified in DIN/TS 12831-1 and DIN EN 12831-1. The implementation contains 73 equations from both standards, covering the standard procedure and both simplified procedures. Test coverage for all equations is ensured through a combination of reference calculations extracted from the evaluation sources and manually constructed unit tests.

4.2.1. Standard Procedure

For the standard procedure, the heat load module implements the four-level hierarchy of structs described in Section 3.4.1, consisting of `Building`, `VentilationZone`, `Room`, and `Component` types. DIN EN 12831-1 further defines the concept of building subunits, which are subject to the same calculations as a full building but use a different aggregation for their total heat load. Since the aggregation differs only in one additional term in the final summation, both buildings and building subunits are represented by the `Building` type, which provides a flag to indicate whether the instance represents a subunit.

The data stored at the component level depends on the component's adjacency type. Four types are defined, namely outside air, ground, heated adjacent room, and unheated room or a room of an adjacent building unit. While components of different adjacency types share common properties such as U-values and surface areas, each type additionally requires adjacency-specific parameters. Rather than defining a separate struct for each adjacency type, a single `Component struct` is introduced, augmented by an `enum` containing one case per adjacency type. Each case stores the parameters specific to that adjacency type, combining the shared and type-specific data within a single type.

Every room is assigned a unique string identifier, which is either provided by the user of the module or generated automatically as the string representation of a Type 4 universally unique identifier (UUID). The building instance validates the uniqueness of all room identifiers at runtime. These identifiers serve two purposes: First, they allow rooms to be referenced from outside the module, enabling the heat loads to be associated with corresponding heat emitters in the hydronic balancing module. Second, they help to resolve the design challenge discussed in Section 3.4.1 of introducing references between rooms, by allowing rooms and components to refer to other rooms by their id. Prior to the calculation, a map from room identifier to temperature is constructed for both the standard and comfort cases, which rooms and components use to resolve the temperatures of adjacent rooms. If a room or component references a non-existing room id, an error is thrown describing the bad reference and where it occurred.

An alternative approach to resolving inter-room references is to represent rooms as reference types using Swift classes, rather than value types. Swift provides reference types for cases where shared references are required, allowing components and rooms to hold direct references to other rooms, with correctness enforced at compile-time rather than at runtime. However, this approach introduces the complication that rooms that reference each other form reference cycles, which prevent the automatic reference counting (ARC)

memory management of Swift from deallocating the involved objects, thereby causing memory leaks. The standard mitigation for this in Swift is the use of `weak` or `unowned` references, which are permitted to become invalid at runtime. This reintroduces the possibility of invalid references that must be handled at runtime, negating the primary advantage of using reference types in this context. Furthermore, adopting reference types would forfeit the thread safety guarantees and other benefits associated with value types in Swift, such as automatically synthesised equality and serialization. For these reasons, value types are retained and the identifier-based resolution approach is adopted.

Many of the equations implemented in the standard procedure are realized as static functions that take all required inputs, including intermediate results, as explicit parameters. This design decision is motivated by efficiency considerations. If the corresponding methods were instance methods, multiple methods relying on the same intermediate value would each independently trigger its computation, causing repeated evaluation of the same calculation. Especially for intermediate calculations that require iteration over collections, such as a building aggregating the surface areas of all its components, repeated calculation can be computationally expensive. Therefore, static functions are employed for calculations requiring potentially expensive intermediate values.

4.2.2. Simplified Procedures

For the simplified procedures, dedicated simplified versions of the data structures are introduced. Both simplified procedures share a common simplified component `struct`. The building-level simplified procedure uses a simplified building `struct` containing a list of components directly, while the room-level procedure introduces a simplified room struct containing components. Although simplified rooms are standalone, a building `struct` is introduced for the simplified room-level procedure as a convenience container that is not part of the computation itself.

As neither rooms nor components in the simplified procedures require references to other rooms, the identifier resolution mechanism described above is not required. Room identifiers are nonetheless used in the simplified procedures to facilitate the association of rooms with heat emitters in the hydronic balancing module. Furthermore, due to both procedures being computationally inexpensive, all calculations are implemented as instance methods, as this is considered to improve readability. The considerations of recalculating intermediate values multiple times do not apply to the simplified procedures, as no such intermediate values exist.

4.2.3. Limitations

While the implementations of both the simplified and standard procedures support the majority of calculations specified in DIN/TS 12831-1 and DIN EN 12831-1, some limitations apply. The procedure for estimating the building heat load based on heat metering or consumption data defined in DIN/TS 12831-1 is not implemented by the module.

For the standard procedure, DIN/TS 12831-1 additionally provides a method for estimating volume flows through large openings in the building envelope, which is likewise not implemented. This is unlikely to be relevant in practice, as such openings are not typical, especially in residential buildings [23]. Nevertheless, the module allows for externally calculated values for such volume flows to be specified as input parameters, so that buildings requiring this consideration can still be calculated without modifications to the module.

Furthermore, in the standard procedure, two models for the calculation of ventilation losses are specified, a general calculation model and a simplified one. The simplified model is derived from the general model, but does not include terms related to mechanical ventilation, ATDs, and air change rates. Therefore, the simplified model may only be applied for buildings that are sufficiently air tight, and have no mechanical ventilation system or ATDs [21]. The module does not implement the simplified model as a separate calculation. However, when omitting the parameters not applied in the simplified model, the general model becomes equivalent to the simplified model. A separate implementation of the simplified model is therefore redundant.

4.3. Hydronic Balancing Module

The hydronic balancing module implements the heating system graph, pipe network calculation, static and dynamic hydronic balancing, and supply temperature optimization procedures described in Section 3.5. The following describes the implementation-level decisions made for each of these components, with particular attention to aspects where Swift-specific considerations influenced the chosen approach.

4.3.1. Heating System Graph

In line with the preference for value types, the heating system graph is realized by using identifiers for system elements, and dictionaries to store both the elements themselves and the connections between their ports. More specifically, two `struct` types are defined to represent the heating system graph. The first is a mutable version used during graph construction. The second is an optimized, immutable version derived from the mutable graph and used for calculations.

The mutable version contains two dictionaries, one storing system elements with their id as a key, and the other containing all pipes of the system, with their key being the undirected connection between two ports connected by the pipe. These undirected connections are implemented such that a connection from port A to port B yields the same hash value as a connection from B to A, with both versions considered to be equal. This undirected approach reflects reality as pipes are inherently unidirectional before a flow is established and further prevents two pipes being added to the same set of connections but in opposite directions. The mutable version acts as a builder for the unmutable version and provides multiple functions to add or modify system elements and pipes.

When the network definition is complete, the mutable graph is converted into its immutable counterpart. A consistency check ensuring that the graph is well formed is performed as part of the conversion process. In addition to the two dictionaries of the mutable version, the immutable version contains a third dictionary mapping ports to connections. This allows for faster access times during graph operations such as breadth-first search (BFS), as without this dictionary, finding the connection of a port would require a linear time search through the keys of the pipe dictionary.

Alternatively, the immutable graph could also be realized by using reference type nodes, and references within those nodes to represent connections. This approach would preserve the thread-safety guarantees of Swift's structured concurrency by defining each node and the containing graph type as a `final class` with exclusively immutable member variables. In this case, circular references would need to be addressed using `unowned` references, as `weak` references imply mutability and are therefore incompatible with `Sendable` conformance. While accessing a invalid `unowned` reference causes a crash, such references could still be used safely in this case, as the containing class representing the graph would ensure that a reference is kept to all nodes, preventing them from being deallocated. This approach would offer faster access times of $\mathcal{O}(1)$ for connections between elements, compared to the $\mathcal{O}(n \log n)$ access time incurred by dictionary-based lookups. However, given that heating system graphs are expected to contain only hundreds of elements, the expected performance benefit is modest, and the additional implementation complexity can be considered disproportionate to the gain. Therefore, the identifier-based approach using value types was chosen.

4.3.2. Radiator Mass Flow Rates

The first step in performing the pipe network calculation is determining the flow rate through each pipe. The flow rates through the pipes are governed by the mass flow rates required by each emitter, which for radiators are calculated by the module. As described in Section 2.2.2, two conflicting excess temperature definitions exist, namely the arithmetic definition of DIN EN 442-2, adopted by E DIN 94679-1, and the logarithmic definition of DIN 4703-3. Although largely superseded by DIN EN 442-2, DIN 4703-3 retains authority over the excess temperature definition and explicitly mandates the logarithmic definition for calculations at supply temperatures deviating from design conditions, which is the primary use case of the module.

To address this, the module supports both definitions, adopting the logarithmic definition of DIN 4703-3 as the default. This choice reflects the principle of prioritising national specifications over European ones, as DIN EN 442-2 represents an adoption of a European standard, whereas DIN 4703-3 is a national standard that explicitly overrides it [17, 20]. Although the logarithmic approach requires iterative solving and is therefore computationally more expensive than the arithmetic alternative, the number of radiators in a typical heating system is sufficiently small that this overhead remains negligible, making the logarithmic definition a sensible default.

4.3.3. Pipe Friction

A prerequisite to performing a pipe network calculation is the ability to determine the pressure losses caused by individual pipes and system elements. For system elements, pressure losses are calculated unambiguously from either the resistance coefficient ζ or the flow coefficient K_v. The dimensionless resistance coefficient ζ is typically used to characterize the pressure losses caused by fittings such as tee pieces and elbow connectors. It is either specified by the manufacturer of the component or estimated from standard values for common component types available in engineering references [1, 26]. The flow coefficient K_v characterizes the pressure losses caused by valves, with manufacturers typically supplying a set of K_v values corresponding to the adjustment setting of the valve [1, 15, 26].

For pipes, the pressure loss depends not only on water velocity, but also on pipe diameter, length, and surface roughness. As discussed in Section 2.2.5, multiple correlations exist for estimating the pipe friction factor λ [1, 13, 32, 34, 38]. In this module, λ is calculated through a combination of approaches drawn from applicable engineering references [1, 38], based most notably on the publications of Moody [32] and Nikuradse [34]. As dictated by these references, the Hagen-Poiseuille law is applied for laminar flow [1, 32, 34, 38]:

$$\lambda = \frac{64}{Re} \tag{34}$$

For turbulent flow in hydraulically smooth pipes, the implicit equation of Prandtl is used [32, 35, 38]:

$$\frac{1}{\sqrt{\lambda}} = -2 \cdot \log_{10}\left(\frac{2.51}{Re \cdot \sqrt{\lambda}}\right) \tag{35}$$

For hydraulically rough pipes, the equation of Nikuradse is employed [32, 34]:

$$\frac{1}{\sqrt{\lambda}} = -2 \cdot \log_{10}\left(\frac{k}{3.71 \cdot d_i}\right) \tag{36}$$

Finally, for turbulent flow in the transitional roughness region, λ is determined using the Colebrook-White equation [13, 32, 38]:

$$\frac{1}{\sqrt{\lambda}} = -2 \cdot \log_{10}\left(\frac{k}{3.71 \cdot d_i} + \frac{2.51}{Re \cdot \sqrt{\lambda}}\right) \tag{37}$$

where:

λ	Darcy friction factor	
Re	Reynolds number	
k	Surface roughness of the pipe	m
d_i	Inner diameter of the pipe	m

The laminar regime is defined in this implementation as $Re < 2320$, deviating from the threshold of $Re < 2000$ used by Moody. Furthermore, pipe friction factors for the transitional region between laminar and turbulent are estimated as turbulent by the implementation. These choices are consistent with applicable engineering references by the Association of German Engineers (VDI) [38].

While the presented correlations for the turbulent regime are well established, a normative specification of the conditions under which each should be applied does not appear to exist within the context of hydronic engineering [1, 3, 38]. The paper by Nikuradse, by contrast, provides explicit definitions for both boundaries [34], and these criteria are therefore adopted in the module. These boundaries are expressed in terms of the roughness Reynolds number η, defined as [32, 34]:

$$\eta = \frac{v_* \cdot k}{\nu} \tag{38}$$

where v_* denotes the friction velocity [34]:

$$v_* = \sqrt{\frac{\tau_0}{\rho}} \tag{39}$$

with [34]:

$$\frac{\tau_0}{\rho} = \lambda \cdot \frac{v^2}{8} \tag{40}$$

where:

η	Roughness Reynolds number, determination factor of Nikuradse	
v_*	Friction velocity	m/s
v	Average flow velocity	m/s
ν	Kinematic viscosity	m^2/s
τ_0	Wall shear stress	Pa
ρ	Fluid density	kg/m^3

Rearranging the definition of the Reynolds number (see Section 2.2.5, Equation (8)) yields a kinematic viscosity of $\nu = v \cdot d_i/Re$, which allows η to be expressed entirely in terms of quantities available during the calculation:

$$\eta = \frac{\sqrt{\lambda \cdot \frac{v^2}{8}} \cdot k}{\nu} = \frac{\sqrt{\lambda \cdot \frac{v^2}{8}} \cdot k}{(v \cdot d_i)/Re} = \frac{\sqrt{\lambda/8} \cdot k \cdot Re}{d_i} \tag{41}$$

Since k and d_i are stored as pipe parameters and Re is determined during the pipe network calculation, η can be obtained directly from known quantities. The applicable friction correlation is then selected based on the value of $\log(\eta)$ according to the criteria of Nikuradse [34], summarised in Table 2.

Table 2: Selection criteria for pipe friction correlations based on the roughness Reynolds number η, following Nikuradse, Moody and Colebrook [13, 32, 34].

Condition	**Used correlation**
$\log_{10}(\eta) < 0.55$	Prandtl (hydraulically smooth)
$0.55 \leq \log_{10}(\eta) \leq 1.83$	Colebrook-White (transitional roughness)
$\log_{10}(\eta) > 1.83$	Nikuradse (hydraulically rough)

The friction correlations for the turbulent regime, namely the Prandtl, Nikuradse, and Colebrook-White equations, are implicit in λ and therefore require iterative solving. Furthermore, since the selection of the applicable turbulent correlation depends on η, which itself depends on λ, the correlation cannot be chosen definitively prior to solving. An outer iteration loop is therefore introduced that allows the applicable correlation to be re-evaluated after each update of λ, ensuring that the final solution is consistent with both the converged friction factor and the correct correlation selection. Iterative solving was found to converge within fewer than ten iterations to a precision of 10^{-10}. As iterative solving promises more accurate results than explicit approximations such as that of Blasius [1], and the associated computational overhead is considered acceptable, iterative solving is implemented.

