Received September 9, 2021, accepted September 19, 2021, date of publication September 22, 2021, date of current version September 30, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3115038

Port-Hamiltonian Modeling of Thermofluid Systems and Object-Oriented Implementation With Modelica I: Thermodynamic Part

FRANCISCO M. MÁRQUEZ[1,2], PEDRO J. ZUFIRIA[2], AND LUIS J. YEBRA[3]
[1]Departamento de Automática, Universidad de Alcalá (UAH), 28801 Alcalá de Henares, Spain
[2]Information Processing and Telecommunications Center, Departamento de Matemática Aplicada a las Tecnologías de la Información y las Comunicaciones, ETSI Telecomunicación, Universidad Politécnica de Madrid (UPM), 28040 Madrid, Spain
[3]Plataforma Solar de Almería, CIEMAT, 04200 Tabernas, Spain

Corresponding author: Francisco M. Márquez (francisco.marquez@uah.es)

The work of Luis J. Yebra was supported by the Universidad de Alcalá, through the Programa propio Giner de los Ríos Research Grant.

ABSTRACT In this paper, we present the physical foundations and the development of the thermodynamic part of a Modelica library with the fundamental components for modeling thermofluid systems. We have chosen Modelica because it is an object-oriented modeling language that allows an elegant design of the library, with a top-down conception that starts from very general components where we model the thermodynamic properties common to all simple substances and descend by inheritance to model the properties of each particular substance. To model the behavior of each component, we have used: classical thermodynamics to define the equilibrium states, the local equilibrium hypothesis of Classical Irreversible Thermodynamics to model the changes of state, and the port-Hamiltonian approach to obtain the equations of the system dynamics. With this formulation, we implement the thermodynamic behavior of ideal gases (including monatomic gases as a particular case), the 2073 substances defined for the CEA (Chemical Equilibrium with Applications) NASA Glenn computer program, the IAPWS Formulation 1995 for the Thermodynamic Properties of Water Substance for General and Scientific Use, and the Syltherm 800 HTF (Heat Transfer Fluid). We also define graphical symbols for each library component that facilitate modeling complex systems with simple drag-and-drop manipulations, component connection, and parameter selection. These symbols are a slightly modified version of those used in bond graphs to facilitate their reading and the representation of the structure of complex systems. We also show the modeling, simulation, and comparison for accuracy, performance, and scalability of some thermodynamic systems implemented with the Modelica Standard Library (MSL) and the proposed library.

INDEX TERMS Bond graphs, Modelica language, object oriented modeling, port-Hamiltonian systems, thermodynamic systems, thermofluid systems.

I. INTRODUCTION

This paper is the first part of a two-part series where we will implement a library of components to model the storage and transport phenomena of mass, momentum, and energy in thermofluid systems from a port-Hamiltonian approach. In [28], we presented the general theory of port-Hamiltonian systems, its graphical representation with bond graphs, its object-oriented implementation with the Modelica language, and its application to simple mechanical systems and electrical networks. In the current paper, we present the port-Hamiltonian modeling of thermodynamic systems, we implement in Modelica the main building blocks of these systems, we propose a graphical representation based on bond graphs adapted for thermodynamic systems, and we model the thermodynamic behavior of many substances of scientific and industrial interest.

The associate editor coordinating the review of this manuscript and approving it for publication was Yizhang Jiang.

A. MOTIVATION

Thermofluid systems play a central role in industries such as oil and gas extraction and distribution, water processing and distribution, or solar thermal energy production and storage. Therefore, it is important to have tools for easy and quick modeling and simulation of these systems. These models will also be helpful to inexpensively test control strategies for an efficient system management.

A general thermofluid system may be composed of several subsystems with different substances in motion or at rest, mixing or chemically reacting and exchanging mass, momentum, and energy with each other and their surroundings.

The modeling of these systems is necessarily multidisciplinary and relies on disciplines such as fluid dynamics to model the movement of thermofluids, chemistry to model the reactions between them, thermodynamics to model energy exchanges, electromechanics to model fluid pumping devices and their control, etc. Added to this complexity is the fact that a complete description of the behavior of a continuous substance requires describing its time evolution at each point in space, i.e., each magnitude x that provides information about a thermofluid system will depend on the time variable $t \in \mathbb{R}$ and the spatial variable $r \in \Omega \subset \mathbb{R}^3$.

To tackle a problem of this complexity and to be able to draw engineering applicable conclusions, it is necessary to make some simplifications that make it intellectually and computationally tractable. The main simplification of this work is to consider thermofluid systems composed of connected subsystems that can be described by a set of easily characterized lumped parameters. With this simplification, we can describe the system behavior with a differential-algebraic system of equations (DAE) which allows using the tools for modeling lumped parameter multiphysics systems described in [28]: the port-Hamiltonian formalism, the bond graph representation, and the object-oriented implementation with the Modelica language. We can summarize the rationale for this choice as follows:

- The port-Hamiltonian formalism starts from the Hamiltonian formalism of classical mechanics and extends it to include non-conservative systems (i.e, systems with energy dissipation) [50]. It provides a framework for modeling physical systems based on the storage and exchange of energy through power ports that interconnect the parts of a system with each other and with their surroundings. The connection of systems with port-Hamiltonian structure forms a new system that preserves this structure, so this formalism is suitable for modeling complex systems by connecting simpler ones. The port-Hamiltonian approach has proved to be also suitable for controller designing [50, Chap.7].
- The bond graph methodology proposed by H. Paynter in the 1960s arises from modeling ideas used in electrical networks and it is based on principles similar to the port-Hamiltonian formalism, but is not formulated with the rigor of differential geometry as the latter [4], [40]. However, bond graphs define a symbology to represent the structure of complex systems and the routes of energy exchange between their components, so that they are commonly used to represent port-Hamiltonian systems.
- The Modelica language is also used to describe the behavior of complex systems by connecting simple components (composition mechanism) [35]. As an object-oriented language, it is designed to model the behavior of general systems and to extend those models to particular systems derived from the previous ones (inheritance mechanism). Tools such as OpenModelica [19] or Dymola® process symbolically these models to obtain the differential equations for simulating the system behavior.

In this article, we study the modeling of phenomena related to energy storage in thermofluid systems without mass transfer (fluids at rest). For this purpose, we employ Equilibrium Thermodynamics [7], [48], which allows us to identify states in a thermodynamic system, and also employ Classical Irreversible Thermodynamics [12] to study the time evolution of these states.

We also show the implementation in Modelica language and the graphical representation of components necessary to model these phenomena, thus extending the library pHlib for modeling multiphysics systems started in [28].

We exemplify the applicability of the procedure by showing how to model the thermodynamic behavior of some substances of industrial interest: we started with the model of general simple substances, continue with the model of ideal gases, add the models of 2073 substances described in [30] for the CEA NASA Glenn computer program, implement the model for water and steam described by the International Association for the Properties of Water and Steam (IAPWS-95 [22]), and finish with the model of the commercial Syltherm 800 HTF [15].

Finally, we also implement models of simple systems with MSL and with pHlib in order to compare the results of both simulations. The new library components have been implemented and simulated with the Dymola tool.

B. LITERATURE REVIEW

The development of the bond graph methodology for modeling multiphysics systems began in the 1960s with the work of H. Paynter [40] and, since then, this methodology has been used to model multidisciplinary engineering systems [4], [5], [24].

The theory of port-Hamiltonian systems began in the late 1990s [49]. It is an evolution of Hamiltonian mechanics [1, Chap.3] to which it adds the concepts of power ports (inspired by bond graph modeling) and Dirac structures [11] to formalize with the language of differential geometry a framework for modeling complex systems by connecting simple systems that exchange energy [16].

In [20], [28], we can find a discussion of the relationship between bond graphs and port-Hamiltonian systems, and [14], [42] have presented the derivation of port-Hamiltonian models from bond graphs.

We can find the Modelica language specification in [35], and [18] is a comprehensive guide showing the use of the language for modeling multidomain systems. References [10], [13], [54] show examples of the development of a Modelica library to model systems using the bond graph methodology.

Two standard references in Equilibrium Thermodynamics and Classical Irreversible Thermodynamics are [7] and [12], respectively. The extension of the Hamiltonian formalism to include the phenomena of dissipation that appear in irreversible thermodynamics can be found in [25], [32]. In [29], [33] this formalism is expressed in terms of metriplectic geometry where the phase space of a system with dissipation is modeled with a metriplectic manifold. This geometric object is a manifold with a double structure: simplectic and metric. The conservative part of the behavior (Hamiltonian dynamics) is modeled with the simplectic structure, and the dissipative behavior (irreversible thermodynamics) with the metric structure. In Section III, we show the relationship between the metriplectic and port-hamiltonian formalisms. We will focus in this article on the modeling of lumped parameter systems, as discussed in the previous section. For a review of the literature on port-Hamiltonian modeling of distributed systems, we refer to [44].

The use of bond graphs for modeling thermodynamic systems can be found in [8, Chap.8], [45], [16, Chap.3]. The description of the MSL implementation for modeling thermofluid systems is in [17], and [9] describes the Modelica implementation of a bond graph library for modeling thermofluid systems.

C. CONTRIBUTIONS

In this paper, we develop an object-oriented implementation, with the Modelica language, of library (pHlib) which provides components for modeling thermofluid systems. As this objective is quite ambitious, we start modeling the thermodynamic properties (fluid at rest). In a later article, we will include the transport phenomena (fluid in motion). The implementation is based on a port-Hamiltonian approach to the concepts of Equilibrium Thermodynamics and Classical Irreversible Thermodynamics and is complemented by a new set of symbols derived from the traditional bond graph representation, modified for expressiveness and readability.

This approach brings together three traditions for modeling physical systems whose communities have had little connection with each other: port-based modeling with bond graphs, Hamiltonian modeling, and object-oriented modeling with Modelica. For this purpose, we have taken advantage of the fact that these three approaches share concepts such as power ports (Modelica generalizes this concept with the class connector) or the hierarchical organization (class inheritance and composition in Modelica). Although there are references that combine some of these modeling approaches (e.g., [13], [16], [20], and [42]), they do not address all three together. Moreover, our approach has an additional advantage: it is not necessary to perform a causality analysis when building the models (as required by traditional bond graph modeling) because neither the port-Hamiltonian modeling nor the tools that process the Modelica language need causality indications to automatically generate the computational equations of the models and simulate them.

Most developments based on the ISO representation of port-Hamiltonian systems rely on coordinate-dependent matrix representations [50, Chap.6]. In Section III, we propose a coordinate-free definition based on differential geometry that allows us to compare the port-Hamiltonian formalism with the metriplectic one [29] and to demonstrate in a general way its ability to model dissipative systems.

The object-oriented approach of the library allows to:

(i) Implement a generic component to model the thermodynamic behavior of any substance defined by its fundamental equation.

(ii) Particularize, from the previous generic component, other components to model less general substances such as ideal gases, perfect gases, and monatomic gases (in this phase of the implementation, we have used the inheritance mechanism of the Modelica language and the possibility of adding new behavioral equations in each refinement step).

(iii) Implement the substance library defined for the CEA NASA Glenn computer program. Our implementation takes into account the different coefficients defined for each temperature range (from one-range substances up to five-range substances). The MSL implementation takes into account only two intervals.

(iv) Implement the version for scientific use (known as IAPWS-95) of the behavioral model and thermodynamic properties of water defined by the International Association for the Properties of Water and Steam. MSL implements the industrial formulation IAPWS-IF97.

(v) Implement the thermodynamic behavior of substances whose fundamental equation is unknown, but some thermodynamic properties (e.g., specific heat, density, etc.) are available (we have chosen the commercial Syltherm 800 HTF as an example).

We have always worked with the variables that Thermodynamics, the port-Hamiltonian formalism, and the bond graph modeling postulate for the thermodynamic domain:

- entropy S, volume V, and mass m for the state x,
- temperature T, pressure p, and chemical potential g for the effort $\mathrm{d}U$, and
- entropy flow rate $\dot{S}$, volume flow rate $\dot{V}$, and mass flow rate $\dot{m}$ for the flow $\dot{x}$.