4.3.4. Pipe Network Calculation

The pipe network calculation begins by traversing the hydraulic network twice using BFS. The graph is traversed once starting from the inlet of the heat generator and once from its outlet, thereby identifying the supply and return paths for each emitter and establishing all emitter circuits. As the traversal is performed on the validated, immutable graph, the existence of a unique supply and return path per emitter is guaranteed, rendering the shortest-path guarantee of BFS irrelevant. BFS is chosen over depth-first search (DFS) solely because Swift's standard library does not provide an efficient stack implementation.

The flow rate through each pipe is determined as the sum of the flow rates of all emitter circuits passing through it. The water temperature within each pipe is determined separately for the supply and return networks. Supply pipe temperatures are equal to the system supply water temperature. In alignment with E DIN 94679-1 [26] and VDI 2073 Part 2 [41], heat losses along the pipes are not modeled by the implementation. Return pipe temperatures are calculated as the flow-weighted average of the return temperatures of all emitter circuits passing through a specific pipe, since each emitter returns water at an individually determined temperature. This simplified formulation of the return water mixing temperature is applicable because the specific heat capacity of the mixing water can be assumed to be approximately equal [26, 27].

From the pipe flow rates and temperatures, flow metrics including water velocity and pressure loss per meter are calculated as described in Sections 2.2.5 and 4.3.3. Subsequently the pressure losses caused by the system elements are determined. The

total pressure loss of an emitter circuit is obtained by summing the pressure losses of all pipes and system elements along the circuit path, including those of the emitter and the heat generator. Branch balancing valves, TRVs, and return line valves are deliberately excluded from this summation, as their pressure losses are adjustable and their determination is the objective of the subsequent balancing process (see Section 3.5.3).

4.3.5. MILP Solver

The formulation of the static hydronic balancing problem as a MILP requires a suitable solver library available on the target platforms iOS, macOS, and Linux. No publicly available MILP library with an API to current Swift versions could be identified as of the time of writing. To address this, the open-source solver `lp_solve` [7], which is written in C, is compiled and integrated into the module via Swift to C interoperability.

The integration is structured into four layers. At the lowest level, the compiled `lp_solve` C library provides the underlying solver functionality. Directly above, the `Shim` layer uses Swift-C interoperability to exposes the C interface to Swift. The third layer, `LPSolver`, adds a matrix-based Swift interface to the solver, allowing MILP problems to be constructed and solved by directly manipulating the constraint matrix. This layer leverages Swift's ARC to abstract away the manual memory management required by the underlying C library.

The fourth and uppermost layer, `LPModel`, provides an idiomatic Swift abstraction over `LPSolver`, allowing MILP problems to be formulated in terms of named variables and constraints without requiring direct manipulation of the underlying matrix representation. This layer eliminates the risk of off-by-one errors and similar index-related mistakes to which direct manipulation of the constraint matrix in the third layer is prone. All types defined at this layer are value types that conform to the `Sendable` protocol, making model definitions inherently compatible with structured concurrency (i.e., thread-safe) and enabling automatic synthesis of serialization and deserialization functionality. An example `LPModel` definition is shown in Figure 7.

```
var model = LPModel()

let x1 = model.addVariable("x1", kind: .integer, lowerBound: 0)
let x2 = model.addVariable("x2", kind: .continuous, lowerBound: 0)

model.objective = 13*x1 + 12*x2 + 69

model.addConstraints {
    2*x1 + x2 <= 42
    7*x1 + 3*x2 <= 161
}

let solution = try model.solve()
print(solution.objective, solution.value(of: x1))
```

Figure 7: A simple example MILP formulation using the high-level `LPModel` API.

The hydronic balancing module interacts exclusively with the `LPModel` layer due to its safety guarantees, and the MILP models described in Sections 3.5.3 and 3.5.4 are realized through this interface. Although the static hydronic balancing problem is formulated as a single lexicographic optimization in Section 3.5.3, it is implemented by solving the three objective levels sequentially, as `lp_solve` [7] does not support lexicographic objectives natively. After each stage, the optimal value of the corresponding objective is fixed as a constant and inherited by the subsequent stage, producing a result equivalent to that of a native lexicographic optimisation.

5. Evaluation

To verify the correctness of the implementation, both modules are evaluated against reference and example calculations drawn from the referenced standards and established comparable software. As the heat load module follows a normative standard prescribing the calculation procedure in detail, its results must match the reference values exactly, barring rounding imprecision. For the hydronic balancing module, greater latitude exists in places where the applicable technical rules and standards do not specify exact procedures, necessitating implementation-level choices. This can introduce deviations from the reference calculations in the results which are reflections of different implementation choices rather than calculation errors. The evaluation is performed separately for the heat load and hydronic balancing modules.

5.1. Heat Load Module

The heat load module is evaluated using three independent reference sources, covering transmission losses, ventilation losses, and additional heating-up loads. The reference sources were selected to represent a range of perspectives, including normative examples published by the DIN, output from certified commercial software, and calculations performed by a professional service.

5.1.1. Methodology

The evaluation of the heat load module is carried out by comparing its calculation results against three independent reference calculations from different sources. The objective of this comparison is to verify the correctness of the implemented equations and to assess the consistency of the module with established calculation procedures.

For each reference case, the set of input parameters provided by the reference are converted into the data structures described in Section 3.4.1. These parameters include geometric building data, thermal properties of building components, ventilation system design, indoor design temperatures, and the applicable climate conditions. Where necessary, assumptions are made in order to resolve ambiguities or inconsistencies in the published examples. Such assumptions are explicitly documented to ensure transparency and reproducibility.

The results provided by the heat load module for transmission and ventilation losses, and any additional heat loads, are compared against the corresponding values reported in the reference calculations. In cases of discrepancies, all available intermediate values from the reference calculations are compared with the module results to determine the cause of the deviation. Comparisons are performed at a least at room level, but also at the zone, component, and building level, where applicable.

Since the heat load values provided by the reference calculations are generally rounded to integer values, whereas the module outputs unrounded values, a value is only considered to deviate if the absolute difference is greater than 0.5 W. Differences below this threshold are attributed to rounding rather than computational errors of the module. However, this deviation threshold does not include cumulative rounding effects that may arise when reference sources also round intermediate calculation steps. As a consequence, deviations exceeding the 0.5 W threshold could still be attributable to compounded rounding. Such cases are identified and discussed explicitly in the evaluation of the respective reference case.

5.1.2. Reference Data

DIN/TS 12831-1:2020-04 In annex B of DIN/TS 12831-1:2020-04, two example calculations using the standard procedure including intermediate results are provided. As the calculations are part of the standard document, and are therefore published by the same institution that published the standard itself, they would be expected to provide reliable values.

However, the example calculations were found to contain multiple inconsistencies. An example of such a discrepancy is the envelope area of the buildings single ventilation zone, $A_{env,z}$, which is assumed to be 416 m^2 on page 132, and 442 m^2 on page 138. Since both values for the envelope area are used in the calculation of early intermediate results, this error is propagated into virtually all ventilation system losses, making them unreliable. Due to the extent of the found inconsistencies, the examples were deemed too unreliable to use for verification and therefore not utilized.

E DIN SPEC 12831-1:2018-10 Before publishing DIN/TS 12831-1:2020-04, the DIN first released a draft version of that standard labeled E DIN SPEC 12831-1:2018-10 [22], which also includes an informative annex B containing two example calculations. After manual comparison of both documents, it was determined that all equations, constants and tables relevant to the example calculations remain the same between both documents. Therefore, the results published in draft E DIN SPEC 12831-1 were deemed likely to be suitable for verification of the implementation.

Both example calculations reference the same house as the later released DIN/TS 12831-1 standard. The first example assumes no mechanical ventilation, so only infiltration through the building envelope and minimal air change rates contribute to ventilation losses. The second example assumes a full ventilation system with heat recovery and supply, exhaust, and transfer air streams into, out of, and in between rooms respectively.

Consequently, the input parameters for the first example are equal to the those of the second example, when setting the supply, exhaust, and transfer air volume rates to zero.

For 4 of the 16 rooms of the example building, detailed room- and component-level input data and resulting transmission losses are specified, including intermediate results. For the 12 remaining rooms, not all required parameters could be determined unambiguously, making them unsuitable for transmission calculation verification. While ventilation input parameters and reference losses are available for all rooms, the lack of transmission input parameters for some rooms prevents comparison at the building level. Since the inconsistency issues found in DIN/TS 12831-1 were not found in the examples of E DIN SPEC 12831-1, the examples are considered suitable for verification.

EVEBI EVEBI [28] is a certified energy consulting software manufactured by ENVISYS GmbH & Co. KG. According to the manufacturer, the software is able to accurately calculate the heat load of residential buildings in accordance with DIN/TS 12831-1.

The software generates the results of the heat load calculation in the form of a portable document format (PDF) document, following the format of normative templates presented in DIN/TS 12831-1. As per specification of the standard, this document includes all parameters required to perform the calculation, and includes intermediate results.

The software (Version 13.10.5) ships with an example building project, containing 1 ventilation zone, 10 rooms, and 111 components, and a specification for a mechanical ventilation system. From the output document for the heat load calculation for this building, all required parameters and reference results are extracted and used to create a verification test case. As the software is certified, it can be assumed that the results produced by the software are reliable and are therefore suitable for verification.

wbs The company wbs [43] offers calculating heat loads in accordance with DIN/TS 12831-1 as a service. On the company website, wbs offers a sample calculation in the form of a PDF document. As the EVEBI output document, the document provided by wbs largely complies with the format standardized by DIN/TS 12831-1. Consequently, all parameters and results required for verification are extracted from the document.

Due to wbs using this example calculation as a reference for the services they offer, the presented results are assumed to be valid and suitable for verification. While no numerical errors comparable in severity to those in DIN/TS 12831-1 were identified, one structural inconsistency was nevertheless found. Internal walls separating rooms of different temperatures must be specified as components, whereas walls between rooms of equal temperature can be omitted. The wbs document lists an internal wall for the bathroom, which is heated to a higher temperature than all other rooms in the building. Since such a wall must border an adjacent room, at least one other room should list a corresponding internal wall component. However, no such component is specified for any other room in the document, making the building definition structurally inconsistent. Nevertheless, the inconsistency affects only the completeness of the building definition rather than the correctness of the calculations performed, and the document can be considered suitable for verification purposes.

5.1.3. Standard Procedure Results

E DIN SPEC 12831-1:2018-10 The two example calculations included in E DIN SPEC 12831-1 differ only in the presence of a mechanical ventilation system. As transmission losses are not influenced by the presence of a ventilation system, the transmission values are equal for both examples, such that a single transmission test case is sufficient to cover both scenarios.

Of the 16 total rooms, only 4 are provided with the parameters required for transmission loss calculations. Figure 8 compares the transmission losses calculated by the implementation to the respective reference values for these 4 rooms and their combined 25 components. As all deviations lie below the threshold of 0.5 W, the transmission losses calculated are considered successfully verified against the reference.

For the test case without a mechanical ventilation, some ventilation loss figures deviate more than the threshold, with the highest deviation amounting to 0.73 W. Closer investigation reveals that the reference values provided in the example are calculated using rounded values for the related volumetric air flow rates, causing cumulative rounding errors. The calculations performed by the module are therefore still considered to be accurate to the reference.

Figure 9 shows that, in the test case with a ventilation system, five rooms exhibit substantial deviations in total ventilation heat loss relative to the reference values. The largest deviation is approximately 231 W, with the implementation yielding 152 W compared to 338 W in the reference. A deviation of this magnitude would have a

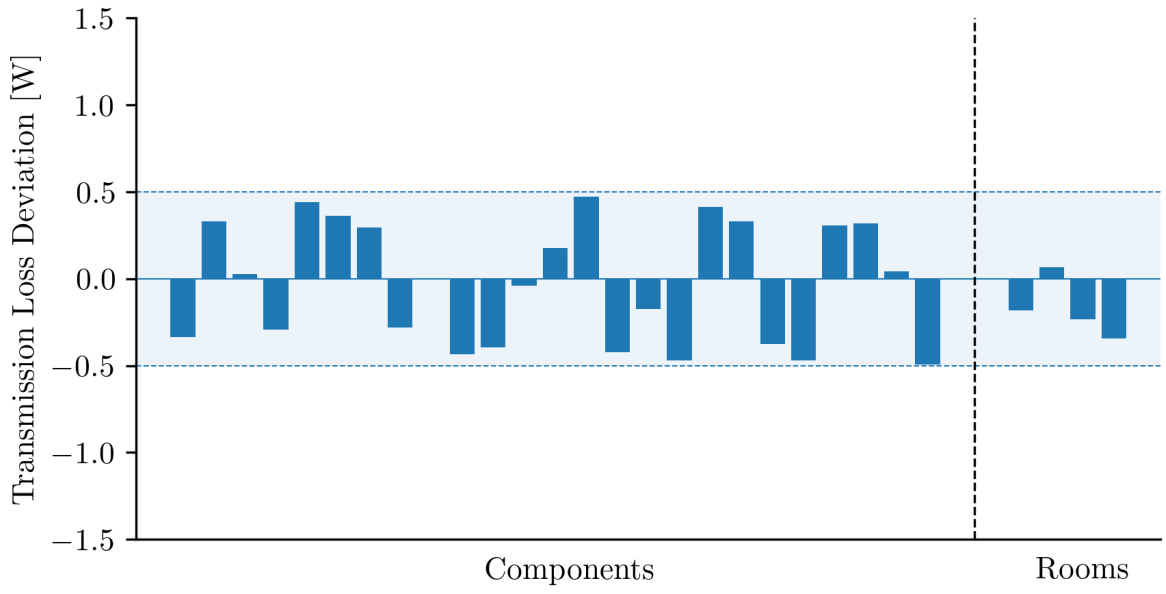

Figure 8: Bar plot illustrating the deviations between the transmission losses calculated by the module and the reference values from the example calculations provided in E DIN SPEC 12831-1 [22]. Results are presented for all 25 components and 4 rooms. Deviations below a magnitude of 0.5 W are treated as negligible.

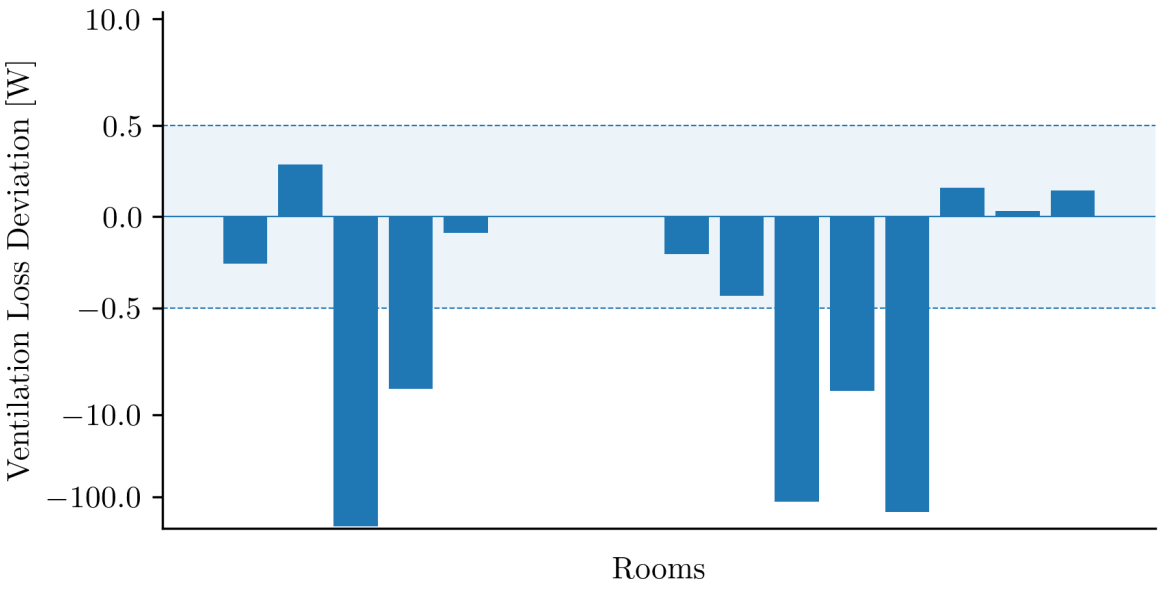

Figure 9: Bar plot illustrating the deviations between the ventilation losses calculated by the module and the reference values from the second example calculation provided in E DIN SPEC 12831-1 [22]. Results are presented for all 16 rooms using a symmetric logarithmic (symlog) scale with a linear range between −0.5 and 0.5 W. Deviations below a magnitude of 0.5 W are treated as negligible. Five of the sixteen rooms exhibit deviations beyond this range, the largest amounting to approximately 231 W.

considerable influence on heating system dimensioning and is therefore of practical relevance. Further analysis determined that these deviations can be traced back to deviations in the computation of losses caused by supply air streams, which are propagated into the total ventilation loss. Both DIN/TS 12831-1 and E DIN SPEC 12831-1 define the supply ventilation loss using the following equation:

$$\Phi_{V,sup,i} = \rho \cdot c_p \cdot q_{v,sup,i} \cdot (\theta^*_{int,i} - \theta_{rec,z}) \tag{42}$$

where:

$\Phi_{V,sup,i}$	Ventilation heat loss caused by supply air	W
ρ	Density of the air	$\mathrm{kg/m^3}$
c_p	Specific heat capacity of the air	$\mathrm{Wh/(kg \cdot K)}$
q_v	Supply air volumetric flow rate	$\mathrm{m^3/h}$
$\theta^*_{int,i}$	Average air temperature of the room	°C
$\theta_{rec,z}$	Temperature of the supply air volume flow	°C

For each parameter of Equation 42, no deviation beyond the respective rounding precision is observed between the computed and reference values. Consequently, the discrepancy could not be attributed to differences in the input parameters and instead appeared to originate from the application of Equation 42.

A manual recalculation of $\Phi_{V,sup,i}$ using the parameters provided in the reference yielded different results than those stated therein. Notably, this inconsistency occurs precisely for those rooms in which deviations in the total ventilation losses were observed. In fact, the results of the manual recalculations closely match the results computed by the heat load module. This suggests that the source of the discrepancy lies in the reference values themselves rather than in a computational error of the module. Accordingly, the identified deviations are not interpreted as evidence of an implementation error.

In summary, the comparison with the example calculations of E DIN SPEC 12831-1 indicates that the implementation behaves correctly. The deviations observed in the ventilation loss results of the second test case are attributable to inconsistencies in the reference values rather than to errors in the implementation.

EVEBI For the EVEBI test case, 4 of the 111 components exhibit transmission loss deviations above the 0.5 W threshold. These deviations can likely be attributed to limited input data precision. The surface area of a component is a linear factor in the equation for transmission loss, and is rounded to one decimal place in the output document. This rounded value is subsequently used as an input for the verification calculation. However, it is, likely that EVEBI performs its internal calculations using the full, unrounded surface area. Consequently, if a surface area A_k is specified in the output document, the effective surface area used internally by EVEBI must lie within the interval $[A_k - 0.05; A_k + 0.05)$. To assess the impact of this uncertainty, the transmission losses for the affected components were recalculated using both the lower and upper bounds of this interval. The resulting range of transmission losses was found to include the reference values reported by EVEBI in all cases of deviation. This suggests that the observed deviations may be attributable to rounding effects rather than implementation errors. A detailed representation of the difference between calculated and reference transmission losses for all 111 components and 10 rooms is shown in Figure 13 in Appendix A.

In addition to the component-level deviations, five rooms exhibit differences exceeding the defined threshold, with the largest deviation amounting to approximately 1 W. Since the room-level transmission loss is the sum of its component-level contributions, the uncertainty in room-level results can be bounded by accumulating the component-level uncertainty intervals. For each of the five affected rooms, the reference value falls within the resulting interval. Consequently, these discrepancies are not interpreted as indications of errors in the implementation, but as results of cumulative rounding errors.

The evaluation of the ventilation losses revealed an inconsistency in the data output generated by EVEBI. To calculate ventilation losses caused by infiltration through the building envelope, the envelope area of each room is defined in Equation (43).

$$A_{env,i} = \sum_{k \in K_{H,i}} A_k \tag{43}$$

where:

$A_{env,i}$	Envelope surface area of room i	m^2
A_k	Surface area of component k	m^2
$K_{H,i}$	Set of components considered to be part of the rooms envelope	

Furthermore, DIN/TS 12831-1 allows the surface area of a component to be specified using a decomposition into gross area and deducted area:

$$A_k = A_{\text{gross},k} - A_{\text{deduct},k} \tag{44}$$

where:

A_k	Surface area (i.e., net area) of component k	m^2
$A_{\text{gross},k}$	gross surface area of component k, potentially including embedded components (e.g., windows, doors).	m^2
$A_{\text{deduct},k}$	Cumulated surface area of embedded components included in $A_{\text{gross},k}$.	m^2

This method of specifying component areas enables a transparent declaration of excluded sub-areas. An example application for this is a rectangular wall containing a window. This configuration is represented by two components, one for the wall and one for the window. The window is defined with $A_{\text{gross,window}}$ equal to its surface area and $A_{\text{deduct,window}} = 0$. The wall component is defined with $A_{\text{gross,wall}}$ corresponding to the product of its width and height, with $A_{\text{deduct,wall}} = A_{\text{window}}$.

Both EVEBI and the heat load module support this representation of component areas. However, inspection of the EVEBI output files indicates that, when determining the envelope areas of the rooms in the example, EVEBI appears to not sum up the net component areas A_k, but instead aggregates the gross areas $A_{\text{gross},k}$. As a consequence, embedded components such as windows and doors are effectively counted twice in the envelope area, leading to overstated envelope areas and increased ventilation losses.

Consequently, for some rooms, the ventilation losses calculated by the heat load module deviate by up to 15 W. In order to investigate whether the aforementioned issue is the sole cause for these deviations, the input parameters used by the heat load module to calculate the ventilation losses were adjusted: For every component, $A_{\text{deduct},k}$ was set to 0. Thereby, the areas of embedded components are no longer deducted from the gross area of their enveloping component, effectively replicating the error made by EVEBI. With the adjusted input parameters, the deviations in ventilation loss were reduced to be within the threshold, as illustrated by Figure 14 in Appendix A. This corroborates the theory that the deviations are solely caused by erroneous envelope areas in the reference data. Consequently, the deviations in ventilation data are not interpreted as evidence for implementation errors in the heat load module.

Furthermore, two rooms in the EVEBI example have additional heating-up loads caused by intermittent heating defined (see Section 2.1.5). The magnitude of the increases in heat load caused by these additions is within the threshold for both rooms.

Overall, the verification against the reference data extracted from EVEBI is considered successful. On a building level, even though EVEBI makes an error in calculating part of the ventilation losses, the overall heat load only deviates by 12 W, with the heat load module yielding 4730 W compared to 4742 W calculated by EVEBI.

wbs For the wbs test case, 5 of the 62 included components exhibit transmission loss deviations exceeding the 0.5 W threshold, with the largest amounting to 0.65 W. At room level, two of the twelve rooms exceed the threshold, with a deviation of 0.77 W. As with the EVEBI test case, the surface areas of components are reported to one decimal place in the wbs output document, introducing an input precision uncertainty of up to 0.05 m^2 per component. For the affected components, recalculation using the bounds of this uncertainty interval shows that the reference values fall within the resulting range. The room-level deviation can likewise be bounded by accumulating the component-level uncertainty intervals with the reference value falling within the resulting interval. Consequently, none of the observed deviations are interpreted as evidence of implementation errors. Figure 15 in Appendix A shows the transmission loss deviations at component and room level for the wbs test case.

Figure 16 in Appendix A shows the ventilation loss deviations at room level for the wbs test case. Of the twelve ventilation loss values compared, two exceed the 0.5 W threshold, with deviations of approximately 0.51 W and 0.52 W respectively. Both deviations could be traced back to the calculated envelope surface area of the affected rooms, which enters the infiltration-based ventilation loss calculation as a linear factor. Applying the previously used bounding approach on component areas, the reference values for both rooms fall within the uncertainty interval resulting from the limited surface area precision. Consequently, these deviations are not interpreted as evidence of implementation errors.

However, one intermediate result of the ventilation loss calculation, namely the technical air flow rate $q_{v,techn,i}$, which is defined for each room, was found to significantly deviate between the module and reference. This air volume flow is defined by DIN EN 12831-1 with the following equation:

$$q_{v,techn,i} = max(q_{v,sup,i} + q_{v,transfer,ij}\ ,\ q_{v,exh,i} + q_{v,comb,i}) \tag{45}$$

where:

$q_{v,techn,i}$	Technical volumetric air flow rate of room i	m^3/h
$q_{v,sup,i}$	Supply volumetric air flow rate into room i	m^3/h
$q_{v,transfer,ij}$	Transfer volumetric air flow rate into room i from room j	m^3/h
$q_{v,exh,i}$	Exhaust volumetric air flow rate from room i	m^3/h
$q_{v,comb,i}$	Combustion, or any other technically required, volumetric air flow rate out of room i	m^3/h

The values of $q_{v,techn,i}$ were found to differ between the module and the reference for multiple rooms. Manual recalculation of the $q_{v,techn,i}$ value of each room shows that the module applies Equation (45) correctly, and the discrepancy is attributable to an error in the reference document. For example, the document defines one room with $q_{v,sup,i} = 40\,\mathrm{m}^3/\mathrm{h}$, $q_{v,transfer,ij} = 40\,\mathrm{m}^3/\mathrm{h}$, $q_{v,exh,i} = 0\,\mathrm{m}^3/\mathrm{h}$, and $q_{v,comb,i} = 0\,\mathrm{m}^3/\mathrm{h}$. Applying Equation (45) to these values yields $q_{v,techn,i} = 80\,\mathrm{m}^3/\mathrm{h}$, however, $q_{v,techn,i} = 40\,\mathrm{m}^3/\mathrm{h}$ is listed in the wbs document. This discrepancy does not influence the final heat loads, as the technical air flow rate only affects ventilation losses under conditions not met by any room in the reference building.

As there are no errors found in the document published by wbs that would affect transmission or ventilation losses, this test case also allows for comparisons on the building level. The heat load for the entire building is reported as 5696 W in the reference document, and computed as 5695.6 W by the module, well within the threshold of rounding precision. In summary, all results produced by the heat load module for the wbs test case are consistent with the reference values, with all observed deviations attributable to the limited precision of the input data reported in the reference document.

5.1.4. Simplified Building Procedure Results

To verify the implementation of the simplified building-level calculation procedure, the EVEBI test case was used, as it is the only reference source among those listed that also provides results for the simplified building-level procedure. The input parameters already extracted for the standard procedure were translated into the data structure defined for the simplified building-level procedure, additionally enabling direct comparison between the results of both procedures for the same building. For the EVEBI building, the time constant is considered in the transmission loss calculation under the standard procedure (see Section 2.1.4). In accordance with DIN/TS 12831-1, which states that the influence of the time constant is generally insignificant for the simplified procedures, it was omitted accordingly. Using the simplified procedure, EVEBI reported a heat load of 5103 W for the example building, while the implementation yielded 5100 W. The deviation of approximately 0.06% falls within the uncertainty range attributable to the limited surface area precision discussed above and is therefore considered practically insignificant.