Note that the handling of entropy introduces conceptual and computational complexity into the models. For this reason, in industrial practice, it is common to work with heat Q and heat flow rate $\dot{Q}$ instead of entropy and entropy flow rate. The terms *true bond graphs* and *pseudo-bond graphs* have been introduced to distinguish bond graphs using entropy flow rate from those using heat flow rate [4], [24].

Despite the difficulty of working with entropy, the use of this variable can help to detect inconsistencies in thermodynamic models by checking compliance with the second principle of thermodynamics.

D. PAPER ORGANIZATION

This paper is organized as follows. In Section II, we summarize the main results on Equilibrium Thermodynamics and justify the election of Classical Irreversible Thermodynamics for modeling the time evolution of thermodynamic systems. From the concept of the *fundamental equation* that models the internal energy of a thermodynamic system, we can define port-Hamiltonian models for the storage elements of that internal energy. Since this fundamental equation is not always available, in many practical cases the system dynamics must be reconstructed from measurable thermodynamic properties. For this reason, in Section II we also review: i) other ways of expressing the fundamental equation by the Legendre transformation (Helmholtz free energy, enthalpy, and Gibss free energy), ii) the fundamental set of response functions (specific isochoric heat capacity, adiabatic compressibility, and adiabatic expansibity), and iii) the primary set of response functions (specific isobaric heat capacity, isothermal compressibility, and isobaric expansibity).

In Section III, we study the ISO representation of dissipative port-Hamiltonian systems to identify its energy conservative and entropy generating parts, and to relate this representation to the metriplectic formalism.

In Section IV, we explain the equations and the Modelica implementation of simple thermodynamic systems. The components that make up the implemented library are as follows: power ports, bonds, effort and flow sources, storage elements, general models of single substances, models of ideal substances (ideal gases, perfect gases, monatomic gases), empirical models for non-perfect substances, IAPWS-95 model for water, and model for Syltherm 800.

In Section V, we model simple systems combining electrical and thermodynamic parts, implement them with MSL and pHlib, simulate them, and analyze the results.

In Appendix, one can see two lists with acronyms and nomenclature.

II. PORT-HAMILTONIAN MODELING OF THERMODYNAMIC SYSTEMS

A. SIMPLE THERMODYNAMIC SYSTEMS

A *simple system* is a homogeneous and isotropic mixture of r chemical substances not acted by electric, magnetic, or gravitational fields. The postulates of equilibrium thermodynamics [7, Chap.1] establish for simple systems that:

(i) There exist equilibrium states characterized by the following extensive parameters: internal energy U, volume V, and the amount n_j (in mole) of each substance.

(ii) For all equilibrium states there exists a real valued function of the extensive parameters (called entropy S), that is continuous, differentiable, additive over the constituent subsystems, and strictly monotonically increasing with the energy.

(iii) The values assumed by the extensive parameters are those that maximize the entropy over the configuration space (manifold of equilibrium states).

The strict monotonic property of the entropy function implies that it can be inverted with respect to the energy and, by postulate (ii), the energy is a single-valued, continuous and differentiable function of S, V, and each $n_1,\ldots, n_r$ (simplify $n_{1\ldots r}$). Both $S = S(U, V, n_{1\ldots r})$ and $U = U(S, V, n_{1\ldots r})$ are homogeneous first-order functions that contain all thermodynamic information about a system and each one is known as the *fundamental equation*.

The partial derivatives of U are called *intensive parameters* and are zero-order homogeneous functions with a concrete meaning in thermodynamics:

$$\begin{aligned} \partial_S U &= T(S, V, n_{1\ldots r}) \equiv \text{temperature}, \\ -\partial_V U &= p(S, V, n_{1\ldots r}) \equiv \text{pressure}, \\ \partial_{n_j} U\big|_{1\le j\le r} &= \mu_j(S, V, n_{1\ldots r}) \equiv \text{chemical potential}. \end{aligned} \tag{1}$$

This complete set of equations (called *equations of state*) is equivalent to the fundamental equation, so it contains all thermodynamic information about a system.

Each curve that takes values in the configuration space is a dense sequence of equilibrium states that lacks information about the dynamics of state change. However, a real thermodynamic process is a sequence of equilibrium and non-equilibrium states, but no general theory of non-equilibrium thermodynamics is currently available. Attempts to extend equilibrium thermodynamics yielded the elaboration of the Classical Irreversible Thermodynamics (CIT) in the first half of the 20th century [12], [36], [37], [43]. Subsequent efforts to overcome the limitations of the CIT have resulted in formulations such as: Rational Thermodynamics (RT) [47] which has contributed to the development of continuous media thermodynamics; the Extended Irreversible Thermodynamics (EIT) [23] which provides a bridge between CIT and RT; and the formalism known as GENERIC (General Equation for the Non-Equilibrium Reversible–Irreversible Coupling) [39] which develops a non-equilibrium thermodynamics with Hamiltonian structure. Nowadays, more efforts to model non-equilibrium thermodynamic processes join the program of geometrization in physics and employ the language of differential geometry to address an invariant (coordinate-free) formulation of phenomenological thermodynamics [51].

In this paper, we have chosen to use CIT because it fits easily into the port-Hamiltonian approach and it has proved to be suitable for modeling a large number of non-equilibrium thermodynamic processes, as confirmed by experimental results. The main additional postulate of CIT is the *local-equilibrium hypothesis*: "small volume elements of a system may be considered to be in thermodynamic equilibrium locally, although the whole system may not be so, and the same equilibrium postulates (i–iii) can be applied to those local volumes" [48, Chap.23].

One can use this approach to approximate continuous systems models using networks of lumped parameter subsystems (also called networks of control volumes [38], [41]). Thus, if we call $\mathbb{R}_+ = \{t \in \mathbb{R} : t > 0\}$, we consider the smooth manifold canonical structure of $\mathbb{R}_+^{2+r}$, the thermodynamic

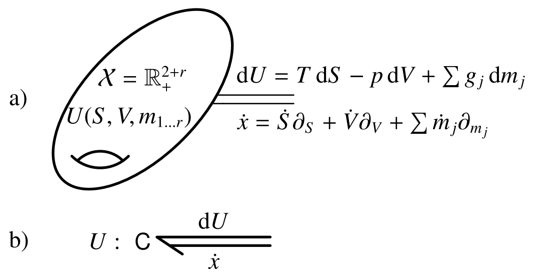

a)
$\mathcal{X} = \mathbb{R}^{2+r}_{+}$
$U(S, V, m_{1...r})$
$\mathrm{d}U = T\,\mathrm{d}S - p\,\mathrm{d}V + \sum g_j\,\mathrm{d}m_j$
$\dot{x} = \dot{S}\partial_S + \dot{V}\partial_V + \sum \dot{m}_j\partial_{m_j}$
b)
$U :$ C
$\mathrm{d}U$
$\dot{x}$

FIGURE 1. **a) Storage element that represent a simple thermodynamic system and b) bond graph representation (see [28, Figure 7]).**

state of a control volume $x = (S, V, n_{1...r}) \in \mathbb{R}^{2+r}_{+}$, and the energy state function $U : \mathbb{R}^{2+r}_{+} \to \mathbb{R}$, then we can write the following energy flow rate for thermodynamic systems [12, p.23] (see [28, Appendix] for a summary on vector bundles and braket notation)

$$\begin{aligned}\dot{U} &= \langle \mathrm{d}U \mid \dot{x} \rangle \\ &= \left\langle T\mathrm{d}S - p\mathrm{d}V + \sum \mu_j \mathrm{d}n_j \,\middle|\, \dot{S}\partial_S + \dot{V}\partial_V + \sum \dot{n}_i \partial_{n_i} \right\rangle \\ &= T\dot{S}\,\langle \mathrm{d}S \mid \partial_S \rangle - p\dot{V}\,\langle \mathrm{d}V \mid \partial_V \rangle + \sum \mu_j \dot{n}_i \left\langle \mathrm{d}n_j \,\middle|\, \partial_{n_i} \right\rangle \\ &= T\dot{S} - p\dot{V} + \sum_{j\in 1...r} \mu_j \dot{n}_j, \qquad (2)\end{aligned}$$

that corresponds to a storage element $\mathsf{C} = (\mathbb{R}^{2+r}_{+}, U)$ (see [28, Definition 23]) whose state space is the smooth manifold $\mathbb{R}^{2+r}_{+}$, and with Hamiltonian $U(S, V, n_{1...r})$. The power port of C (see [28, Definition 1]) is $P = T\mathbb{R}^{2+r}_{+} \oplus T^{*}\mathbb{R}^{2+r}_{+}$, the flow variable is the vector field

$$\dot{x} = \dot{S}\partial_S + \dot{V}\partial_V + \sum_{j\in 1...r} \dot{n}_j \partial_{n_j} \in T\mathbb{R}^{2+r}_{+}, \qquad (3)$$

and the effort variable is the covector field

$$\mathrm{d}U = T\mathrm{d}S - p\mathrm{d}V + \sum_{j\in 1...r} \mu_j \mathrm{d}n_j \in T^{*}\mathbb{R}^{2+r}_{+}. \qquad (4)$$

If we express the fundamental equation as a function of the mass $m_j = n_j \cdot M_{\mathrm{mol},j}$ of each chemical species, where $M_{\mathrm{mol},j}$ kg/mol is the molar mass of the substance j, then $U = (S, V, m_{1...r})$ and

$$\dot{U} = T\dot{S} - p\dot{V} + \sum_{j\in 1...r} g_j \dot{m}_j, \qquad (5)$$

$$g_j = \frac{\partial U}{\partial m_j} = \frac{\partial U}{\partial n_j}\frac{\partial n_j}{\partial m_j} = \frac{\mu_j}{M_{\mathrm{mol},j}}. \qquad (6)$$

Figure 1 is the thermodynamic domain adaptation of [28, Figure 7]. Part a) represents the storage element of a simple thermodynamic system defined by the smooth manifold $\mathcal{X}$, Hamiltonian U, and power port $P = \langle \mathrm{d}U | \dot{x} \rangle$. Part b) is the bond graph representation of this storage element.

Since energy and entropy are homogeneous first-order functions, they can be scaled from their definition for a unitary mass system:

$$\begin{aligned} U(S, V, m_{1...r}) &= mU\,(S/m, V/m, m_{1...r}/m) \\ &= mu(s, v, w_{1...r}), \qquad (7)\end{aligned}$$

$$m = \sum_{j\in 1...r} m_j; \quad \sum_{j\in 1...r} w_j = 1, \qquad (8)$$

where $u = U/m$, $s = S/m$, and $v = V/m$ are called *specific* (or *massic* [46, Section 8.9]) energy, specific entropy, and specific volume respectively, and $w_j = m_j/m$ is called *mass fraction*. As intensive parameters are zero-order homogeneous functions, they have the same form when expressed with specific variables, i.e.,

$$\begin{aligned} T(S, V, m_{1...r}) &= T(s, v, w_{1...r}) \\ p(S, V, m_{1...r}) &= p(s, v, w_{1...r}) \\ g_j(S, V, m_{1...r}) &= g_j(s, v, w_{1...r}).\end{aligned}$$

By applying Euler's theorem[1] to the first-order homogeneous function U we obtain the Euler's relation [7, p.51]

$$U = TS - pV + \sum_{j\in 1...r} g_j m_j, \qquad (9)$$

$$\begin{aligned} u &= T\frac{S}{m} - p\frac{V}{m} + \sum_{j\in 1...r} g_j \frac{m_j}{m} \\ &= Ts - pv + \sum_{j\in 1...r} g_j w_j, \qquad (10)\end{aligned}$$

that in the entropy representation has the form

$$S = \left(\frac{1}{T}\right) U + \left(\frac{p}{T}\right) V - \sum_{j\in 1...r} \left(\frac{g_j}{T}\right) m_j, \qquad (11)$$

$$\partial_U S = 1/T, \quad \partial_V S = p/T, \quad \partial_{m_j} S = g_j/T. \qquad (12)$$

Single-substance thermofluid systems have a special interest in thermal engineering to design energy transport and storage systems. In this case, the fundamental equation, the equations of the intensive parameters, and the equation of the energy flow rate take a simple form when expressed with specific quantities:

$$\begin{aligned} u(s, v) &= \frac{1}{m} U(S, V, m) = U(S/m, V/m, 1) \\ &= T\frac{S}{m} - p\frac{V}{m} + g\frac{m}{m} \\ &= Ts - pv + g, \qquad (13) \\ T(s, v) &= \partial u/\partial s, \qquad (14) \\ p(s, v) &= -\partial u/\partial v, \qquad (15) \\ \dot{u} &= \langle \mathrm{d}u | \dot{x} \rangle \\ &= \langle T\mathrm{d}s - p\mathrm{d}v \mid \dot{s}\partial_s + \dot{v}\partial_v \rangle = T\dot{s} - p\dot{v}. \qquad (16)\end{aligned}$$

B. THERMODYNAMIC POTENTIALS AND THERMODYNAMIC PROPERTIES

Although the fundamental equation (in either of its two representations: energy or entropy) contains all the information of

[1]If $f(x_{1...r})$ is a homogeneous functions of degree k and has continuous first partial derivatives, then:

$$\sum_{j\in 1...r} \frac{\partial f}{\partial x_j} x_j = kf(x_{1...j}).$$

a thermodynamic system, its theoretical deduction is beyond the scope of thermodynamics and corresponds to statistical mechanics [27], [7, Part II]. Experimentally, however, some approximations to the fundamental equation or its differential can be obtained.