Compared to the implementation of the standard procedure, which yielded a heat load of 4730 W for the same building, the simplified procedure overestimated the heat load by approximately 8%. To investigate whether this discrepancy is consistent across test cases, the wbs reference building was adapted for applicability of the simplified procedure. Since the simplified procedure does not permit mechanical ventilation systems, the ventilation system was removed and replaced with standard minimal air exchange rates for all affected rooms, reflecting the typical specification of buildings without ventilation systems [23]. Calculating the heat load for the modified wbs reference building, yielded 6181 W using the simplified and 6156 W using the standard procedure, constituting a substantially smaller overestimation of approximately 0.4%.

The difference in accuracy of the simplified procedure between the two test cases is likely attributable to the handling of the time constant. For the wbs building, the time

constant is omitted in both procedures, whereas for the EVEBI building, it is considered in the standard procedure but not in the simplified procedure, introducing an asymmetry. Recalculating the heat load for the EVEBI building using the simplified procedure, but including the influence of the time constant, yielded a heat load of 4729 W. This value deviates by just 1 W compared to the 4730 W obtained using the standard procedure, supporting the conclusion that the difference in accuracy is attributable solely to the asymmetry in the inclusion of the time constant.

5.1.5. Simplified Room Procedure Results

None of the sources which provided references for the standard procedure also provide a reference for the room-level simplified procedure. However, since the room-level and building-level procedure share the code used to calculate transmission losses, large parts of the code are already verified by the building-level test cases. The code for the computation of ventilation losses was unit tested using input data from the EVEBI and wbs reference buildings.

Additionally, the simplified room-level procedure was applied to the wbs reference building to enable a direct comparison with the standard procedure at room level. As with the building-level simplified procedure, mechanical ventilation systems are not supported by the procedure, and the building was therefore modified to use a non-mechanical ventilation configuration as described in Section 5.1.4. The results for both procedures are compared in Figure 10.

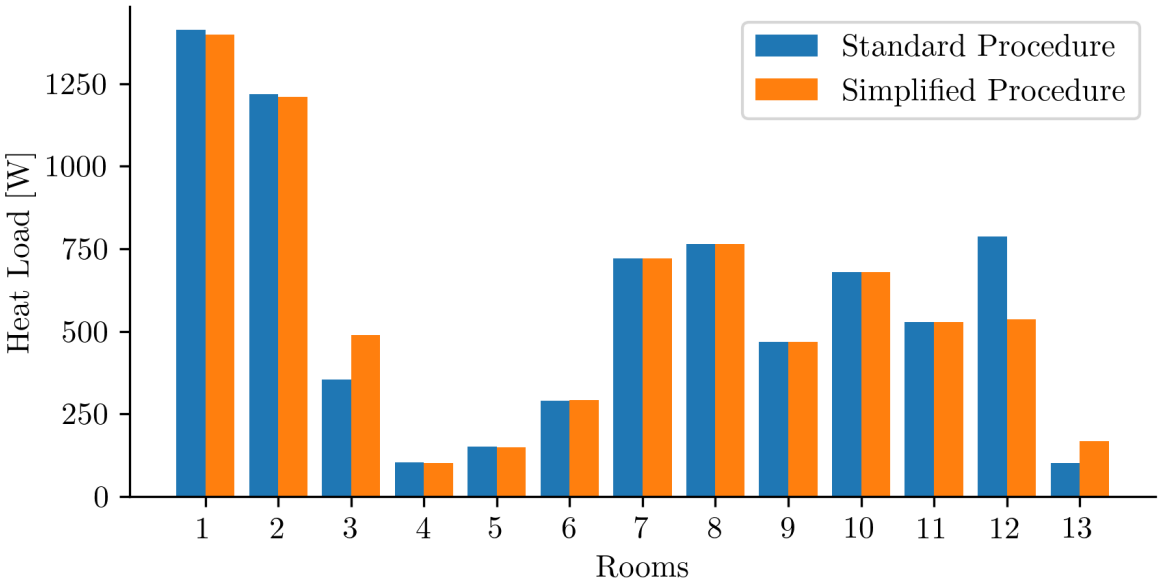

Figure 10: Bar plot comparing the room-level heat loads calculated by the module using the standard and simplified procedures for the modified wbs reference building [43], from which the mechanical ventilation system was removed.

For rooms 3 and 13, the simplified procedure severely overestimates heat load by up to 63% for room 13. Notably, both of these rooms are corridors, for which no minimum air exchange rate is assumed in the standard procedure. In the simplified procedure, however, a fixed minimum air exchange rate is assumed for all rooms, leading to increased values for ventilation losses for such rooms, and thereby increasing their overall heat load. The overestimation is therefore an expected result of the procedure. For room 12, the simplified procedure underestimates the heat load. This is a result of room 12 having a design temperature 4 K higher than the rooms surrounding it, causing it to loose heat to these rooms via internal walls and doors. While this increase in heat load is considered in the standard procedure, the simplified procedure only considers walls with a temperature difference of more than 4 K. Therefore, this internal heat transfer is not included in the simplified procedure, making the underestimation consistent with the procedure. For all other rooms, the difference in heat load is small, with a maximum relative error of 2%.

5.1.6. Discussion

The results of the evaluation demonstrate that the implementation of the standard procedure produces results consistent with all three reference sources across transmission losses, ventilation losses, and additional heating-up loads. Observed deviations could be traced back to either rounding of input data in the reference documents or errors in the reference values. No deviation was found that is attributable to an implementation error in the module.

The evaluation further revealed errors and inconsistencies in multiple reference sources. The examples published in DIN/TS 12831-1 were found to contain inconsistencies of sufficient extent to render them unsuitable for verification. The examples in E DIN SPEC 12831-1 contain inconsistent $\Phi_{V,sup,i}$ values for five rooms, and the wbs reference document contains structural inconsistencies and erroneous values for $q_{v,techn,i}$. The EVEBI software was found to aggregate gross component areas rather than net areas when computing room envelope areas, resulting in overstated ventilation losses. In each case, the module was found to implement the relevant equations correctly, with the reference constituting the source of the discrepancy.

The evaluation of the simplified building-level procedure indicates that its results are consistent with those generated by EVEBI. Furthermore, the results of the simplified procedure closely match those of the standard procedure across the tested buildings when evaluated under equivalent conditions. The observed overestimation of approximately 8% for the EVEBI building relative to the standard procedure could be fully accounted for by the asymmetric treatment of the time constant (see Section 5.1.4).

The comparison of room-level results between the standard and simplified procedures reveals that the simplified procedure can produce substantial deviations for specific room types. As these deviations are consistent with the differences between the standard and the simplified procedure, they do not reflect implementation errors.

Overall, the evaluation provides strong evidence that the heat load module implements the calculation procedures defined in DIN/TS 12831-1 correctly, with no indication of implementation errors across any of the tested reference cases.

5.2. Hydronic Balancing Module

Unlike the heat load module, the hydronic balancing module involves implementation choices in areas where the applicable standards do not prescribe exact procedures, and some deviations from the reference values are therefore anticipated. The evaluation addresses this by combining unit tests for isolated calculations with full pipe network reference cases from E DIN 94679-1 [26], VDI 2073 Part 2 [41], and ZVPlan [14]. Where deviations are observed, the underlying cause is investigated and, where possible, attributed to specific differences in modeling approach or numerical assumptions between the implementation and the reference.

5.2.1. Methodology

The evaluation of the hydronic balancing module is structured into two stages. The first stage consists of targeted unit tests verifing the correctness of individual calculations in isolation. Here, unit tests are used to verify the computation of radiator return water temperatures and flow rates, as well as the calculation of hydraulic resistances for pipe sections based on roughness coefficients. Reference values for these unit tests were obtained from E DIN 94679-1 and ZVPlan [14].

The second stage evaluates the module against complete pipe network reference cases, assessing the correctness of the full balancing procedure. Three complete pipe network test cases are evaluated, one each from E DIN 94679-1, ZVPlan, and VDI 2073 Part 2. For each test case, the described pipe network was recreated using the data structures defined by the hydronic balancing module, as described in Section 3.5. For the E DIN 94679-1 and ZVPlan reference cases, static balancing results are additionally compared. Where necessary, assumptions made to resolve ambiguities in the references are explicitly documented.

The module is then executed using the reconstructed input data, and the resulting flow rates, pressure losses, and valve settings computed by the implementation are compared against the corresponding reference values. Available intermediate results are additionally compared in order to ensure correctness and identify the underlying causes of any occurring discrepancies. As reference values are generally reported with limited numerical precision, deviations are only considered significant if they exceed the uncertainty attributable to the rounding of the published values. Such cases are identified and discussed explicitly in the evaluation.

5.2.2. Radiator Flow Rates

Unit tests for the computation of radiator return water temperatures and flow rates were derived from example calculation C.1 of E DIN 94679-1, covering six radiators with differing nominal and required heat outputs. For five of the six radiators, the return water temperature computed by the module matches the reference. However, for one, the module computes a return water temperature 2 K higher than the reference value. The source of this discrepancy remains unclear, however, manual recalculation using the approach presented in E DIN 94679-1 supports the higher value computed by the module.

As shown in Table 3, the computed mass flow rates deviate by up to 0.66% from the reference values. As the reference does not specify the heat capacity values used for the flow rate calculation (see Section 2.2.2), the deviations cannot be traced back to a specific cause. The module therefore falls back to its built-in water heat capacity approximation based on the supply water temperature. Although the relative error remains below 1%, the non-linear nature of hydraulic resistance means that even small deviations may have a notable impact on subsequent pipe network calculations.

Table 3: Reference and computed mass flow rates for E DIN 94679-1 example C.1, including input parameters room heat load and rated radiator output, with $\vartheta_s = 55°C$.

Circuit Number	**Heat Load [W]**	**Rated Output [W]**	**Ref. Flow Rate [kg/h]**	**Comp. Flow Rate [kg/h]**	**Rel. Error [%]**
1.1	591	1200	42.4	42.255	0.34%
1.2	618	1200	52.8	53.150	0.66%
1.3	911	1850	65.4	65.101	0.46%
2.1	932	1850	72.9	72.809	0.12%
2.2	1724	3500	123.6	123.328	0.22%
2.3	304	590	26.1	26.205	0.40%

To further verify the computation across a wider range of operating conditions, reference values were extracted from ZVPlan [14] for a radiator with a rated heat output of 4000 W under three different load conditions, for supply water temperatures between 30 °C and 80 °C. In contrast to E DIN 94679-1, ZVPlan adopts the logarithmic excess temperature definition (see Section 2.2.2). The three load conditions of 400 W, 2000 W, and 3000 W represent one overdimensioned radiator suitable for operation at very low supply water temperatures, one reasonably dimensioned radiator, and an underdimensioned radiator requiring high supply water temperatures respectively. Reference values are rounded to one decimal place and were extracted at 2 K intervals across the full supply temperature range. Like the E DIN 94679-1 example, ZVPlan does state the heat capacity values used for the calculations, and the module uses its built-in temperature based approximation. The results are shown in Figure 11, where module outputs are represented as lines and ZVPlan reference values as markers.

Across all data points, the mean relative error is 0.8%, with a maximum of 2%. The relative error tends to increase for smaller mass flow rates, which is likely attributable to rounding of the extracted reference values constituting a larger share of the total value at lower magnitudes.

Overall, the computation of radiator return water temperatures and mass flow rates is therefore considered successfully verified for both excess temperature definitions.

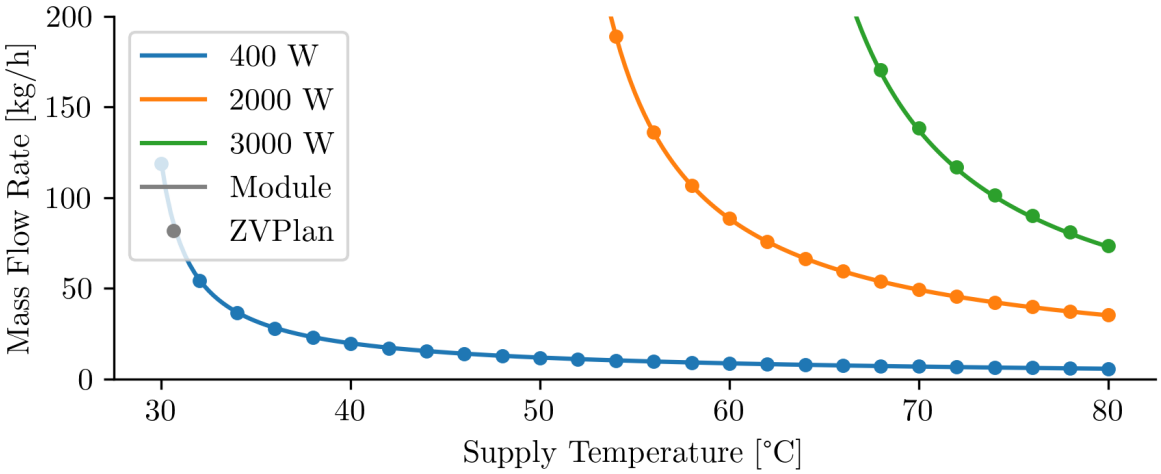

Figure 11: Comparison of mass flow rates computed by the module (lines) against ZVPlan reference values (markers) for a 4000 W radiator under load conditions of 400 W, 2000 W, and 3000 W, across supply temperatures from 30 °C to 80 °C. Reference values are shown in 2 K steps.

5.2.3. Pipe Friction

Reference pipe friction values were obtained from two reference tables provided in E DIN 94679-1, one covering copper and soft steel pipes with a surface roughness of $k = 0.03$ mm and one covering plastic and plastic-metal composite pipes with a surface roughness of $k = 0.01$ mm. Both tables cover a range of pipe diameters, flow rates, and roughness values. The results of the comparisons between the pipe friction values computed by the module and the reference values from the tables are shown in Figures 17 and 18 in Appendix A. Across all entries, relative errors lie within a range of 0.8% for both tables, with the module producing slightly higher values than the ones in the tables of E DIN 94679-1 for most configurations. As E DIN 94679-1 does not specify the correlation used to calculate the friction coefficient (see Section 2.2.5), the origin of these deviations is unclear. Furthermore, while the majority of pipe diameter and flow rate combinations fall within the turbulent flow regime, some correspond to conditions in the transitional regime, for which determination of the friction coefficient λ is not definitive. Based on the values stated for these cases, E DIN 94679-1 appears to adopt the conservative approach of treating the flow as turbulent, which is mirrored in the implementation of the hydronic balancing module. Given that the deviations are small across all tested configurations, the pipe friction calculation is considered well-aligned with the E DIN 94679-1 reference.