When experimenting with a thermodynamic system, it has been found that some variables can be more easily measured than others. Thus, for example, there are no instruments to measure the entropy (extensive variable) of a system. However, the temperature (intensive variable) can be easily measured with a thermometer. It would thus be convenient to be able to formulate the fundamental equation taking as independent variables those extensive and intensive variables that can be easily measured. The appropriate tool to perform this alternative formulation without loss of information is the *Legendre transformation* [2].

Let $y(x_{1\ldots\ell\ldots n})$ be a function with partial derivatives $p_k = \partial y/\partial x_k$. The Legendre transformation of y with respect to $x_{1\ldots\ell}$ is a function of the independent variables $p_{1\ldots\ell}, x_{\ell+1\ldots n}$, denoted $y[p_{1\ldots\ell}]$ and defined

$$y[p_{1\ldots\ell}] = y - \sum_{k=1}^{\ell} p_k x_k. \tag{17}$$

Note that the sign criterion used in thermodynamics for the Legendre transformation is the opposite of the one used in mechanics.

1) THERMODYNAMIC POTENTIALS

The Legendre transforms of the fundamental equation U are also called *thermodynamic potentials* (by analogy with the mechanical potential energy) and some of the most used are [2]:

- *Helmholtz free energy* F (or *Helmholtz potential*):

$$\begin{aligned} F(T,V,m_{1\ldots r}) &= U[T] = U - TS, \qquad (18)\\ \mathrm{d}F &= \mathrm{d}(U-TS)\\ &= \mathrm{d}U - T\mathrm{d}S - S\mathrm{d}T\\ &= T\mathrm{d}S - p\mathrm{d}V + \sum_{j\in 1\ldots r} g_j \mathrm{d}m_j\\ &\quad - T\mathrm{d}S - S\mathrm{d}T\\ &= -S\mathrm{d}T - p\mathrm{d}V + \sum_{j\in 1\ldots r} g_j\mathrm{d}m_j, \end{aligned}$$

$$\text{for single substance} \begin{cases} f = u - Ts,\\ \mathrm{d}f = -s\mathrm{d}T - p\mathrm{d}v,\\ \dot f = -s\dot T - p\dot v. \end{cases} \tag{19}$$

- *Enthalpy* H:

$$\begin{aligned} H(S,p,m_{1\ldots r}) &= U[-p] = U + pV, \qquad (20)\\ \mathrm{d}H &= T\mathrm{d}S + V\mathrm{d}p + \sum_{j\in 1\ldots r} g_j\mathrm{d}m_j, \end{aligned}$$

$$\text{for single substance} \begin{cases} h = u + pv,\\ \mathrm{d}h = T\mathrm{d}s + p\mathrm{d}v,\\ \dot h = T\dot s + v\dot p. \end{cases} \tag{21}$$

- *Gibbs free energy* G (*Gibbs potential*, or *free enthalpy*):

$$\begin{aligned} G(T,p,m_{1\ldots r}) &= U[T,-p] = U - TS + pV, \qquad (22)\\ \mathrm{d}G &= -S\mathrm{d}T + V\mathrm{d}p + \sum_{j\in 1\ldots r} g_j\mathrm{d}m_j, \end{aligned}$$

$$\text{for single substance} \begin{cases} g = u - Ts + pv,\\ \mathrm{d}g = -s\mathrm{d} + v\mathrm{d}p,\\ \dot g = -s\dot T + v\dot p. \end{cases} \tag{23}$$

It is important to note that the total Legendre transform $U[T,-p,g_{1\ldots r}]$ is identically zero, leading to the *Gibbs-Duhem equation* (24) which establishes a relation between the intensive parameters of a simple system. Indeed,

$$\begin{aligned} &\mathrm{d}\left(U - TS + pV - \sum_{j\in 1\ldots r} g_j m_j\right) = 0, \text{ (by Eq.(9))},\\ &\mathrm{d}U - T\mathrm{d}S + p\mathrm{d}V - \sum_{j\in 1\ldots r} g_j\mathrm{d}m_j\\ &\quad -S\mathrm{d}T + V\mathrm{d}p - \sum_{j\in 1\ldots r} m_j\mathrm{d}g_j = 0,\\ &\Rightarrow S\mathrm{d}T - V\mathrm{d}p + \sum_{j\in 1\ldots r} m_j\mathrm{d}g_j = 0, \qquad (24) \end{aligned}$$

and, although systems with r substances have $r+2$ intensive parameters, only $r+1$ of them are independent. It is said that those systems have $r+1$ *thermodynamics degree of freedom*.

2) PRIMARY SET OF RESPONSE FUNCTIONS

Just as the first partial derivatives of the fundamental equation have an important physical significance (temperature, pressure, and chemical potential), the second partial derivatives for single substances, called *response functions* [48, Chap.10], are also of great physical interest because they characterize the *thermodynamic properties* of these substances (e.g., Syltherm 800, see Example 5).

The *primary set* of response functions is defined from the second-order derivatives of the Gibbs free energy G [26, Chap.1]:

- Specific *isobaric heat capacity* or specific heat capacity at constant pressure

$$\begin{aligned} c_p &= -\frac{T}{m}\left.\frac{\partial^2 G}{\partial T^2}\right|_p = \frac{T}{m}\left.\frac{\partial S}{\partial T}\right|_p\\ &= T\left.\frac{\partial s}{\partial T}\right|_p = \left.\frac{\partial h}{\partial T}\right|_p \quad \mathrm{J/(kg{\cdot}K)}, \end{aligned} \tag{25}$$

is the heat flux per unit mass required to produce a unit increase in the temperature of a system maintained at constant pressure.[2]

- *Isothermal compressibility*

$$\begin{aligned} \kappa_T &= -\frac{1}{V}\left.\frac{\partial^2 G}{\partial p^2}\right|_T\\ &= -\frac{1}{V}\left.\frac{\partial V}{\partial p}\right|_T = -\frac{1}{v}\left.\frac{\partial v}{\partial p}\right|_T \quad \mathrm{Pa}^{-1}, \end{aligned} \tag{26}$$

[2]Remember that $\Delta S = \Delta Q/T$.

is the fractional decrease in volume per unit increase in pressure of a system maintained at constant temperature. The inverse of κ_T is called *isothermal bulk modulus* $\beta_T = 1/\kappa_T$.

- *Isobaric expansibity*

$$\begin{aligned}\alpha_p &= \frac{1}{V}\frac{\partial}{\partial T}\bigg|_p \frac{\partial G}{\partial p}\bigg|_T \\ &= \frac{1}{V}\frac{\partial V}{\partial T}\bigg|_p = \frac{1}{v}\frac{\partial v}{\partial T}\bigg|_p \quad \mathrm{K}^{-1}, \qquad (27)\end{aligned}$$

is the fractional increase in volume per unit increase in temperature of a system maintained at constant pressure.

3) FUNDAMENTAL SET OF RESPONSE FUNCTIONS

The *fundamental set* of response functions is defined from the second-order derivatives of the internal energy U [48, Chap.10]:

- Specific *isochoric heat capacity* or specific heat capacity at constant volume

$$\begin{aligned}c_v &= \frac{T}{m}\left(\frac{\partial^2 U}{\partial S^2}\bigg|_V\right)^{-1} = \frac{T}{m}\frac{\partial S}{\partial T}\bigg|_V \\ &= T\frac{\partial s}{\partial T}\bigg|_v = \frac{\partial u}{\partial T}\bigg|_v \quad \mathrm{J/(kg{\cdot}K)}, \qquad (28)\end{aligned}$$

is the heat flux per unit mass required to produce a unit increase in the temperature of a system maintained at constant volume.

- *Adiabatic compressibility*

$$\begin{aligned}\kappa_s &= \frac{1}{V}\left(\frac{\partial^2 U}{\partial V^2}\bigg|_S\right)^{-1} \\ &= -\frac{1}{V}\frac{\partial V}{\partial p}\bigg|_S = -\frac{1}{v}\frac{\partial v}{\partial p}\bigg|_s \quad \mathrm{Pa}^{-1}, \qquad (29)\end{aligned}$$

is the fractional decrease in volume per unit increase in pressure of a system maintained at constant entropy.

- *Adiabatic expansibity*

$$\begin{aligned}\alpha_s &= \frac{1}{V}\left(\frac{\partial}{\partial V}\bigg|_S \frac{\partial U}{\partial S}\bigg|_V\right)^{-1} \\ &= \frac{1}{V}\frac{\partial V}{\partial T}\bigg|_S = \frac{1}{v}\frac{\partial v}{\partial T}\bigg|_s \quad \mathrm{K}^{-1}, \qquad (30)\end{aligned}$$

is the fractional increase in volume per unit increase in temperature of a system maintained at constant entropy.

Some useful expressions relating the fundamental and primary sets of response functions are as follows:

$$\begin{aligned}\gamma &= c_p/c_v \ (\textit{heat capacity ratio}), \qquad (31)\\ &= \kappa_T/\kappa_s \\ &= 1 - \alpha_p/\alpha_s, \\ c_p - c_v &= T v \alpha_p^2/\kappa_T, \qquad (32)\\ \kappa_T - \kappa_s &= T v \alpha_p^2/c_p. \qquad (33)\end{aligned}$$

III. ISO REPRESENTATION OF DISSIPATIVE PORT-HAMILTONIAN SYSTEMS

In this section, we will use the input-state-output (ISO) representation of dissipative port-Hamiltonian systems to compare the port-Hamiltonian and metriplectic formalisms [29], [33] introduced to extend the Hamiltonian formalism to dissipative systems. Here H stands for Hamiltonian, not for enthalpy. The kernel representation of the port-Hamiltonian system $(\mathcal{X}, \mathsf{F}, \mathcal{D}, H)$ is (see [28, Definition 6 and Remark 12])

$$F(-\dot{x} \oplus f) + E(\mathrm{d}H \oplus e) = 0, \qquad (34)$$

that yields the following local matrix equation

$$F(x)\begin{bmatrix}-\dot{x}\\ f\end{bmatrix} + E(x)\begin{bmatrix}\partial_x H\\ e\end{bmatrix} = 0, \qquad (35)$$

where $\dot{x} = [\dot{x}^1, \ldots, \dot{x}^n]^\intercal$ and $\partial_x H = [\frac{\partial H}{\partial x^1}, \ldots, \frac{\partial H}{\partial x^n}]^\intercal$ are the matrix representation of $\dot{x}$ and $\mathrm{d}H$ respectively.

Provided the matrices F and E can be split as

$$\begin{aligned}F &= [F_n|F_m], \quad E = [E_n|E_m] \\ &\qquad F_n, E_n \in \mathbb{R}^{(n+m)\times n}, \quad F_m, E_m \in \mathbb{R}^{(n+m)\times m},\end{aligned}$$

with rank$F_n = n$ and rank$[F_n|E_m] = n+m$, Equation 35 can be written

$$F'\begin{bmatrix}-\dot{x}\\ e\end{bmatrix} + E'\begin{bmatrix}\partial_x H\\ f\end{bmatrix} = 0,$$

where $F' = [F_n|E_m]$ and $E' = [E_n|F_m]$. Now F' is invertible, since rank $F' = n+m$; and (35) can be explicitly solve for $\dot{x}$

$$\begin{bmatrix}-\dot{x}\\ e\end{bmatrix} = F'^{-1}E'\begin{bmatrix}\partial_x H\\ f\end{bmatrix}.$$

Since $F'E'^\intercal + E'F'^\intercal = 0$, $F'^{-1}E'$ is skew-symmetric and can be written

$$F'^{-1}E' = \begin{bmatrix}J & g\\ -g^\intercal & B\end{bmatrix},$$

with J and B skew-symmetric matrices. If there is not a direct f to e coupling, then $B = 0$ and the explicit equations that model the behavior of a port-Hamiltonian system are

$$\begin{aligned}\dot{x} &= J(x)\partial_x H + g(x)f\\ e &= g^\intercal(x)\partial_x H.\end{aligned}$$

If we split $\mathsf{F}\oplus\mathsf{E}$ into an open input-output port $\mathsf{U}\oplus\mathsf{Y}$ (with $\mathsf{Y} = \mathsf{U}^*$) and a resistive port $\mathsf{F}_R \oplus \mathsf{E}_R$ (energy-dissipating port, see Figure 2), the matrix $F'^{-1}E$, and the equations of the system can be written

$$F'^{-1}E' = \begin{bmatrix}J & g_R & g\\ -g_R^\intercal & 0 & 0\\ -g^\intercal & 0 & 0\end{bmatrix}, \quad \begin{cases}\dot{x} = J\partial_x H + g_R f_R + gu\\ e_R = g_R^\intercal \partial_x H\\ y = g^\intercal \partial_x H.\end{cases}$$

If the resistive port is defined by the linear relation $f_R = -R' e_R$ (with R' symmetric and positive definite), then $f_R = -R' g_R^\intercal \partial_x H$ and

$$\begin{aligned}\dot{x} &= [J(x) - R(x)]\partial_x H + g(x)u\\ y &= g^\intercal(x)\partial_x H,\end{aligned}$$

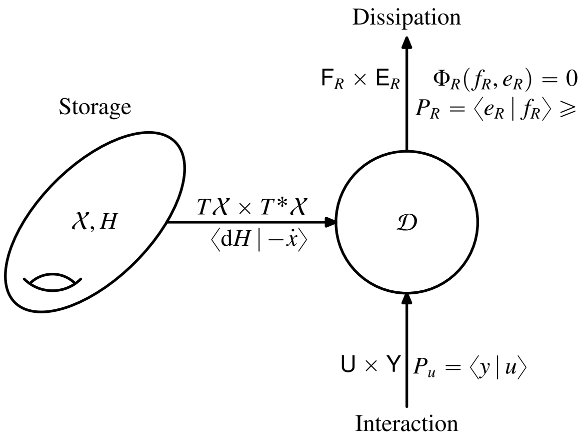

Dissipation
Storage
$\mathsf{F}_R \times \mathsf{E}_R$
$\Phi_R(f_R, e_R) = 0$
$P_R = \langle e_R \,|\, f_R \rangle \geqslant 0$
$\mathcal{X}, H$
$T\mathcal{X} \times T^*\mathcal{X}$
$\langle \mathrm{d}H \,|\, -\dot{x} \rangle$
$\mathcal{D}$
$\mathsf{U} \times \mathsf{Y}$
$P_u = \langle y \,|\, u \rangle$
Interaction

FIGURE 2. Input-state-output port-Hamiltonian system with energy dissipation.

where $R(x) = g_R(x)R'(x)g_R^{\mathsf{T}}(x)$ is a symmetric positive semidefinite matrix.

The previous equations let us introduce the following geometric and coordinate-free definition of an ISO port-Hamiltonian system:

Definition 1: An ISO port-Hamiltonian system is a dynamical system defined by the 6-tuple $(\mathcal{X}, H, J, R, \mathsf{U}, g)$ where:

1) The smooth manifold $\mathcal{X}$ is the state space.
2) The Hamiltonian smooth function $H : \mathcal{X} \to \mathbb{R}$ models the energy of the system.
3) The skew-symmetric bundle morphism $J : T^*\mathcal{X} \to T\mathcal{X}$ represents the internal power-preserving interconnections. The isomorphism $\mathrm{Hom}(T^*\mathcal{X}, T\mathcal{X}) \cong \mathrm{Hom}(T^*\mathcal{X} \oplus T\mathcal{X}, \mathbb{R})$ allows to identify J with a skew-symmetric bilinear form (or antisymmetric order 2 tensor).
4) The symmetric positive semidefinite bundle morphism $R : T^*\mathcal{X} \to T\mathcal{X}$ represents the dissipation of the system. The isomorphism $\mathrm{Hom}(T^*\mathcal{X}, T\mathcal{X}) \cong \mathrm{Hom}(T^*\mathcal{X} \oplus T\mathcal{X}, \mathbb{R})$ allows to identify R with a symmetric positive semidefinite bilinear form (or symmetric positive semidefinite order 2 tensor).
5) The input power variable u is a constant section of the trivial vector bundle $\mathcal{X} \oplus \mathsf{U}$, i.e., u depends on the time t but not on the state x.
6) The output power variable y is a constant section of the trivial vector bundle $\mathcal{X} \oplus \mathsf{Y} = \mathcal{X} \oplus \mathsf{U}^*$.
7) The bundle morphism $g : \mathcal{X} \oplus \mathsf{U} \to T\mathcal{X}$ describes the distribution of the power incoming into the system from the environment. Its dual morphism is $g^* : T^*\mathcal{X} \to \mathcal{X} \oplus \mathsf{Y}$.
8) The behavior of the system is represented by the equations:

$$
\begin{aligned}
|\dot{x}\rangle &= (J - R)\,|\mathrm{d}H\rangle + g\,|u\rangle \\
|y\rangle &= g^*\,|\mathrm{d}H\rangle \quad \text{or} \quad \langle y| = \langle \mathrm{d}H|\, g. \qquad (36)
\end{aligned}
$$

Since J is a skew-symmetric morphism then $\langle \mathrm{d}H \,|\, J \,|\, \mathrm{d}H \rangle = 0$, and the power balance of an ISO port-Hamiltonian system is

$$
\begin{aligned}
\dot{H} &= \langle \mathrm{d}H \,|\, \dot{x} \rangle \\
&= \langle \mathrm{d}H \,|\, (J - R) \,|\, \mathrm{d}H \rangle + \langle \mathrm{d}H \,|\, g \,|\, u \rangle \\
&= \langle \mathrm{d}H \,|\, J \,|\, \mathrm{d}H \rangle - \langle \mathrm{d}H \,|\, R \,|\, \mathrm{d}H \rangle + \langle y \,|\, u \rangle \\
&= \langle y \,|\, u \rangle - \langle \mathrm{d}H \,|\, R \,|\, \mathrm{d}H \rangle = P_u - P_R.
\end{aligned}
$$

To relate the dissipative port-Hamiltonian systems to the metriplectic formalism we split the Hamiltonian into two summands $H = H_c - H_d$, with the additional requirements (known as degeneracy constraints [39, Eqs.1.4-5]):

$$
R\,|\mathrm{d}H_c\rangle = 0, \qquad (37)
$$
$$
J\,|\mathrm{d}H_d\rangle = 0. \qquad (38)
$$

The dynamics of a closed system is then ([33, Eq.25], [29, Eq.18])

$$
|\dot{x}\rangle = J\,|\mathrm{d}H_c\rangle + R\,|\mathrm{d}H_d\rangle\,.
$$

The first term corresponds to the symplectic structure and the second one to the metric structure (so the name *metriplectic system*). In addition, the following properties are fulfilled:

$$
\begin{aligned}
\dot{H}_c &= \langle \mathrm{d}H_c \,|\, \dot{x} \rangle = 0, \\
\dot{H}_d &= \langle \mathrm{d}H_d \,|\, \dot{x} \rangle = \langle \mathrm{d}H_d \,|\, R \,|\, \mathrm{d}H_d \rangle \geq 0, \\
\dot{H} &= \langle \mathrm{d}H \,|\, \dot{x} \rangle = -\langle \mathrm{d}H_d \,|\, R \,|\, \mathrm{d}H_d \rangle = -\dot{H}_d.
\end{aligned}
$$

The above equations mean that H_c is the conserved energy ($\dot{H}_c = 0$), H_d is the dissipated energy, and $\dot{H}_d \geq 0$ implies that dissipation is an irreversible process (the metriplectic formalism mentions this term as an *entropy-like* function). For closed thermodynamic systems we can identify $H = F$ (Helmholtz free energy, see Equation 18), $H_c = U$ (internal energy), $H_d = TS$, and write

$$
\begin{aligned}
\dot{H}_d &= \langle \mathrm{d}(TS) \,|\, \dot{x} \rangle \\
&= T\,\langle \mathrm{d}S \,|\, \dot{x} \rangle + \langle S\mathrm{d}T \,|\, \dot{x} \rangle \\
&= T\,\langle \mathrm{d}S \,|\, \dot{x} \rangle + \langle S\mathrm{d}T \,|\, (J - R) \,|\, S\mathrm{d}T \rangle \\
&= T\,\langle \mathrm{d}S \,|\, \dot{x} \rangle - S^2\,\langle \mathrm{d}T \,|\, R \,|\, \mathrm{d}T \rangle \geq 0,
\end{aligned}
$$

i.e., $\dot{S} = \langle \mathrm{d}S \,|\, \dot{x} \rangle \geq S^2/T\,\langle \mathrm{d}T \,|\, R \,|\, \mathrm{d}T \rangle \geq 0$, which means an increase of entropy in an irreversible process.

IV. MODELICA IMPLEMENTATION OF SIMPLE THERMODYNAMIC SYSTEMS

In this section, we take a further step in the development of the Modelica library pHlib started in [28]. This library consists of two main packages: Dirac and pHS. In the package Dirac, we modeled the following concepts: power port, bond, general Dirac structure, and junction structures (0-junctions, 1-junctions, transformers TF, and gyrators GY). With these elements, one can define the non-dissipative energy exchange structure of a complex system by connecting elementary junction structures. In the package pHS, we modeled the following concepts: flow and effort sources (Sf, Se), storage element (C), and resistors (R, RS). These elements are used to model

```
import SI = Modelica.Units.SI;
import Dirac.Interfaces.Ports.*;
package Ports
  connector ThermalPort extends Port(dimPort= 1,
    redeclare SI.Temperature e, redeclare SI.EntropyFlowRate f);
  end ThermalPort;

  connector PressurePort extends Port(dimPort= 1,
    redeclare SI.Pressure e, redeclare SI.VolumeFlowRate f);
  end PressurePort;

  connector MassPort extends Port(dimPort= 1,
    redeclare SI.SpecificGibbsFreeEnergy e,
    redeclare SI.MassFlowRate f);
  end MassPort;

  connector ThermodynamicPort extends Port(dimPort = 3);
  end ThermodynamicPort;
end Ports
```

Listing 1. Thermodynamics.Ports.

the energy exchanged with the environment, the energy stored in the system, and the dissipation of energy.

In this article, we take full advantage of the fact that Modelica is an object-oriented language to extend the pHlib library to the thermodynamics domain. To do so, we implement a new package called Thermodynamics by reusing the classes defined in the packages Dirac and pHS.

A. POWER PORTS AND BONDS

In the remainder of this paper, "thermodynamic system" will refer to a simple single-substance thermodynamic system unless otherwise stated.