When comparing the pipe friction losses computed by the module against those produced by ZVPlan for the pipe sections in the full reference pipe network test cases, the module consistently produces lower friction loss values, with deviations of up to 11%. ZVPlan does not state the friction coefficient correlation employed or the water temperature assumed for its calculations, both of which can have a substantial impact

on pipe friction losses (see Section 2.2.5). Attempts to reproduce the ZVPlan values by substituting other common correlations, such as the Blasius correlation [1], and by varying common estimates for water density and dynamic viscosity, were unsuccessful. The origin of these deviations therefore remains unidentified.

Nevertheless, given the small magnitude of the deviations relative to E DIN 94679-1, the pipe friction calculation is considered sufficiently accurate for the intended application. The discrepancies observed against ZVPlan remain unexplained due to the unavailability of its underlying correlation, and consultation of an additional reference source would be beneficial for more conclusive verification in future work.

5.2.4. Pipe Network Calculation and Balancing

E DIN 94679-1 In E DIN 94679-1, one of the provided example calculations for static hydronic balancing describes a two-pipe heating system consisting of a single heat generator and six radiators, arranged in two branches of three radiators each in a direct-return configuration (see Section 2.2.1). The example assumes TRVs with 16 discrete settings and includes a detailed enumeration of the pressure losses caused by each pipe section and system component, enabling comparison of intermediate values.

Comparing the pressure losses computed by the module against those stated in the reference revealed multiple deviations, with relative errors of up to 3.1% observed for some pipe sections. Several factors may contribute to these discrepancies. The module calculates friction losses for each pipe section individually, using the hydraulic properties of water at the local pipe temperature, which results in higher friction losses in the return network due to the increased dynamic viscosity μ of water at lower temperatures. The reference, combines each supply pipe and its corresponding return pipe into a single section, computing a single friction coefficient for both pipes. As the physical properties of water assumed in the reference calculations are not stated, the magnitude of the impact of this difference in approach cannot be estimated reliably. The previously observed tendency of the module to produce slightly higher friction coefficients than the E DIN 94679-1 reference tables, as well as deviations in mass flow rates discussed in Section 5.2.3, could additionally contribute to the observed discrepancies.

Additionally, inconsistencies were identified within the reference data itself. For example, the pressure loss attributed to the heat generator is stated as 406 Pa in the calculation of one emitter circuit, and 426 Pa in the remaining five circuits. Multiple comparable inconsistencies were found, affecting pipe diameters, water velocities, and overall circuit pressure losses. In cases where multiple conflicting values were present for a single input parameter, the value most consistent with the surrounding reference data was selected, and the assumption is noted accordingly.

Despite these discrepancies at the intermediate level, Table 4 shows that the total circuit pressure losses excluding valves exhibit only minor deviations, with the largest being 46 Pa, constituting a relative error of 0.8%. Furthermore, the valve settings determined by the module match those stated in the reference across all circuits. As the valve settings are the most important output of the static balancing procedure, the verification against this reference case is overall considered successful.

Table 4: Comparison of total circuit pressure losses without valves Δp_{const} and TRV settings computed by the module against the reference values from example C.1 of E DIN 94679-1 [26], covering all six emitter circuits.

Circuit Number	**Ref.** Δp_{const} [Pa]	**Comp.** Δp_{const} [Pa]	**Ref. TRV Setting**	**Comp. TRV Setting**
1.1	2969	2961.91	3.0	3.0
1.2	4132	4133.81	3.5	3.5
1.3	5466	5499.39	4.0	4.0
2.1	3902	3925.04	4.0	4.0
2.2	5769	5722.80	8.0	8.0
2.3	5458	5426.02	2.0	2.0

ZVPlan The ZVPlan reference case describes a two-pipe heating system consisting of two branches with two and three radiators respectively. As discussed in Section 5.2.3, the module consistently produces pipe friction losses substantially lower than those reported by ZVPlan, and the origin of this discrepancy could not be identified. In order to still allow for meaningful comparison of the balancing procedure itself, the pipe friction losses reported by ZVPlan were used to override the values computed by the module for this test case. The example employs continuously adjustable TRVs with a valve position range of 1 to 6, in combination with adjustable return line valves.

Tables 5 and 6 compare the circuit pressure losses and valve settings computed by the module against the ZVPlan reference values for all five emitter circuits. The circuit pressure losses excluding valves deviate by less than 1%, and the pressure losses attributed to the adjusted TRVs deviate by less than 0.6%. The computed K_v values and valve settings match the reference values exactly within rounding precision, except for a minor deviation in the deducted position settings of circuit 2.3. Consistent with the result produced by the module, the return line valves remain fully open across all circuits in the reference solution. As these valve settings are the primary output of the balancing process, the verification against this reference case is considered successful.

Table 5: Comparison of total circuit pressure losses without valves Δp_{const} and pressure loss of adjusted TRVs Δp_{trv} computed by the module against the reference values from ZVPlan [14], covering all five emitter circuits.

Circuit Number	**Ref.** Δp_{const} [Pa]	**Comp.** Δp_{const} [Pa]	**Ref.** Δp_{trv} [Pa]	**Comp.** Δp_{trv} [Pa]
1.1	4310	4304.48	10200	10157.89
1.2	4290	4248.29	10370	10371.77
2.1	6630	6574.30	7600	7616.33
2.2	8670	8655.11	5900	5880.89
2.3	9790	9765.52	4700	4673.67

Table 6: Comparison of TRV K_v values and corresponding setting positions computed by the module against the reference values from ZVPlan [14], covering all five emitter circuits.

Circuit Number	**Ref. TRV** K_v [m³/h]	**Comp. TRV** K_v [m³/h]	**Ref. TRV Position**	**Comp. TRV Position**
1.1	0.188	0.188	4.5	4.47
1.2	0.073	0.073	2.5	2.49
2.1	0.323	0.323	6.0	6.00
2.2	0.172	0.172	4.3	4.27
2.3	0.249	0.249	5.3	5.23

VDI 2073 Part 2 To further verify the pipe network calculation, the module was evaluated against a calculation example provided in VDI 2073 Part 2. The example describes a heating system with five emitter circuits and specifies pressure losses for every pipe section. As with the E DIN 94679-1 example, the reference performs calculations for the supply and return networks jointly.

Comparing the pressure losses computed by the module against the reference values yielded a maximum relative error of 1.5%. Under the assumption that these deviations arise from the module calculating supply and return pipes separately using local water temperatures, the calculation was repeated assuming a constant water temperature throughout the system. For four of the five circuits, this reduced the maximum relative error to below 0.6%. For the remaining circuit, circuit 3, the relative error increased to 2.7%. Closer investigation revealed that this is caused by an arithmetic error in the reference calculation. Most notably, Table A.2 of VDI 2073 Part 2 states $16 \cdot 102 = 2006$, an error that cannot be attributed to rounding imprecision. Several further minor inconsistencies are present in the same table, such as $32 \cdot 102 = 3279$. However, these inconsistencies lie within the range of rounding imprecision and are likely the cause of the small deviations observed for the other circuits. After correcting the identified arithmetic error in circuit 3, the relative error for the circuit reduces to 0.07%. The verification against the VDI 2073 Part 2 reference is therefore considered successful, with all deviations attributable to rounding imprecision and calculation errors in the reference document rather than errors in the implementation.

5.2.5. Supply Temperature Optimization

The supply temperature optimization algorithm builds upon the pipe network calculation implementation, which was successfully verified against E DIN 94679-1 and VDI 2073 Part 2 and, with the exception of pipe friction losses, against ZVPlan.

ZVPlan does not support cost-optimised replacement recommendations for pipes and radiators, but offers functionality for reducing the supply temperature as far as possible without replacing radiators, while allowing pipe dimensions to be increased without cost constraints. When applied to the reference case used for static balancing verification,

ZVPlan determines that the original supply temperature of 64 °C can be reduced to 59 °C by upsizing a subset of the system pipes.

To replicate the behavior of ZVPlan of only upsizing pipes in order to enable comparison, the supply temperature optimization algorithm was configured with the following replacement costs: Upsizing a pipe section was priced at a fixed cost of 1, and replacing a radiator was priced at $100 + \Phi_N/10^6$, where the small additional term proportional to the rated output ensures that smaller replacement models are preferred over larger ones, preventing substantial oversizing. This pricing structure biases the algorithm towards replacing pipes before radiators, reflecting the behavior of ZVPlan, and further incentivises the selection of the smallest admissible replacement dimensions. The set of radiator models available for replacements is based on standard sizes defined in DIN EN 442-2 [20]. With these settings, the algorithm likewise determined that a supply temperature of 59 °C is achievable without radiator replacement, identifying the same pipe sections for upsizing as ZVPlan. The verification of the supply temperature optimization procedure against the ZVPlan reference is therefore considered successful.

The algorithm was further able to reduce the supply temperature to 48 °C, a total reduction of 14 K from the design supply temperature, by replacing all five radiators and multiple pipe sections. Using pipe network calculations, all solutions found by the MILP were successfully verified.

To assess the quality of the solutions produced by the MILP formulation, a greedy algorithm was constructed as a baseline for comparison. The greedy algorithm, described in Algorithm 2, always performs the single cheapest available action towards a feasible solution at each step, without consideration of global optimality.

Algorithm 2: Greedy supply temperature feasibility restoration

Require: Target supply temperature ϑ_s, heating system graph G, replacement options
Ensure: Modified graph G' with all pipe velocity constraints satisfied, or failure
1: Replace all radiators that cannot operate at ϑ_s with the cheapest admissible model
2: **while** pipe $p \in G$ is overloaded **do**
3: Perform single cheapest radiator or pipe upsizing influencing p
4: **end while**
5: **if** all pipe velocity constraints satisfied **then return** modified graph G'
6: **else return** failure
7: **end if**

Both algorithms were evaluated on the previously defined reference case and cost function. Figure 12 compares the cumulative replacement cost required by each algorithm to achieve a given supply temperature. Starting at 57 °C, the greedy algorithm consistently produces solutions with higher costs than the MILP. However, for supply temperatures lower than 51 °C, the magnitude of these discrepancies starts to decrease. This is likely attributable to two factors. Unlike higher temperatures where only pipe upsizing is needed, the temperature range between 57 °C and 51 °C requires at least one radiator replacement.

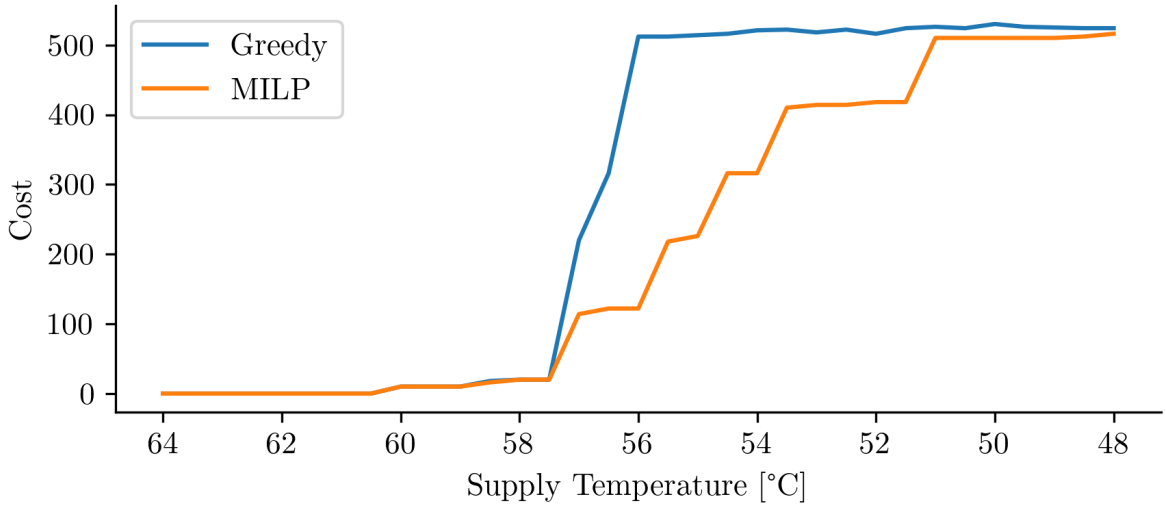

Figure 12: Cumulative replacement cost as a function of target supply temperature for the MILP-based optimization and the greedy baseline algorithm, applied to the ZVPlan reference heating system. The MILP consistently achieves lower or equal costs across the full temperature range. The x-axis is reversed to reflect decreasing supply temperature during iteration.

As the greedy algorithm always selects the cheapest and not the most optimal radiator, the differences between the approaches become most apparent from that point on. At lower temperatures, many radiator models are excluded due to insufficient output capacity, which reduces the solution space and limits the degree to which the greedy approach can deviate from the optimum.

5.2.6. Discussion

The evaluation demonstrates that the hydronic balancing module implements its calculation procedures correctly across all tested reference cases. Radiator flow rate computations for both excess temperature definitions, and pipe friction calculations exhibit only minor deviations from their respective references. The systematic discrepancy of up to 11% observed against ZVPlan pipe friction values remains unexplained and is identified as a topic for further investigation.

The pipe network calculation was verified against all three reference cases, with deviations consistent with the methodological difference of treating supply and return pipes separately. As the output of the pipe network calculation contains the mass flow rates needed to perform dynamic balancing, it too is considered successfully verified. Both E DIN 94679-1 and VDI 2073 Part 2 example calculations were additionally found to contain internal inconsistencies. The static balancing procedure produced valve settings matching the reference exactly in all circuits for both the E DIN 94679-1 and ZVPlan reference cases, which constitutes the most practically significant result of the evaluation.