We remember that, for the smooth manifold $\mathcal{X}$ with tangent bundle $T\mathcal{X}$ and cotangent bundle $T^*\mathcal{X}$, the fiber product $P = T\mathcal{X} \oplus T^*\mathcal{X}$ is a *power port* if the dual product $\langle\omega|X\rangle$ —where the effort $\omega \in T^*\mathcal{X}$ is a covector field and the flow $X \in T\mathcal{X}$ is a vector field—, has physical dimensions of power [28, Definition 1]. This definition applies to thermodynamic systems, where:

$$\begin{aligned}\mathcal{X} &= \mathbb{R}^3_+,\\ \dot{x} &= \dot{S}\partial_S + \dot{V}\partial_V + \dot{m}\partial_m \in T\mathbb{R}^3_+,\\ dU &= T\mathrm{d}S - p\mathrm{d}V + g\mathrm{d}m \in T^*\mathbb{R}^3_+, \text{ and}\\ \dot{U} &= \langle \mathrm{d}U|\dot{x}\rangle\,.\end{aligned}$$

Listing 1 shows the Modelica implementation of thermodynamic ports.

As we can see, the effort and flow variables of type Real that appear in the general definition of the connector Dirac.Interfaces.Ports.Port have been *redeclared* with units of the International System (SI), so they have meaning in the thermodynamic domain and can be checked for dimensional consistency.

Listing 2 shows the extension of the class Bond (used to connect subsystem ports in a complex system) to the thermodynamic domain.

It should be noted that, in defining new bonds for the thermodynamic domain, we have extended the base types defined in Dirac.Interfaces.Bonds to include variables with names associated with each type of bond. These variables

```
import Dirac.Interfaces.Bonds.*;
package Bonds
  class ThermalBond
    SI.Temperature T = from.e[1];
    SI.EntropyFlowRate dot_S = from.f[1];
    extends OneDimensionalBond(
      redeclare connector BasePort = ThermalPort);
  end ThermalBond;

  class PressureBond
    SI.Pressure p = −from.e[1];
    SI.VolumeFlowRate dot_V = from.f[1];
    extends OneDimensionalBond(
      redeclare connector BasePort = PressurePort);
  end PressureBond;

  class MassBond
    SI.SpecificGibbsFreeEnergy g = from.e[1];
    SI.MassFlowRate dot_m = from.f[1];
    extends OneDimensionalBond(
      redeclare connector BasePort = MassPort);
  end MassBond;

  class ThermodynamicBond
    SI.Temperature T;
    SI.Pressure p;
    SI.SpecificGibbsFreeEnergy g;
    SI.EntropyFlowRate dot_S;
    SI.VolumeFlowRate dot_V;
    SI.MassFlowRate dot_m;
    //
    extends ThreeDimensionalBond(
      redeclare replaceable connector BasePort = ThermodynamicPort);
    //
  equation
    // Efforts
    {T,−p,g} = from.e[1:3];
    // Flows
    {dot_S,dot_V,dot_m} = from.f[1:3];
  end ThermodynamicBond;
end Bonds;
```

Listing 2. Thermodynamics.Bonds.

facilitate access and visualization of the values transferred by the bond.

In Figure 3 we can see the symbols used for a graphical representation of bonds and other elements defined in the package Thermodynamics.

B. SOURCES

The sources of flow $(\dot{S}, \dot{V}, \dot{m})$ and effort (T, p, g) have been derived from the general ones defined in pHS, and redeclared according to their physical meaning. Listing 3 shows the implementation of modulated and constant sources.

C. STORAGE ELEMENTS

The class pHS.Storage.C is the base class for defining storage elements. The application of [28, Definition 23] to thermodynamic systems is as follows:

Definition 2: A *thermodynamic storage element* is a 1-port element defined by the pair $(\mathbb{R}^3_+, U)$ where $\mathbb{R}^3_+$ has the structure of a smooth manifold that models the state variables $x = (S, V, m)$, and $U \in C^\infty(\mathbb{R}^3_+)$ is the Hamiltonian that models the system energy. The flow and effort variables of the port are the vector field $\dot{x} \in T\mathbb{R}^3_+$ and the covector field $\mathrm{d}U \in T^*\mathbb{R}^3_+$.

Listing 4 shows the implementation of the class InternalEnergy which derives from pHS.Storage.C according to Definition 2.

```
package Sources
  package Modulated
    class Temperature "Source of temperature"
      replaceable SI.Temperature T;
      extends pHS.Sources.Modulated.MSe(
        redeclare SI.Temperature e, redeclare ThermalPort port);
    equation
      e = {T};
    end Temperature;

    class EntropyFlowRate "Source of entropy flow rate"
      replaceable SI.EntropyFlowRate dot_S;
      extends pHS.Sources.Modulated.MSf(
        redeclare SI.EntropyFlowRate f, redeclare ThermalPort port);
    equation
      f = {dot_S};
    end EntropyFlowRate;

    class Pressure "Source of pressure"
      replaceable SI.Pressure p;
      extends pHS.Sources.Modulated.MSe(
        redeclare SI.Pressure e, redeclare PressurePort port);
    equation
      e = {−p};
    end Pressure;

    class VolumeFlowRate "Source of volume flow rate"
      replaceable SI.VolumeFlowRate dot_V;
      extends pHS.Sources.Modulated.MSf(
        redeclare SI.VolumeFlowRate f, redeclare PressurePort port);
    equation
      f = {dot_V};
    end VolumeFlowRate;

    class SpecificGibbsFreeEnergy "Specific Gibbs free energy source"
      replaceable SI.SpecificGibbsFreeEnergy g;
      extends pHS.Sources.Modulated.MSe(
        redeclare SI.SpecificGibbsFreeEnergy e,
        redeclare MassPort port);
    equation
      e = {g};
    end SpecificGibbsFreeEnergy;

    class MassFlowRate "Source of mass flow rate"
      replaceable SI.MassFlowRate dot_m;
      extends pHS.Sources.Modulated.MSf(
        redeclare SI.MassFlowRate f, redeclare MassPort port);
    equation
      f = {dot_m};
    end MassFlowRate;
  end Modulated;

  package Constant
    class Temperature "Constant source of temperature"
      extends Sources.Modulated.Temperature(
        redeclare replaceable parameter SI.Temperature T);
    end Temperature;

    class EntropyFlowRate "Constant source of entropy flow rate"
      extends Sources.Modulated.EntropyFlowRate(
        redeclare replaceable parameter SI.EntropyFlowRate dot_S);
    end EntropyFlowRate;

    class Pressure "Constant source of pressure"
      extends Sources.Modulated.Pressure(
        redeclare replaceable parameter SI.Pressure p);
    end Pressure;

    class VolumeFlowRate "Constant source of volume flow rate"
      extends Sources.Modulated.VolumeFlowRate(
        redeclare replaceable parameter SI.VolumeFlowRate dot_V);
    end VolumeFlowRate;

    class SpecificGibbsFreeEnergy
      "Constant source of specific Gibbs free energy"
      extends Modulated.SpecificGibbsFreeEnergy(
        redeclare replaceable parameter SI.SpecificGibbsFreeEnergy g);
    end SpecificGibbsFreeEnergy;

    class MassFlowRate "Constant source of mass flow rate"
      extends Sources.Modulated.MassFlowRate(
        redeclare replaceable parameter SI.MassFlowRate dot_m);
    end MassFlowRate;
  end Constant;
end Sources;
```

Listing 3. Thermodynamics.Sources.

The assertions in the equation section are intended to stop the simulation when the storage element volume or mass takes values out of range, i.e., when $V < V_{\min}$ or $m < m_{\min}$.

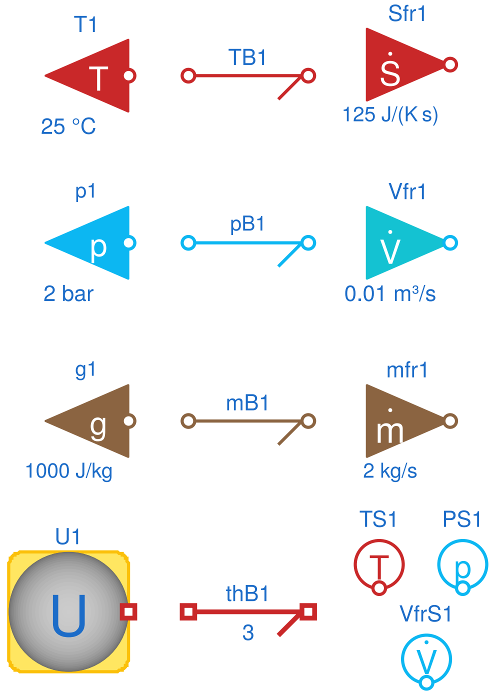


FIGURE 3. Symbols proposed for the package Thermodynamics: T1 (temperature), p1 (pressure), and g1 (specific Gibbs free energy) are sources of effort; Sfr1 (entropy flow rate), Vfr1 (volume flow rate), and mfr1 (mass flow rate) are sources of flow; U1 is a storage element for internal energy; TB1 $\langle T|\dot{S}\rangle$, pB1 $\langle p|\dot{V}\rangle$, and mB1 $\langle g|\dot{m}\rangle$ are one-dimensional bonds; thB1 $\langle (T, p, g)|(\dot{S}, \dot{V}, \dot{m})\rangle$ is a three-dimensional bond; TS1 (temperature), pS1 (pressure), and VfrS1 (volume flow rate) are ideal sensor. These symbols, together with those proposed in [28] for the elementary Dirac structures, facilitate the composition of complex systems (see Figure 5) with the Modelica drag-and-drop graphical user interface.

D. MODELING THE BEHAVIOR OF SINGLE SUBSTANCES

In the implementation of the class InternalEnergy, the instance substance of the abstract class SingleSubstance models the thermodynamic behavior of any simple substance, as described in its fundamental equation. The class InternalEnergy encodes the structure of the storage element from the point of view of the port-Hamiltonian approach, and the class SingleSubstance provides information about the behavior of the particular substance used to define the storage element.

Listing 5 shows the Modelica code used to implement the class SingleSubstance, which is defined in the package Substances.Partial included in the package Thermodynamics. The assertions in the equation section are intended to stop the simulation when the substance temperature, pressure, or density takes values outside the valid range established for the fundamental equation. The security ranges are: $T_{\min} \leq T \leq T_{\max}$, $p_{\min} \leq p \leq p_{\max}$, and $\varrho \geq \varrho_{\min}$.

SingleSubstance is a partial class (or abstract class in the terminology of object-oriented modeling). It is a template that can be used to define derived classes for modeling particular substances or families of substances with common features. In the following, we illustrate the applicability of the developed tools by modeling some ideal thermodynamic systems.

```
class InternalEnergy "Simple thermodynamic storage element"
  extends pHS.Storage.C(
    redeclare Interfaces.Ports.Thermodynamic port,
    x_0={S_0,V_0,m_0});
  replaceable Substances.Partial.SingleSubstance substance(
    T_0 = T_0, rho_0 = m_0/V_0);
  // Initial conditions
  parameter SI.Entropy S_0;
  parameter SI.Volume V_0;
  parameter SI.Mass m_0;
  parameter SI.Temperature T_0 = T_amb;
  // MAX-min values.
  parameter SI.Volume V_min = 0;
  parameter SI.Mass m_min = 0;
  // State variables
  SI.Entropy S(start=S_0) = m*substance.s;
  SI.Volume V(start=V_0);
  SI.Mass m(start=m_0) = V*substance.rho;
  // Efforts
  SI.Temperature T = substance.T;
  SI.Pressure p = substance.p;
  SI.SpecificGibbsFreeEnergy g = substance.g;
  // Internal energy
  SI.InternalEnergy U = substance.u*m;
  SI.HelmholtzFreeEnergy F = substance.f*m;
  SI.Enthalpy H = substance.h*m;
  SI.GibbsFreeEnergy G = substance.g*m;
equation
  // State equations (see [28, Definition 23]).
  // x ∈ 𝒳 = ℝ^3_+.
  x = {S,V,m};
  // dH ∈ T*𝒳.
  // Here H ∈ C^∞(𝒳) stands for Hamiltonian, not for enthalpy.
  dH = {T,−p,g};
  //
  assert(V >= V_min, "ERROR (Storage.InternalEnergy): V="
    +String(V)+" < "+String(V_min), AssertionLevel.error);
  assert(m >= m_min, "ERROR (Storage.InternalEnergy): m="
    +String(m)+" < "+String(m_min), AssertionLevel.error);
end InternalEnergy;
```

Listing 4. Thermodynamics.Storage.InternalEnergy.

```
partial class SingleSubstance
  //
  // Initial conditions.
  replaceable parameter SI.Temperature T_0;
  replaceable parameter SI.Density rho_0;
  //
  // MAX-min values.
  parameter SI.Temperature T_min = 0;
  parameter SI.Temperature T_max = 20000;
  parameter SI.Pressure p_min = 0;
  parameter SI.Pressure p_max = 2000*MPa;
  parameter SI.Density rho_min = 0;
  //
  SI.Temperature T(start = T_0);
  SI.Pressure p;
  SI.SpecificEntropy s;
  SI.Density rho(start = rho_0);
  SI.SpecificVolume v = 1/rho;
  //
  // Internal energy and thermodynamic potentials.
  SI.SpecificInternalEnergy u;
  SI.SpecificHelmholtzFreeEnergy f = u − T*s; // See (19).
  SI.SpecificEnthalpy h = u + p/rho; // See (21).
  SI.SpecificGibbsFreeEnergy g = u − T*s + p/rho; // See (23).
  //
  // Specific isobaric heat capacity J/(kg·K).
  replaceable SI.SpecificHeatCapacity c_p;
equation
  assert(T >= T_min,
    "ERROR (Substances.Partial.SingleSubstance): T="
    +String(T)+" < "+String(T_min), AssertionLevel.error);
  assert(T <= T_max,
    "ERROR (Substances.Partial.SingleSubstance): T="
    +String(T)+" > "+String(T_max), AssertionLevel.error);
  assert(p >= p_min,
    "ERROR (Substances.Partial.SingleSubstance): p="
    +String(p)+" < "+String(p_min), AssertionLevel.error);
  assert(p <= p_max,
    "ERROR (Substances.Partial.SingleSubstance): p="
    +String(p)+" > "+String(p_max), AssertionLevel.error);
  assert(rho >= rho_min,
    "ERROR (Substances.Partial.SingleSubstance): rho="
    +String(rho)+" > "+String(rho_min), AssertionLevel.error);
end SingleSubstance;
```

Listing 5. Thermodynamics.Substances.Partial.SingleSubstance.

E. IDEAL THERMODYNAMIC SYSTEMS

In statistical mechanics, some idealized systems have been defined for which it is possible to calculate their fundamental equation. Although these models may seem too simple, they are sometimes a good first approximation to actual physical systems.

The main characteristic of the fundamental equation of an *ideal* thermodynamic system —in either of its two forms: internal energy or entropy, and expressed in specific quantities— is the separability of its variables [48, Chap.11].

1) IDEAL GASES

For *ideal gases*, the fundamental equation in the specific entropy representation is [6, Appendix D]

$$s = s_0 + \phi(u) + R\ln(v/v_0), \tag{39}$$

where $R = R_{\mathrm{mol}}/M_{\mathrm{mol}}$ J/(kg·K) is the specific gas constant of the substance, $R_{\mathrm{mol}} = 8.314472(15)$ J/(mol·K) is the molar gas constant [31], and (s_0, u_0, v_0) are values of a *reference state*. The function ϕ depends on the characteristics of each gas and satisfies the condition $\phi(u_o) = 0$.

The intensive parameters of an ideal gas are (see (13))

$$\begin{aligned} 1/T &= \partial_u s = \mathrm{d}\phi/\mathrm{d}u \\ \Rightarrow u(T) &= Tc(T) \end{aligned} \tag{40}$$

$$\begin{aligned} \Rightarrow \partial_T u &= \mathrm{d}u/\mathrm{d}T \\ p/T &= \partial_v s = R/v \\ \Rightarrow pv &= RT, \end{aligned} \tag{41}$$

$$\begin{aligned} \Rightarrow h(T) &= Tc(T) + RT \\ \Rightarrow \partial_T h &= \mathrm{d}h/\mathrm{d}T, \\ g/T &= u/T - s + pv/T \\ &= u/T - s_0 - \phi(u) - R\ln(v/v_0) + R. \end{aligned} \tag{42}$$

From (40) and (41), we can deduce some values of the response functions common to all ideal gases. In particular,

$$\alpha_P = \frac{1}{v}\partial_T v|_p = 1/T, \tag{43}$$

$$\kappa_T = -\frac{1}{v}\partial_p v|_T = 1/p, \tag{44}$$

$$\begin{aligned} c_v &= \partial_T u|_v = \mathrm{d}u/\mathrm{d}T \\ \Rightarrow u(T) &= u_0 + \int_{T_0}^{T} c_v(\theta)\mathrm{d}\theta, \end{aligned} \tag{45}$$

$$\begin{aligned} c_p &= \partial_T h|_p = \mathrm{d}h/\mathrm{d}T \\ \Rightarrow h(T) &= h_0 + \int_{T_0}^{T} c_p(\theta)\mathrm{d}\theta \end{aligned} \tag{46}$$

$$c_p - c_v = \partial_T h|_p - \partial_T u|_v = R \text{ [by Eq.(32)]}, \tag{47}$$

```
partial class IdealGas
  extends Partial.SingleSubstance;
  parameter SI.SpecificHeatCapacity R; // Specific gas const. J/(kg·K).
  // Specific isochoric heat capacity c_v J/(kg·K).
  SI.SpecificHeatCapacity c_v = c_p−R; // See (47).
equation
  p*v = R*T; // See (41).
end IdealGas;
```

Listing 6. Thermodynamics.Substances.Partial.IdealGas.

$$\frac{\mathrm{d}\phi}{\mathrm{d}T} = \frac{\mathrm{d}\phi}{\mathrm{d}u}\frac{\mathrm{d}u}{\mathrm{d}T} = \frac{1}{T}c_v$$
$$\Rightarrow \phi(T) = \int_{T_0}^{T} c_v(\theta)\frac{\mathrm{d}\theta}{\theta}. \qquad (48)$$

The Modelica implementation of the ideal gases behavior has resulted in the class IdealGas (see Listing 6), defined in the package Thermodynamics.Substances.Partial. We implement only equations (41) and (47), that are common to all ideal gases and depend on the temperature.

2) PERFECT GASES

For *perfect* gases, $c(T) = c$ is constant, and this implies $c_v = \mathrm{d}(Tc)/\mathrm{d}T = c$ is also constant. This feature allows to obtain explicit expressions for the fundamental equation and the intensive parameters. Indeed,

$$u(T) = c_v T \text{ [by Eq.(40)]} \Rightarrow T(u) = u/c_v, \qquad (49)$$
$$\phi(T) = \int_{T_0}^{T} c_v \frac{\mathrm{d}\theta}{\theta} = c_v \ln(T/T_0) \text{ [by Eq.(48)]},$$
$$\phi(u) = \phi(T(u)) = c_v \ln(u/u_0), \quad u_0 = c_v T_0, \qquad (50)$$
$$s(u,v) = s_0 + c_v \ln(u/u_0) + R\ln(v/v_0) \text{ [Eq.(39)]} \qquad (51)$$
$$u(s,v) = u_0 \mathrm{e}^{(s-s_0)/c_v}(v/v_0)^{-R/c_v}, \qquad (52)$$
$$\begin{aligned} T &= u/c_v \\ &= T_0\mathrm{e}^{(s-s_0)/c_v}(v/v_0)^{-R/c_v} = \partial_s u, \qquad (53) \\ p &= RT/v \\ &= RT_0/v_0\mathrm{e}^{(s-s_0)/c_v}(v/v_0)^{-(R/c_v+1)} \\ &= p_0\mathrm{e}^{(s-s_0)/c_v}(v/v_0)^{-c_p/c_v} = -\partial_v u, \qquad (54) \\ g &= u - Ts + pv \\ &= c_vT - Ts + RT = (c_v + R - s)T \\ &= (c_p - s)T_0\mathrm{e}^{(s-s_0)/c_v}(v/v_0)^{-R/c_v}. \qquad (55) \end{aligned}$$

Remark 3: Another way to obtain (55) is to calculate g from its definition $g = \partial U/\partial m$. The internal energy of an ideal gas is

$$\begin{aligned} U(S,V,m) &= mu\,(S/m, V/m) \\ &= m\left[u_0\mathrm{e}^{\frac{S/m-s_0}{c_v}}\left(\frac{V/m}{v_0}\right)^{-R/c_v}\right], \end{aligned}$$

therefore

$$\begin{aligned} g = \frac{\partial U}{\partial m} &= u_0\frac{\partial}{\partial m}\left[m\mathrm{e}^{\frac{S/m-s_0}{c_v}}\left(\frac{V/m}{v_0}\right)^{-R/c_v}\right] \\ &= u_0\left[\mathrm{e}^{\frac{S/m-s_0}{c_v}}\left(\frac{V/m}{v_0}\right)^{-R/c_v}\right. \\ &\quad + m\left\{\frac{-S}{c_v m^2}\mathrm{e}^{\frac{S/m-s_0}{c_v}}\left(\frac{V/m}{v_0}\right)^{-R/c_v}\right. \\ &\quad \left.\left. + \mathrm{e}^{\frac{S/m-s_0}{c_v}}\frac{V}{v_0}\left(\frac{-1}{m^2}\right)\left(\frac{-R}{c_v}\right)\left(\frac{V/m}{v_0}\right)^{-R/c_v-1}\right\}\right] \\ &= u_0\frac{1+R-S/m}{c_v}\mathrm{e}^{\frac{S/m-s_0}{c_v}}\left(\frac{V/m}{v_0}\right)^{-R/c_v}. \end{aligned}$$

```
partial class PerfectGas
  parameter SI.SpecificEntropy s_0 = 0;
  extends Partial.IdealGas(
    redeclare parameter SI.SpecificHeatCapacity c_p);
  SI.SpecificEnergy u_0 = c_v*T_0;
  SI.SpecificVolume v_0 = 1/rho_0;
equation
  u = c_v*T; // See (49).
  s = s_0 + c_v*log(u/u_0) + R*log(v/v_0); // See (51).
end PerfectGas;
```

Listing 7. Thermodynamics.Substances.Partial.PerfectGas.

```
partial class MonatomicPerfectGas
  extends Partial.PerfectGas(
    redeclare parameter SI.SpecificHeatCapacity c_p = 5*R/2);
end MonatomicPerfectGas;
```

Listing 8. Thermodynamics.Substances.Partial.MonatomicPerfectGas.

If we take into account equations (49) ($u = c_vT$, i.e., $u_0 = c_vT_0$) and (47) ($c_p - c_v = R$), then

$$\begin{aligned} g &= \frac{c_vT_0}{c_v}(c_p - S/m)\mathrm{e}^{\frac{S/m-s_0}{c_v}}\left(\frac{V/m}{v_0}\right)^{-R/c_v} \\ &= T_0(c_p - s)\mathrm{e}^{(s-s_0)/c_v}(v/v_0)^{-R/c_v} \text{ (see (55))}. \end{aligned}$$

Listing 7 shows the implementation of the class PerfectGas. Here we implement equations (49) and (51), i.e., we consider the fundamental equation in the entropy representation. Note that PerfectGas inherits equations (41) and (47) from its parent class IdealGas.

3) MONATOMIC PERFECT GASES

For *monatomic* perfect gases $c_v = \frac{3}{2}R$, $c_p = \frac{5}{2}R$, and we can write

$$s(u,v) = s_0 + \frac{3}{2}R\ln(u/u_0) + R\ln(v/v_0), \qquad (56)$$
$$u(s,v) = u_0\mathrm{e}^{\frac{2}{3}(s-s_0)/R}(v/v_0)^{-2/3}, \quad u_0 = c_vT_0, \qquad (57)$$
$$T = T_0\mathrm{e}^{\frac{2}{3}(s-s_0)/R}(v/v_0)^{-2/3}, \qquad (58)$$
$$p = p_0\mathrm{e}^{\frac{2}{3}(s-s_0)/R}(v/v_0)^{-5/3}, \qquad (59)$$
$$g = (\frac{5}{2}R - s)T_0\mathrm{e}^{\frac{2}{3}(s-s_0)}(v/v_0)^{-2/3}. \qquad (60)$$

The implementation of the class MonatomicPerfectGas is shown in Listing 8.