The supply temperature optimization algorithm was verified against ZVPlan for the

pipe upsizing scenario, producing identical results. The comparison with the greedy baseline confirms that the MILP formulation consistently identifies lower-cost solutions and achieves a lower minimum feasible supply temperature than the greedy algorithm.

Overall, no deviation was found that is attributable to an implementation error, and the evaluation provides strong evidence that the module implements its calculation procedures correctly.

5.3. Performance

To asses whether the non-functional performance requirement **N1** is met for both modules, the performance of the implemented algorithms and procedures is evaluated. All execution time measurements are obtained on a single core without parallelism on an Apple MacBook Pro 16-inch equipped with an M2 Max chip and 32 GB of RAM, running macOS 26.3, with the library compiled using the `-Os` optimization setting. Measurements are repeated 1 000 times in order to ensure statistical reliability of the reported mean and standard deviation values.

5.3.1. Heat Load Module

The performance of the heat load module was evaluated by measuring execution times across the standard procedure test cases derived from the E DIN SPEC 12831-1, EVEBI, and wbs reference buildings (see Section 5.1.2), as well as their converted equivalents for the simplified building-level and simplified room-level procedures.

Combining all reference buildings, the standard procedure yielded a mean execution time of 167.0 µs with a standard deviation of 24.6 µs. The simplified building-level procedure was substantially faster, with a mean of 0.8 µs and a standard deviation of 0.6 µs, reflecting its significantly reduced computational scope. The simplified room-level procedure yielded a mean of 5.2 µs with a standard deviation of 1.0 µs, higher than the building-level procedure due to the additional per-room calculations.

Overall, the execution times of all three procedures are well within the interactive performance requirement defined by **N1**, with even the most computationally intensive standard procedure completing in under 1 ms for the tested reference buildings.

5.3.2. Hydronic Balancing Module

The performance of the hydronic balancing module was evaluated using the two pipe network test cases introduced in Section 5.2.4, namely the E DIN 94679-1 and ZVPlan reference networks.

The pipe network calculation yielded a mean execution time of 532.5 µs with a standard deviation of 79.5 µs. Profiling indicates that approximately 10.5% of this time is spent on the iterative solving of pipe friction coefficients, and approximately 20% on the BFS traversals used to identify hydraulic circuits. The remaining time is largely attributable to overhead from dictionary lookups, which accumulates across the large number of lookups required during the pipe network calculation. This overhead could likely be substantially

reduced by switching to the reference type graph discussed in Section 4.3.1. The resulting execution time of under 1 ms is nonetheless considered acceptable for interactive use, satisfying requirement **N1**.

Static hydronic balancing yielded a mean execution time of 8.3 ms with a standard deviation of 0.1 ms. Profiling shows that approximately 63% of this time is spent solving the second MILP model and approximately 18% solving the third, with only 4% attributed to the first. The relatively short solve time of the first model is consistent with the structure of the reference cases, for which selecting the least restrictive valve setting for each valve yields a valid solution. More specifically, due to the non-linear relationship between valve setting and pressure loss, the fully open position is the most optimal valve position for any valve in the system. Therefore, solving the LP-relaxation automatically provides an integer solution for the emitter circuit with the highest pressure loss, resulting in a tight LP relaxation that requires only a small number of branch-and-bound iterations to verify integrality of the remaining valves.

The third model solving faster than the second is expected, as it inherits the tighter bounds established by the preceding two stages, resulting in a better-constrained problem. As the duration of this calculation is less than 16.7 ms, which is the frame time of a 60 frames per second (FPS) device, this value is likewise considered to satisfy the interactivity requirement of **N1**.

The supply temperature optimization algorithm was evaluated on the ZVPlan reference test case. The greedy algorithm completed in 0.72 s, while the MILP-based algorithm required 2.46 s in total. The median solve time per MILP iteration is 63 ms, however, individual iterations exhibit spikes of up to 635 ms, as shown in Figure 19 in Appendix A, illustrating the sensitivity of MILP solve times to small changes in the problem instance. Moreover, applying the MILP approach to a larger building containing 18 radiators resulted in solve times upwards of 10 min per iteration, which is considered not acceptable for interactive use.

One approach to alleviating this is to remove pipe upsizing from the MILP model and instead upsize all replaceable pipe sections unconditionally at the start of each iteration, substantially reducing the solution space. Whether a specific pipe section actually requires upsizing can then be determined after the fact. However, while testing showed the performance increasing potential of this approach, it does not account for pipe replacement costs in the optimization.

Attempts to reduce solve times using the presolve functionality of `lp_solve` [7] were unsuccessful, as the solver incorrectly identified the model as infeasible when presolving was enabled. Specifying alternative branch-and-bound strategies and scaling methods, as well as explicitly declaring applicable constraints as special order sets (SOS) constraints, likewise yielded no improvement. Substituting a more capable MILP solver is therefore identified as a promising avenue for performance improvement.

In summary, the pipe network calculation and static hydronic balancing procedures both complete well within the interactive performance threshold defined by requirement **N1**. The MILP-based supply temperature optimization is considerably too slow for interactive deployment in its current form, and improving its performance is identified as a priority for future work.

6. Conclusion

This work presented the design, implementation, and evaluation of a Swift library that implements the heating system optimization measures required by §60c of the GEG for residential buildings. The library successfully combines heat load calculation in accordance with DIN/TS 12831-1 and DIN EN 12831-1, static and dynamic hydronic balancing for two-pipe heating systems, and supply temperature optimization with cost-optimised, technically feasible replacement recommendations for radiators and pipes. In contrast to existing engineering software, the developed solution provides these functions through a programmatic, interface-independent API, enabling integration into automated workflows and third-party applications. The evaluation showed that the implemented procedures work as expected.

The evaluation of the heat load module demonstrated close agreement with multiple independent reference sources across all tested scenarios, with execution times of less than 1 ms. All observed deviations could be traced back to rounding effects, limited precision of published reference values, or errors within the reference documents, rather than to errors in the implementation. In several cases, discrepancies were identified within the reference documents themselves, including arithmetic mistakes and conflicting parameter values, underscoring the importance of cross-referencing multiple independent sources when implementing normative calculation procedures.

For the hydronic balancing module, the verification confirmed that the implemented pipe friction calculations, flow rate determination, and balancing procedure produce results consistent with the reference examples of E DIN 94679-1 and VDI 2073 Part 2, with only minor deviations attributable to differences in modeling choices or numerical precision. Comparisons of these procedures with ZVPlan revealed close agreement in the overall pipe network calculation and balancing procedure, although systematic differences were observed in the pipe friction losses whose origin could not be conclusively identified.

The supply temperature optimization procedure successfully determined the lowest feasible supply temperature while respecting hydraulic and thermal constraints, and is capable of generating technically valid replacement recommendations using a MILP formulation. The performance requirements for interactive use are satisfied by the greedy optimization approach, whereas the MILP-based optimization currently exhibits execution times higher than desired for interactive applications. However, the MILP consistently produces lower-cost solutions than the greedy algorithm and is the only approach capable of finding a solution at the minimum achievable supply temperature, demonstrating that the quality advantage of the optimal formulation justifies the additional effort required to improve its performance. Profiling indicates that the majority of the runtime is spent in the MILP solver, suggesting that the use of a more capable solver would substantially improve performance.

Overall, the evaluation results demonstrate that the developed library correctly implements the required calculation procedures and provides a flexible foundation for automated heating system analysis and optimization, fulfilling the objective of compliance with the measures mandated by §60c of the GEG.

6.1. Future Work

Based on the findings of this work, several directions for future development emerge.

The most critical aspect requiring improvement is the performance of the MILP-based supply temperature optimization. For some problem instances, the current implementation using `lp_solve` exhibits solve times higher than desired for use in interactive applications. Substituting a more capable MILP solver, or adapting the model, is expected to substantially reduce these times.

Furthermore, the library currently requires all heating system component data to be provided programmatically by the integrating application. Implementing support for the VDI 3805 [42] data exchange interface would allow manufacturer-provided component data, such as radiator characteristics and valve resistance curves, to be imported directly, substantially reducing the effort required to model real heating systems.

The current supply temperature optimization determines the lowest achievable supply temperature and the associated replacement costs. A natural extension would be to incorporate projected energy savings resulting from the reduced supply temperature, based on its impact on heat pump or gas boiler efficiency, and circulation pump energy consumption. This would enable the MILP formulation to be extended to the optimization of the return on investment over a specified time horizon, rather than minimizing replacement cost alone.

Finally, as E DIN 94679-1 is currently available only in draft form, the implementation should be reviewed and adapted once the standard is finalised, to ensure compliance with any changes introduced in the published version.

A. Charts

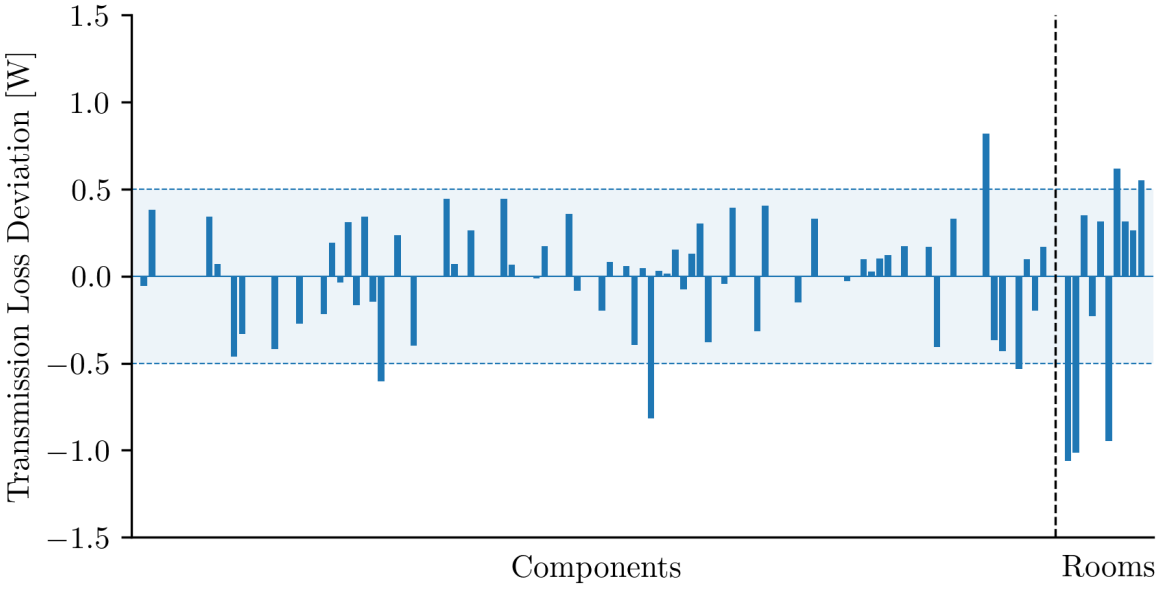

Figure 13: Bar plot illustrating the deviations between the transmission losses calculated by the module and the reference values calculated by EVEBI [28] for the reference building. Results are presented for all 111 components and 10 rooms. Deviations below a magnitude of 0.5 W are treated as negligible.

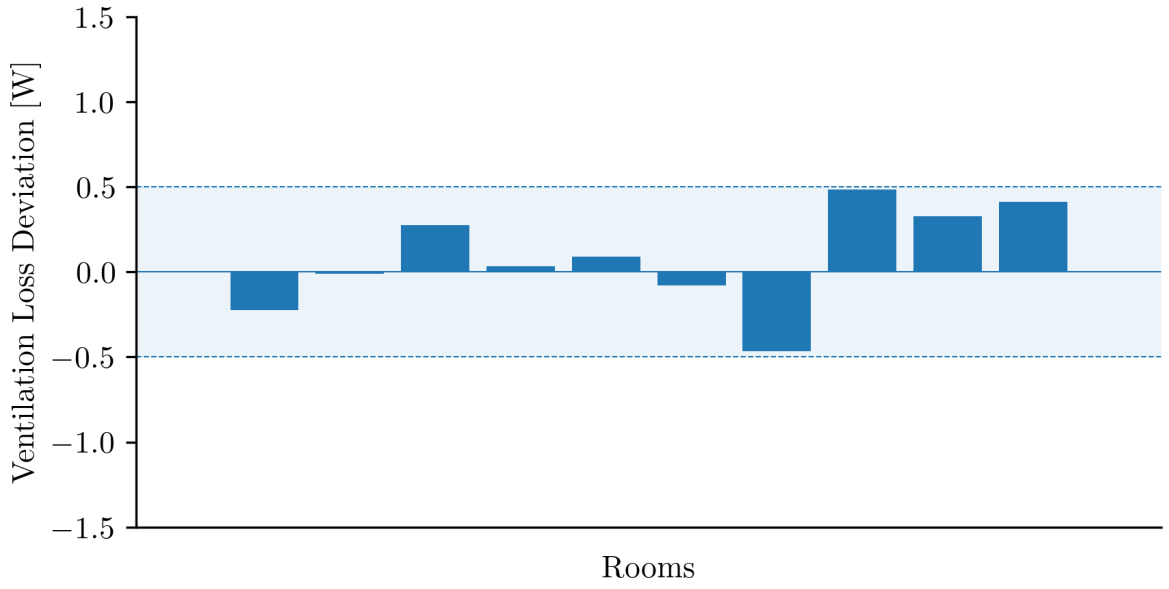

Figure 14: Bar plot illustrating the deviations between the ventilation losses calculated by the module and the reference values calculated by EVEBI [28] for the reference building. For the calculation by the module, the adjusted component surface areas are used. Results are presented for all 10 rooms. Deviations below a magnitude of 0.5 W are treated as negligible.