4) EMPIRICAL MODELS FOR NON-PERFECT SUBSTANCES

The perfect gas condition (constant c_p) may be suitable for modeling the behavior of polyatomic gases or liquids in a first approximation. However, to obtain more accurate results, we must take into account that c_p varies with temperature.

Given the difficulty of theoretically deducing this dependence from first principles, we have to employ a phenomenological approach inferred from experimental measurements. It is usual to opt for a dimensionless polynomial representation $c_p(T)/R = \sum_{i \in I \subset \mathbb{Z}} a_i T^i$ because it simplifies calculations and facilitates symbolic manipulation. The number of coefficients a_i comes from a tradeoff between accuracy and computational cost.

For example, the library of thermodynamic data used with the CEA NASA Glenn computer program contains data for over 2000 substances modeled with seven coefficients for c_p in different temperature ranges from 200 to 20000 K [30]. The expressions to calculate the specific heat capacity, specific entropy, and specific enthalpy with these coefficients are as follows:

$$\begin{aligned} \frac{c_p(T)}{R} &= a_1 T^{-2} + a_2 T^{-1} + a_3 + a_4 T \\ &\quad + a_5 T^2 + a_6 T^3 + a_7 T^4, \end{aligned} \tag{61}$$

$$\begin{aligned} \frac{s(T)}{R} &= -\frac{a_1}{2} T^{-2} - a_2 T^{-1} + a_3 \ln(T) + a_4 T \\ &\quad + \frac{a_5}{2} T^2 + \frac{a_6}{3} T^3 + \frac{a_7}{4} T^4 + b_2, \end{aligned} \tag{62}$$

$$\begin{aligned} \frac{h(T)}{RT} &= -a_1 T^{-2} + a_2 \ln(T) T^{-1} + a_3 + \frac{a_4}{2} T \\ &\quad + \frac{a_5}{3} T^2 + \frac{a_6}{4} T^3 + \frac{a_7}{5} T^4 + \frac{b_1}{T}, \end{aligned} \tag{63}$$

where b_1 and b_2 are integrations constants. Each substance can have several sets of coefficients $(a_{1\ldots7}, b_{1,2})$ grouped by temperature intervals depending on the phase (condensed or non-condensed) adopted in each interval.

To incorporate the NASA Glenn database into our Modelica implementation, we have processed the text file containing the above-mentioned coefficients [30, Appendix D] and generated the Modelica package NASAGlennCoefficients. This package contains the partial class Partial.Substance (see Listing 9) derived from class Thermodynamics.Substances.Partial.IdealGas and implements the equations (61)-(63) taking into account the temperature intervals.

The rest of the package NASAGlennCoefficients is composed of 2073 substance models. Each model is an extension of the previous class Substance.

Example 4: As an example, we show the classes modeling the thermodynamic behavior of CO_2 (Listing 10) and *air* (Listing 11). The latter is a mixture of substances, but in the NASA Glenn library, it is also modeled as an ideal gas. The class CO2 has three temperature intervals: [200…1000 K], (1000…6000 K], and (6000…20000 K].

The class Air has two temperature intervals: [200…1000 K] and (1000…6000 K].

As classes CO2 and Air extend the class Substance, they inherit equations (61)-(63), where the thermodynamic behavior of all substances modeled in the package NASAGlennCoefficients is defined. It is only necessary to initialize the parameters involved in these equations to complete the definition of the derived classes.

```
partial class Substance
  extends Thermodynamics.Substances.Partial.IdealGas;
  // phase=0 for gas, phase≠0 for condensed substances.
  parameter Integer phase;
  parameter SI.MolarMass M_mol "kg/mol";
  parameter Integer numberOfTemperatureIntervals;
  parameter SI.Temperature arrayT_min[:];
  parameter SI.Temperature arrayT_max[:];
  parameter Real a[:,7];
  parameter Real b[:,2];
  //
  Integer i;
equation
  // See (61) − (63).
  c_p/R = sum(a[i,j]*T^(j−3) for j in 1:7);
  s/R = −a[i,1]*T^(−2)/2 − a[i,2]*T^(−1) + a[i,3]*log(T) + a[i,4]*T
            + a[i,5]*T^2/2 + a[i,6]*T^3/3 + a[i,7]*T^4/4 + b[i,2];
  h/(R*T) = −a[i,1]*T^(−2)/2 + a[i,2]*log(T)/T + a[i,3] + a[i,4]*T/2
            + a[i,5]*T^2/3 + a[i,6]*T^3/4 + a[i,7]*T^4/5 + b[i,1]/T;
algorithm
  // Determination of the temperature interval.
  // We initially assume that T is in the first interval.
  i:=1;
  // Then we look to see if T is in an interval other than the first one.
  for j in 1:numberOfTemperatureIntervals loop
    if T >= arrayT_min[j] and T < arrayT_max[j] then
      i:=j; break;
    end if;
  end for;
end Substance;
```

Listing 9. NASAGlennCoefficients.Partial.Substance.

```
class CO2 "CO2"
  extends NASAGlennCoefficients.Partial.Substance(
    phase = 0 "Gas",
    M_mol = 0.0440095000,
    R = 188.9253456640,
    numberOfTemperatureIntervals = 3,
    arrayT_min = { 200.000, 1000.000, 6000.000},
    arrayT_max = { 1000.000, 6000.000, 20000.000},
    T_min = 200.000,
    T_max = 20000.000,
    a = {{ 4.943650540E+04, −6.264116010E+02, 5.301725240E+00,
           2.503813816E−03, −2.127308728E−07, −7.689988780E−10,
           2.849677801E−13},
         { 1.176962419E+05, −1.788791477E+03, 8.291523190E+00,
          −9.223156780E−05, 4.863676880E−09, −1.891053312E−12,
          6.330036590E−16},
         {−1.544423287E+09, 1.016847056E+06, −2.561405230E+02,
           3.369401080E−02, −2.181184337E−06, 6.991420840E−11,
          −8.842351500E−16}},
    b = {{−4.528198460E+04, −7.048279440E+00},
         {−3.908350590E+04, −2.652669281E+01},
         {−8.043214510E+06, 2.254177493E+03}});
end CO2;
```

Listing 10. NASAGlennCoefficients.Substances.CO2.

F. IAPWS FORMULATION 1995 FOR THE THERMODYNAMIC PROPERTIES OF ORDINARY WATER

Water is a substance widely studied by the scientific community, not only for its role in many natural processes, some of them so important as the origin of life or the evolution of Earth's climate, but also for the large number and variety of its industrial applications. For these reasons, the scientific community has endeavored to find the fundamental equation that models its thermodynamic behavior, and as a result of this effort, the *International Association for the Properties of Water and Steam* has published two empirical formulations of it, known as IAPWS-95 [22], [53] and IAPWS-IF97 [21], [52]. IAPWS-95 is the most accurate and is intended for general scientific research. IAPWS-IF97 is a simplification of the previous one and is intended for industrial applications.

```
class Air "Air"
  extends NASAGlennCoefficients.Partial.Substance(
    phase = 0 "Gas",
    M_mol = 0.0289651159,
    R = 287.0525368759,
    numberOfTemperatureIntervals = 2,
    arrayT_min = { 200.000, 1000.000},
    arrayT_max = { 1000.000, 6000.000},
    T_min = 200.000,
    T_max = 6000.000,
    a = {{ 1.009950160E+04, −1.968275610E+02, 5.009155110E+00,
           −5.761013730E−03, 1.066859930E−05, −7.940297970E−09,
           2.185231910E−12},
         { 2.415214430E+05, −1.257874600E+03, 5.144558670E+00,
           −2.138541790E−04, 7.065227840E−08, −1.071483490E−11,
           6.577800150E−16}},
    b = {{−1.767967310E+02, −3.921504225E+00},
         { 6.462263190E+03, −8.147411905E+00}});
end Air;
```

Listing 11. NASAGlennCoefficients.Substances.Air.

Although it is less accurate, it has the advantage of consuming less computational resources when performing simulations.

The Modelica Standard Library implements the IAPWS-IF97 model in the package Modelica.Media.Water.WaterIF97. We have implemented the IAPWS-95 model in the class Thermodynamics.Substances.WaterIAPWS95. This model formulates the fundamental equation of water in terms of the specific Helmholtz free energy $f(T, v) = f(T, 1/\varrho)$. The model has two dimensionless terms: $\phi^{\mathrm{o}}(\delta, \tau)$ for the ideal-gas part and $\phi^{\mathrm{r}}(\delta, \tau)$ for the residual part:

$$
\begin{aligned}
\frac{f(\varrho, T)}{RT} &= \phi^{\mathrm{o}}(\delta, \tau) + \phi^{\mathrm{r}}(\delta, \tau),\\
\delta &= \varrho/\varrho_c, \quad \tau = T_c/T,\\
\varrho_c &= 322\ \mathrm{kg/m^3}, \quad T_c = 647.096\ \mathrm{K},\\
R &= 0.461\,518\,05\ \mathrm{kJ/(kg{\cdot}K)}. \qquad (64)
\end{aligned}
$$

The ideal-gas part of (64) is

$$
\phi^{\mathrm{o}} = \ln\delta + n_1^{\mathrm{o}} + n_2^{\mathrm{o}}\tau + n_3^{\mathrm{o}}\ln\tau + \sum_{i\in 4\ldots 8} n_i^{\mathrm{o}} \ln\left[\mathrm{e}^{-\gamma_i^{\mathrm{o}}\tau}\right], \quad (65)
$$

and we can see that it fulfills the condition of being an separate variables equation (see Section IV-E). The coefficients $n^{\mathrm{o}}_{1\ldots 8}$ and $\gamma^{\mathrm{o}}_{4\ldots 8}$ can be consulted in [22, Table 1].

The residual part of (64) is

$$
\begin{aligned}
\phi^{\mathrm{r}} &= \sum_{i\in 1\ldots 7} n_i\delta^{d_i}\tau^{t_i} + \sum_{i\in 8\ldots 51} n_i\delta^{d_i}\tau^{t_i}\mathrm{e}^{-\delta^{c_i}}\\
&\quad + \sum_{i\in 52\ldots 54} n_i\delta^{d_i}\tau^{t_i}\mathrm{e}^{-\alpha_i(\delta-\varepsilon_i)^2-\beta_i(\tau-\gamma_i)^2},\\
&\quad + \sum_{i\in 55\ldots 56} n_i\Delta^{b_i}\delta\Psi,\\
\Delta &= \Theta^2 + B_i\left[(\delta-1)^2\right]^{a_i},\\
\Theta &= (1-\tau) + A_i\left[(\delta-1)^2\right]^{\frac{1}{2\beta_i}},\\
\Psi &= \mathrm{e}^{-C_i(\delta-1)^2-D_i(\tau-1)^2}. \qquad (66)
\end{aligned}
$$

The coefficients $c_{8\ldots 51}$, $d_{1\ldots 54}$, $t_{1\ldots 54}$, $n_{1\ldots 56}$, $\alpha_{52\ldots 54}$, $\beta_{52\ldots 56}$, $\gamma_{52\ldots 54}$, $\varepsilon_{52\ldots 54}$, $a_{55\ldots 56}$, $b_{55\ldots 56}$, $B_{55\ldots 56}$, $C_{55\ldots 56}$, $D_{55\ldots 56}$ and $A_{55\ldots 56}$ are listed in [22, Table 2].