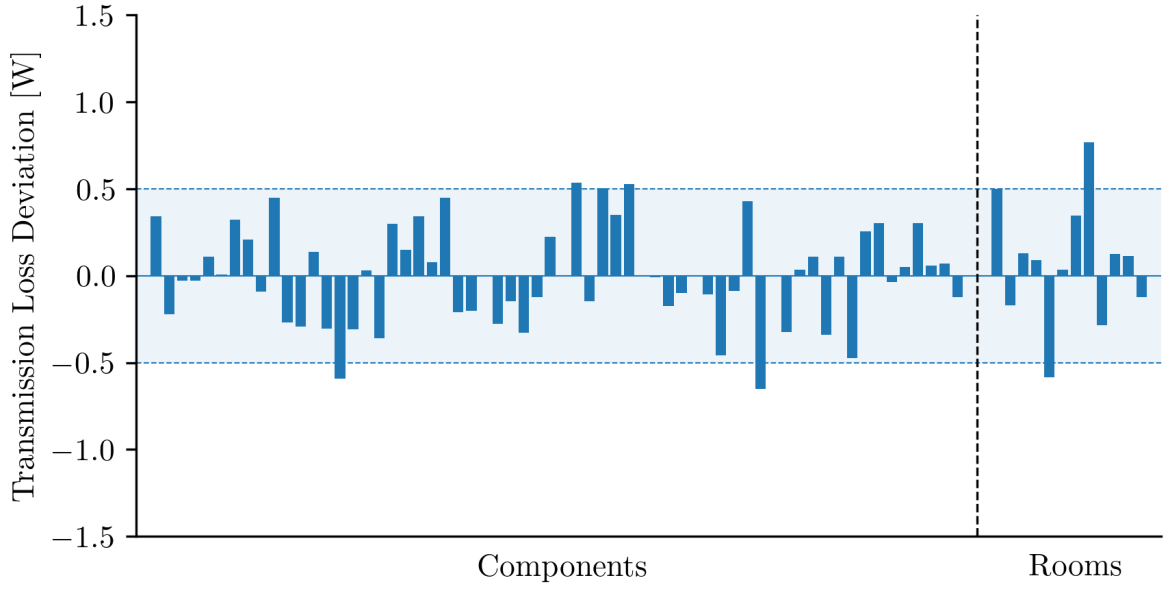

Figure 15: Bar plot illustrating the deviations between the transmission losses calculated by the module and the reference values provided by wbs [43]. Results are presented for all 64 components and 13 rooms. Deviations below a magnitude of 0.5 W are treated as negligible.

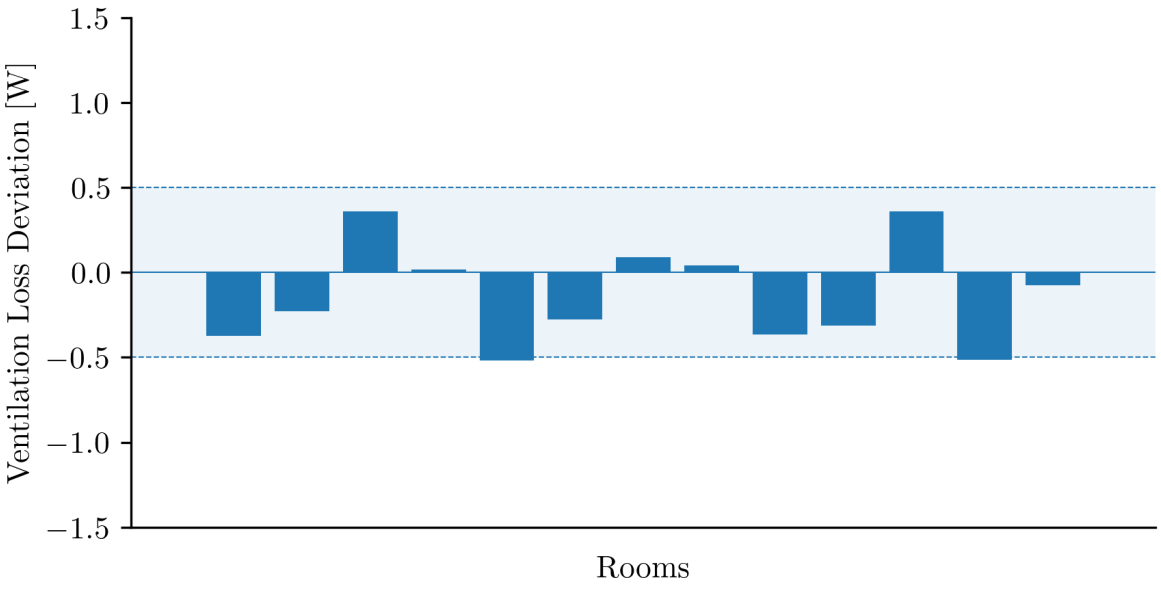

Figure 16: Bar plot illustrating the deviations between the ventilation losses calculated by the module and the reference values provided by wbs [43]. Results are presented for all 13 rooms. Deviations below a magnitude of 0.5 W are treated as negligible

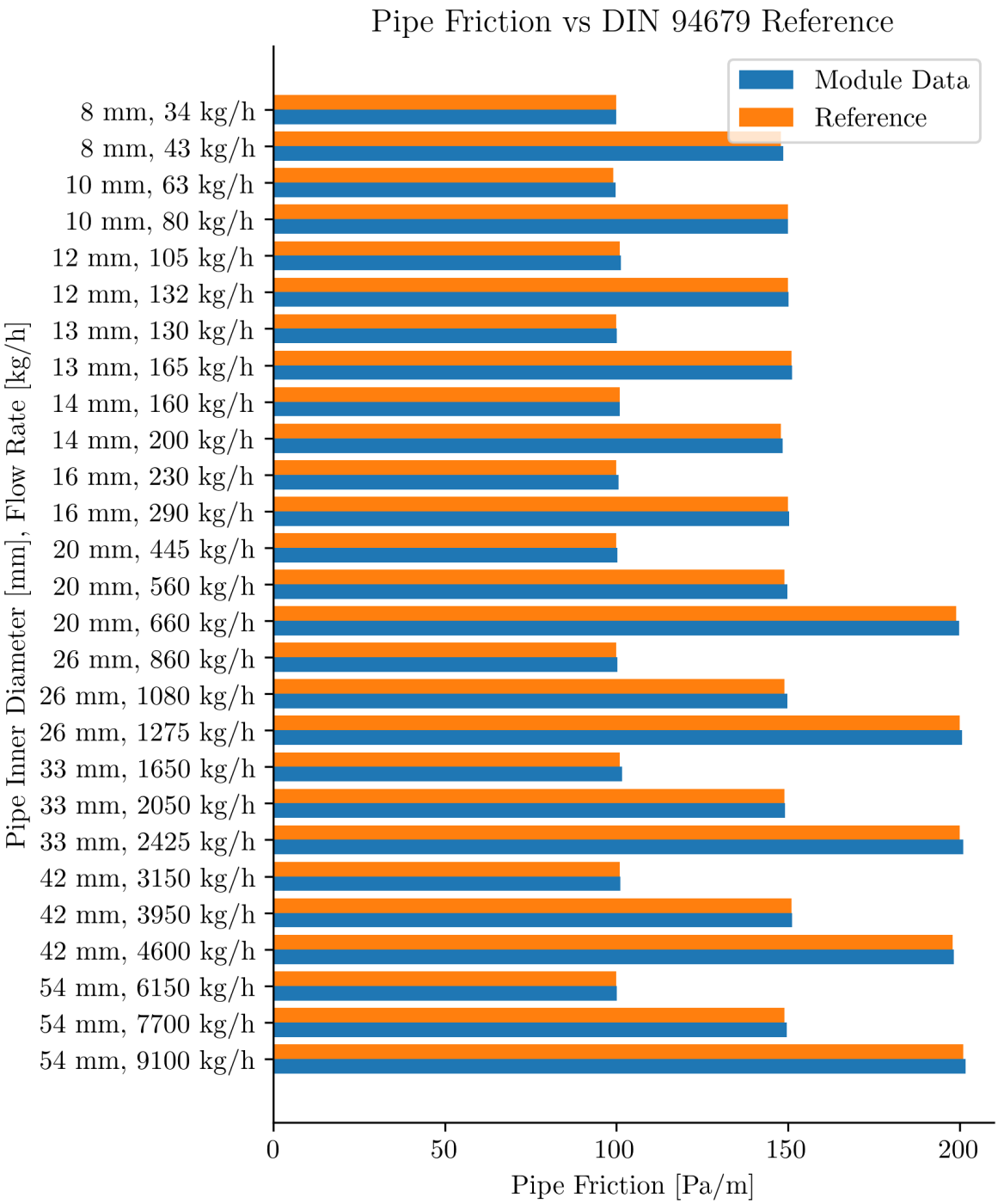

Figure 17: Comparison of pipe friction pressure drop [Pa/m] between module calculations and E DIN 94679-1 Table B.3 reference values for plastic and plastic-metal composite pipes with a roughness of 0.01 mm (used condition). Reference conditions: water at 50 ° C, density 988.04 kg/m^3, dynamic viscosity $547.08 \cdot 10^{-6}$ $Pa \cdot s$. Results are shown for pipe inner diameters ranging from 8 mm to 54 mm and up to three nominal flow rate levels resulting in approximately 100, 150, and 200 Pa/m respectively. Deviations between reference and values computed by the module are less than 0.8%.

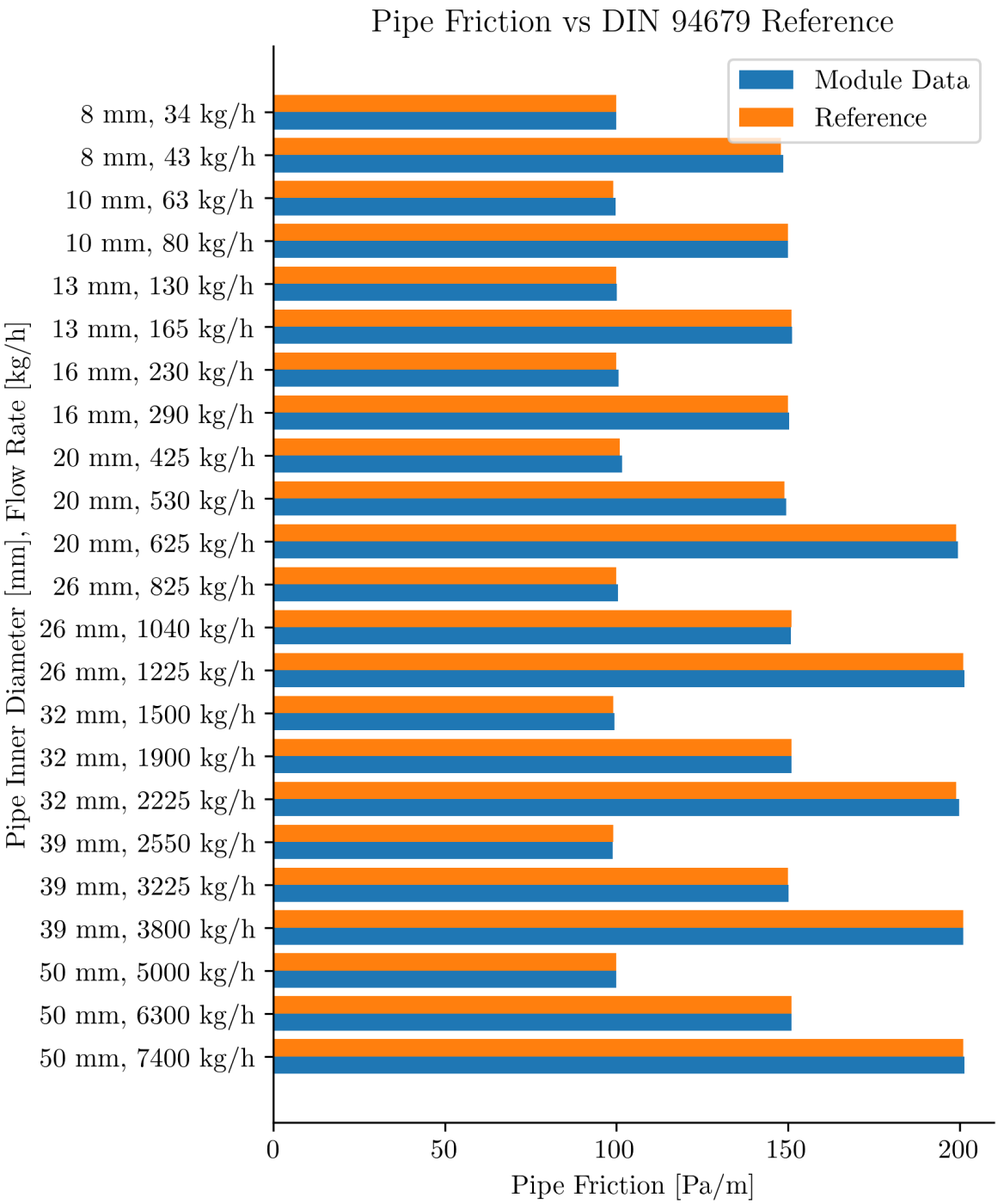

Figure 18: Comparison of pipe friction pressure drop [Pa/m] between module calculations and E DIN 94679-1 Table B.2 reference values for copper and soft steel pipes with a roughness of 0.03 mm (used condition). Reference conditions: water at 50 ° C, density 988.04 kg/m^3, dynamic viscosity $547.08 \cdot 10^{-6}$ $Pa \cdot s$. Results are shown for pipe inner diameters ranging from 8 mm to 50 mm and up to three nominal flow rate levels resulting in approximately 100, 150, and 200 Pa/m respectively. Deviations between reference and values computed by the module are less than 0.8%.

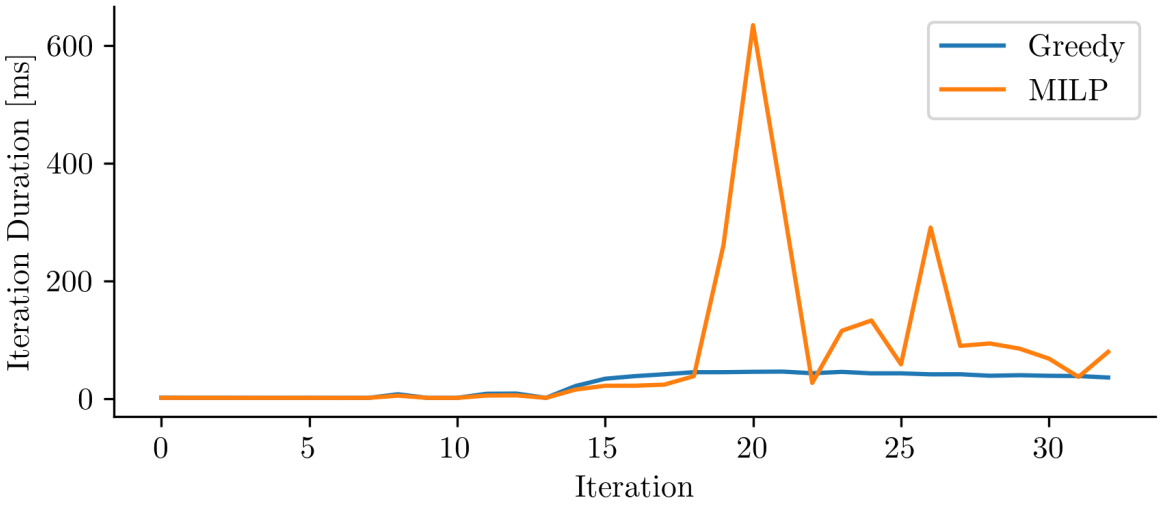

Figure 19: Iteration duration of the greedy and MILP-based supply temperature optimization algorithms across all iterations of the ZVPlan reference test case. The greedy algorithm maintains a consistently low execution time throughout. The MILP solver likewise remains fast for most iterations, but exhibits pronounced spikes of up to 635 ms in the later iterations.

References

[1] Karl-Josef Albers and Ernst-Rudolf Schramek. *Taschenbuch für Heizung und Klimatechnik : einschließlich Trinkwasser- und Kältetechnik sowie Energiekonzepte / herausgegeben von Prof. Dr.-In. Karl-Josef Albers, Hochschule Esslingen 77. 2015/2016,1.* Oldenbourg, München, 2015. ISBN 9783835671362.

[2] Arbeitsgemeinschaft Energiebilanzen e. V. Anwendungsbilanzen zur Energiebilanz Deutschland: Endenergieverbrauch nach Energieträgern und Anwendungszwecken – Detaillierte Anwendungsbilanzen der Endenergiesektoren für 2019 und 2020 sowie zusammenfassende Zeitreihen zum Endenergieverbrauch nach Energieträgern und Anwendungszwecken für Jahre von 2010 bis 2020. Publikation, Arbeitsgemeinschaft Energiebilanzen e. V., May 2021. Accessed 15.03.2026.

[3] ASHRAE American Society of Heating Refrigerating and Air-Conditioning Engineers. *ASHRAE Handbook - Fundamentals.* ASHRAE, 2017. ISBN 978-1-939200-58-7.

[4] autarc energy GmbH. autarc. URL `https://www.autarc.energy`. Accessed 17.03.2026.

[5] Hans Dieter Baehr and Karl Stephan. *Wärme- und Stoffübertragung.* Springer Vieweg, Berlin, Heidelberg, 9 edition, 2016. ISBN 978-3-662-49677-0. doi: 10.1007/978-3-662-49677-0.

[6] BDEW Bundesverband der Energie- und Wasserwirtschaft e. V. Wie heizt deutschland? Ergebnisbericht 2023. Studienbericht, BDEW Bundesverband der Energie- und Wasserwirtschaft e. V., Berlin, November 2023. URL `https://www.bdew.de/media/documents/Wie_heizt_Deutschland_2023_-aktualisierte_Fassung-_BDEW_1.pdf`.

[7] Michel Berkelaar, Jeroen Dirks, Kjell Eikland, and Peter Notebaert. lp_solve: Open source (mixed-integer) linear programming system, 2004. URL `https://lp-solve.github.io/`.

[8] Bundesamt für Wirtschaft und Ausfuhrkontrolle (BAFA). Heizungsoptimierung – Bundesförderung für effiziente Gebäude (BEG): Sanierung Wohngebäude. `https://www.bafa.de/DE/Energie/Effiziente_Gebaeude/%Sanierung_Wohngebaeude/Heizungsoptimierung/%heizungsoptimierung_node.html`, 2026. Accessed 15.03.2026.

[9] Bundesnetzagentur. Gaspreise Großhandel in EUR/MWh. `https://bundesnetzagentur.de/DE/Gasversorgung/%aktuelle_gasversorgung/_svg/Gaspreise/Gaspreise.html`, 2026. Accessed 15.03.2026.

[10] Bundesnetzagentur. Aktuelle Lage der Gasversorgung in Deutschland. `https://bundesnetzagentur.de/DE/Gasversorgung/%aktuelle_gasversorgung/start.html`, 2026. Accessed 15.03.2026.

[11] Bundesrepublik Deutschland. Gesetz zur Einsparung von Energie und zur Nutzung erneuerbarer Energien zur Wärme- und Kälteerzeugung in Gebäuden (Gebäudeenergiegesetz – GEG), 2026. URL https://www.gesetze-im-internet.de/geg/.

[12] Hae-In Cho, Daniel Cabrera, and Martin K. Patel. Estimation of energy savings potential through hydraulic balancing of heating systems in buildings. *Journal of Building Engineering*, 28:101030, March 2020. ISSN 2352-7102. doi: 10.1016/j.jobe.2019.101030. URL https://www.sciencedirect.com/science/article/pii/S2352710219310617.

[13] C. F. Colebrook. Turbulent flow in pipes, with particular reference to the transition region between the smooth and rough pipe laws. *Journal of the Institution of Civil Engineers*, 11(4):133–156, 1939. doi: 10.1680/ijoti.1939.13150.

[14] ConSoft GmbH. ZVPLAN – die kompetente und schnelle lösung, haustechnik professionell zu berechnen. URL https://www.zvplan.de. Software für Heizlast- & Rohrnetzberechnung, Hydraulischen Abgleich und HeizungsCheck nach DIN EN 12831 / DIN EN 15378.

[15] Danfoss. LENO™ MSV-BD manual presetting valves: Data sheet, June 2025. URL https://assets.danfoss.com/documents/513406/AI144986479776en-011401.pdf.

[16] Dendrit Haustechnik-Software GmbH. Dendrit STUDIO. URL https://www.dendrit.com. Accessed 17.03.2026.

[17] Deutsches Institut für Normung. DIN 4703-3:2000-10, Heating appliances - Part 3: Conversion of the standard thermal output, October 2000. URL https://www.dinmedia.de/de/-/-/28818252.

[18] Deutsches Institut für Normung. DIN EN 12828:2014-07, Heating systems in buildings - Design for water-based heating systems; German version EN 12828:2012+A1:2014, July 2014. URL https://www.dinmedia.de/de/-/-/203258170.

[19] Deutsches Institut für Normung. DIN EN 442-1:2015-03, Radiators and convectors - Part 1: Technical specifications and requirements; German version EN 442-1:2014, March 2015. URL https://www.dinmedia.de/de/-/-/207500260.

[20] Deutsches Institut für Normung. DIN EN 442-2:2015-03, Radiators and convectors - Part 2: Test methods and rating; German version EN 442-2:2014, March 2015. URL https://www.dinmedia.de/de/-/-/207503152.

[21] Deutsches Institut für Normung. DIN EN 12831-1:2017-09, Energy performance of buildings - Method for calculation of the design heat load - Part 1: Space heating load, Module M3-3; German version EN 12831-1:2017, September 2017. URL https://www.dinmedia.de/de/-/-/261292587.

[22] Deutsches Institut für Normung. DIN SPEC 12831-1:2018-10 - Draft, Method for calculation of the room heat load - Part 1: National addition to DIN EN 12831-1, with CD-ROM, October 2018. Withdrawn; replaced by DIN/TS 12831-1:2020-04. DOI: https://dx.doi.org/10.31030/2878054.

[23] Deutsches Institut für Normung. DIN/TS 12831-1:2020-04, Method for calculation of the room heat load - Part 1: National addition to DIN EN 12831-1, with CD-ROM, April 2020. URL https://www.dinmedia.de/de/-/-/316645651.

[24] Deutsches Institut für Normung. DIN EN 1264-3:2021-08, Water based surface embedded heating and cooling systems - Part 3: Dimensioning; German version EN 1264-3:2021, 2021. URL https://www.dinmedia.de/de/-/-/332537135.

[25] Deutsches Institut für Normung. DIN EN 1264-1:2021-08, Water based surface embedded heating and cooling systems - Part 1: Definitions and symbols; German version EN 1264-1:2021, August 2021. URL https://www.dinmedia.de/de/-/-/332536841.

[26] Deutsches Institut für Normung. DIN 94679-1:2022-10, Hydronic systems in heating, cooling and ventilation installations - Part 1: Basics of hydronic balancing, October 2022. URL https://www.dinmedia.de/de/-/-/356062673.

[27] Fritz Dietzel and Walter Wagner. *Technische Wärmelehre*. Kamprath-Reihe. Vogel Buchverlag, Würzburg, 10 edition, 2013. ISBN 978-3-8343-3276-9.

[28] ENVISYS GmbH & Co. KG. EVEBI – energieberatersoftware: Groß oder ganz groß – EVEBI oder EVEBI Pro. https://www.envisys.de/energieberatersoftware-evebi/, 2026. Version 14.1.8; Accessed 15.03.2026.

[29] European Building Automation Controls Association (eu.bac). System balancing for technical building systems: a great opportunity for energy savings and comfort. Technical report, eu.bac, 2022. URL https://eubac.org/wp-content/uploads/2021/03/20220520_eubac_System_Balancing_for_TBS.pdf.

[30] Henrik Håkansson. *Model-based optimization of hydronic heating system operations: Enhancing comfort and energy efficiency in the existing building stock*. Licentiate thesis, Chalmers University of Technology, Gothenburg, Sweden, October 2025. URL https://research.chalmers.se/publication/548264.

[31] Frank P. Incropera, David P. DeWitt, Theodore L. Bergman, and Adrienne S. Lavine. *Fundamentals of Heat and Mass Transfer*. John Wiley & Sons, New York, 6 edition, 2006. ISBN 978-0-471-45728-2.

[32] Lewis F. Moody. Friction factors for pipe flow. *Transactions of the American Society of Mechanical Engineers*, 66(8):671–678, 1944. doi: 10.1115/1.4018140.

[33] Minh Duc Nguyen. Bereits jedes dritte Wohngebäude spart dank hydraulischem Abgleich. https://www.co2online.de/news/%bereits-jedes-dritte-wohngebaeude-spart-dank-%hydraulischem-abgleich-1/, November 2023. Accessed 17.03.2026.

[34] Johann Nikuradse. Strömungsgesetze in rauhen Rohren. VDI-Forschungsarbeiten auf dem Gebiete des Ingenieurwesens 361, Verein Deutscher Ingenieure, Berlin, 1933.

[35] Ludwig Prandtl. Neuere Ergebnisse der Turbulenzforschung. *Zeitschrift des Vereins Deutscher Ingenieure*, 77:105–114, 1933.

[36] SOLAR-COMPUTER GmbH. SOLAR-COMPUTER. URL https://www.solar-computer.de. Accessed 17.03.2026.

[37] Statistisches Bundesamt (Destatis). Erzeugerpreisindizes: Energie – GENESIS-Online, Tabelle 61243-0010. https://www-genesis.destatis.de/datenbank/online/%table/61243-0010, 2026. Accessed 15.03.2026.

[38] Peter Stephan, Stephan Kabelac, Matthias Kind, Dieter Mewes, Karlheinz Schaber, and Thomas Wetzel, editors. *VDI-Wärmeatlas*. Springer Reference Technik. Springer Vieweg, Berlin, Heidelberg, 12 edition, 2019. ISBN 978-3-662-52988-1. doi: 10.1007/978-3-662-52989-8.

[39] Swift open source project. Swift.org – Swift is the powerful, flexible, multiplatform programming language. https://www.swift.org/, 2026. Accessed 15.03.2026; current Version: Swift 6.2.4.

[40] Verein Deutscher Ingenieure. VDI 2073 Blatt 1:2014-05, Hydraulic systems in building services - Hydraulic circuits. Technical report, May 2014. URL https://www.dinmedia.de/en/technical-rule/vdi-2073-blatt-1/197958898.

[41] Verein Deutscher Ingenieure. VDI 2073 Blatt 2:2024-02, Hydraulic systems in building services - Hydraulic balancing. Technical report, February 2024. URL https://www.dinmedia.de/en/technical-rule/vdi-2073-blatt-2/374292763.

[42] Verein Deutscher Ingenieure (VDI). VDI 3805 Blatt 1: Produktdatenaustausch in der technischen gebäudeausrüstung - grundlagen. VDI-Richtlinie VDI 3805 Blatt 1, Verein Deutscher Ingenieure, Düsseldorf, July 2022.

[43] wbs – Wärmebedarfservice. Beispiele der heizlastberechnung. https://www.heizlast.de/heizlastberechnung/beispiele, 2018. Accessed 15.03.2026.

[44] Wolfgang M. Willems, editor. *Lehrbuch der Bauphysik: Wärme – Feuchte – Klima – Schall – Licht – Brand*. Springer Fachmedien, Wiesbaden, 2022. ISBN 978-3-658-34092-6 978-3-658-34093-3. doi: 10.1007/978-3-658-34093-3. URL https://link.springer.com/10.1007/978-3-658-34093-3.

[45] Zentralverband Sanitär Heizung Klima (ZVSHK). Zentralverband SHK. https://www.zvshk.de/. Accessed 15.03.2026.

[46] ZUB Systems GmbH. ZUB Helena Heizlast. URL https://www.zub-systems.de/en/produkte/helena/heizlast. Accessed 17.03.2026.

[47] ZVSHK and VdZ – Wirtschaftsvereinigung Gebäude und Energie e. V. Optimierung von Heizungsanlagen im Bestand. Technical report, VdZ – Wirtschaftsvereinigung Gebäude und Energie e. V., April 2022. Nummer 4.2.