Since (64) is a fundamental equation, it contains all the information about the thermodynamic behavior of water. Therefore, all other thermodynamic properties and quantities can be reduced from f. In our implementation of the IAPWS-95 model, we are also interested in calculating the specific entropy s, pressure p, the isobaric heat capacity c_p, and the isochoric heat capacity c_v:

$$
\frac{s(\delta,\tau)}{RT} = \tau(\phi^{\mathrm{o}}_\tau + \phi^{\mathrm{r}}_\tau) - \phi^{\mathrm{o}} - \phi^{\mathrm{r}}, \qquad (67)
$$

$$
\frac{p(\delta,\tau)}{\varrho RT} = 1 + \delta\phi^{\mathrm{r}}_\delta, \qquad (68)
$$

$$
\frac{c_p}{R} = -\tau^2(\phi^{\mathrm{o}}_{\tau\tau} + \phi^{\mathrm{r}}_{\tau\tau}) + \frac{(1+\delta\phi^{\mathrm{r}}_\delta - \delta\tau\phi^{\mathrm{r}}_{\delta\tau})^2}{1+2\delta\phi^{\mathrm{r}}_\delta + \delta^2\phi^{\mathrm{r}}_{\delta\delta}}, \qquad (69)
$$

$$
\frac{c_v}{R} = -\tau^2(\phi^{\mathrm{o}}_{\tau\tau} + \phi^{\mathrm{r}}_{\tau\tau}). \qquad (70)
$$

TABLE 1. Partial derivatives ϕ^{o}_τ, ϕ^{r}_τ, and ϕ^{r}_δ (adapted from [22, Table 5]).

$$
\phi^{\mathrm{o}}_\tau = \frac{\partial\phi^{\mathrm{o}}}{\partial\tau} = n_2^{\mathrm{o}} + \frac{n_3^{\mathrm{o}}}{\tau} + \sum_{i=4}^{8} n_i^{\mathrm{o}}\gamma_i^{\mathrm{o}}\left[\left(1-\mathrm{e}^{-\gamma_i^{\mathrm{o}}\tau}\right)^{-1} - 1\right], \qquad (71)
$$

$$
\begin{aligned}
\phi^{\mathrm{r}}_\tau = \frac{\partial\phi^{\mathrm{r}}}{\partial\tau} &= \sum_{i=1}^{7} n_i t_i\delta^{d_i}\tau^{t_i-1} + \sum_{i=8}^{51} n_i t_i\delta^{d_i}\tau^{t_i-1}\mathrm{e}^{-\delta^{c_i}} + \sum_{i=52}^{54} n_i\delta^{d_i}\tau^{t_i}\mathrm{e}^{-\alpha_i(\delta-\varepsilon_i)^2-\beta_i(\tau-\gamma_i)^2}\left[\frac{t_i}{\tau} - 2\beta_i(\tau-\gamma_i)\right]\\
&\quad + \sum_{i=55}^{56} n_i\delta\left[\frac{\partial\Delta^{b_i}}{\partial\tau}\Psi + \Delta^{b_i}\frac{\partial\Psi}{\partial\tau}\right], \qquad (72)
\end{aligned}
$$

$$
\begin{aligned}
\phi^{\mathrm{r}}_\delta = \frac{\partial\phi^{\mathrm{r}}}{\partial\delta} &= \sum_{i=1}^{7} n_i d_i\delta^{d_i-1}\tau^{t_i} + \sum_{i=8}^{51} n_i\mathrm{e}^{-\delta^{c_i}}\left[\delta^{d_i-1}\tau^{t_i}\left(d_i - c_i\delta^{c_i}\right)\right] + \sum_{i=52}^{54} n_i\delta^{d_i}\tau^{t_i}\mathrm{e}^{-\alpha_i(\delta-\varepsilon_i)^2-\beta_i(\tau-\gamma_i)^2}\left[\frac{d_i}{\delta} - 2\alpha_i(\delta-\varepsilon_i)\right]\\
&\quad + \sum_{i=55}^{56} n_i\left[\Delta^{b_i}\left(\Psi + \delta\frac{\partial\Psi}{\partial\delta} + \frac{\partial\Delta^{b_i}}{\partial\delta}\delta\Psi\right)\right], \qquad (73)
\end{aligned}
$$

$$
\frac{\partial\Delta^{b_i}}{\partial\tau} = -2\Theta b_i\Delta^{b_i-1}, \qquad (74)
$$

$$
\frac{\partial\Delta^{b_i}}{\partial\delta} = b_i\Delta^{b_i-1}\frac{\partial\Delta}{\partial\delta}, \qquad (75)
$$

$$
\frac{\partial\Delta}{\partial\delta} = (\delta-1)\left\{A_i\Theta\frac{2}{\beta_i}\left[(\delta-1)^2\right]^{\frac{1}{2\beta_1}-1} + 2B_i a_i\left[(\delta-1)^2\right]^{a_i-1}\right\}, \qquad (76)
$$

$$
\frac{\partial\Psi}{\partial\tau} = -2D_i(\tau-1)\Psi, \qquad \frac{\partial\Psi}{\partial\delta} = -2C_i(\delta-1)\Psi. \qquad (77)
$$

TABLE 2. Second order partial derivatives $\phi^{\mathrm{o}}_{\tau\tau}$, $\phi^{\mathrm{r}}_{\tau\tau}$, $\phi^{\mathrm{r}}_{\delta\tau}$, and $\phi^{\mathrm{r}}_{\delta\delta}$ (adapted from [22, Table 5]).

$$\phi^{\mathrm{o}}_{\tau\tau} = \frac{\partial^2 \phi^{\mathrm{o}}}{\partial \tau^2} = -\frac{n^{\mathrm{o}}_3}{\tau^2} - \sum_{i=4}^{8} n^{\mathrm{o}}_i (\gamma^{\mathrm{o}}_i)^2 \mathrm{e}^{-\gamma^{\mathrm{o}}_i \tau} \left(1 - \mathrm{e}^{-\gamma^{\mathrm{o}}_i \tau}\right)^{-2}, \tag{78}$$

$$\begin{aligned}\phi^{\mathrm{r}}_{\tau\tau} = \frac{\partial^2 \phi^{\mathrm{r}}}{\partial \tau^2} &= \sum_{i=1}^{7} n_i t_i (t_i - 1)\delta^{d_i}\tau^{t_i-2} + \sum_{i=8}^{51} n_i t_i (t_i - 1)\delta^{d_i}\tau^{t_i-2}\mathrm{e}^{-\delta^{c_i}} \\ &+ \sum_{i=52}^{54} n_i \delta^{d_i}\tau^{t_i}\mathrm{e}^{-\alpha_i(\delta-\varepsilon_i)^2-\beta_i(\tau-\gamma_i)^2}\left[\left(\frac{t_i}{\tau} - 2\beta_i(\tau-\gamma_i)\right)^2 - \frac{t_i}{\tau^2} - 2\beta_i\right] + \sum_{i=55}^{56} n_i \delta \left[\frac{\partial^2 \Delta^{b_i}}{\partial \tau^2}\Psi + 2\frac{\partial \Delta^{b_i}}{\partial \tau}\frac{\partial \Psi}{\partial \tau} + \Delta^{b_i}\frac{\partial^2 \Psi}{\partial \tau^2}\right],\end{aligned} \tag{79}$$

$$\begin{aligned}\phi^{\mathrm{r}}_{\delta\tau} = \frac{\partial^2 \phi^{\mathrm{r}}}{\partial \delta \partial \tau} &= \sum_{i=1}^{7} n_i d_i t_i \delta^{d_i-1}\tau^{t_i-1} + \sum_{i=8}^{51} n_i t_i \delta^{d_i-1}\tau^{t_i-1}\left(d_i - c_i\delta^{c_i}\right)\mathrm{e}^{-\delta^{c_i}} \\ &+ \sum_{i=52}^{54} n_i \delta^{d_i}\tau^{t_i}\mathrm{e}^{-\alpha_i(\delta-\varepsilon_i)^2-\beta_i(\tau-\gamma_i)^2}\left[\frac{d_i}{\delta} - 2\alpha_i(\delta-\varepsilon_i)\right]\left[\frac{t_i}{\tau} - 2\beta_i(\tau-\gamma_i)\right] \\ &+ \sum_{i=55}^{56} n_i \left[\Delta^{b_i}\left(\frac{\partial \Psi}{\partial \tau} + \delta\frac{\partial^2 \Psi}{\partial \delta \partial \tau}\right) + \delta\frac{\partial \Delta^{b_i}}{\partial \delta}\frac{\partial \Psi}{\partial \tau} + \frac{\partial \Delta^{b_i}}{\partial \tau}\left(\Psi + \delta\frac{\partial \Psi}{\partial \delta}\right) + \frac{\partial^2 \Delta^{b_i}}{\partial \delta \partial \tau}\delta\Psi\right],\end{aligned} \tag{80}$$

$$\begin{aligned}\phi^{\mathrm{r}}_{\delta\delta} = \frac{\partial^2 \phi^{\mathrm{r}}}{\partial \delta^2} &= \sum_{i=1}^{7} n_i d_i (d_i - 1)\delta^{d_i-2}\tau^{t_i} + \sum_{i=1}^{51} n_i \mathrm{e}^{-\delta^{c_i}}\left[\delta^{d_i-2}\tau^{t_i}\left(\left(d_i - c_i\delta^{c_i}\right)\left(d_i - 1 - c_i\delta^{c_i}\right) - c_i^2\delta^{\breve{c}i}\right)\right] \\ &+ \sum_{i=52}^{54} n_i \tau^{t_i}\mathrm{e}^{-\alpha_i(\delta-\varepsilon_i)^2-\beta_i(\tau-\gamma_i)^2}\left[-2\alpha_i\delta^{\breve{d}i} + 4\alpha_i^2\delta^{d_i}(\delta-\varepsilon_i)^2 - 4d_i\alpha_i\delta^{d_i-1}(\delta-\varepsilon_i) + d_i(d_i-1)\delta^{d_i-2}\right] \\ &+ \sum_{i=55}^{56} n_i \left[\Delta^{b_i}\left(2\frac{\partial \Psi}{\partial \delta} + \delta\frac{\partial^2 \Psi}{\partial \delta^2}\right) + 2\frac{\partial \Delta^{b_i}}{\partial \delta}\left(\Psi + \delta\frac{\partial \Psi}{\partial \delta}\right) + \frac{\partial^2 \Delta^{b_i}}{\partial \delta^2}\delta\Psi\right],\end{aligned} \tag{81}$$

$$\frac{\partial^2 \Delta^{b_i}}{\partial \tau^2} = 2b_i\Delta^{b_i-1} + 4\Theta^2 b_i(b_i-1)\Delta^{b_i-2}, \tag{82}$$

$$\frac{\partial^2 \Delta^{b_i}}{\partial \delta^2} = b_i\left\{\Delta^{b_i-1}\frac{\partial^2 \Delta}{\partial \delta^2} + (b_i-1)\Delta^{b_i-2}\left(\frac{\partial \Delta}{\partial \delta}\right)^2\right\}, \tag{83}$$

$$\frac{\partial^2 \Delta^{b_i}}{\partial \delta \partial \tau} = -A_i b_i \frac{2}{\beta_i}\Delta^{b_i-1}(\delta-1)\left[(\delta-1)^2\right]^{\frac{1}{2\beta_i}-1} - 2\Theta_i b_i(b_i-1)\Delta^{b_i-2}\frac{\partial \Delta}{\partial \delta}, \tag{84}$$

$$\begin{aligned}\frac{\partial^2 \Delta}{\partial \delta^2} &= \frac{1}{\delta-1}\frac{\partial \Delta}{\partial \delta} + (\delta-1)^2\left\{4B_i a_i(a_i-1)\left[(\delta-1)^2\right]^{a_i-2} + 2A_i^2\left(\frac{1}{\beta_i}\right)^2\left\{\left[(\delta-1)^2\right]^{\frac{1}{2\beta_i}-1}\right\}^2\right. \\ &\left. + A_i\Theta\frac{4}{\beta_i}\left(\frac{1}{2\beta_i} - 1\right)\left[(\delta-1)^2\right]^{\frac{1}{2\beta_i}-2}\right\},\end{aligned} \tag{85}$$

$$\frac{\partial^2 \Psi}{\partial \tau^2} = \left\{2D_i(\tau-1)^2 - 1\right\}2D_i\Psi, \qquad \frac{\partial^2 \Psi}{\partial \delta^2} = \left\{(2C_i(\delta-1)^2 - 1\right\}2C_i\Psi, \qquad \frac{\partial^2 \Psi}{\partial \delta \partial \tau} = 4C_i D_i(\delta-1)(\tau-1)\Psi. \tag{86}$$

Table 1 shows the partial derivatives ϕ^{o}_{τ}, ϕ^{r}_{τ}, and $\phi^{\mathrm{r}}_{\delta}$ used in (67)-(68). The second partial derivatives $\phi^{\mathrm{o}}_{\tau\tau}$, $\phi^{\mathrm{r}}_{\tau\tau}$, $\phi^{\mathrm{r}}_{\delta\tau}$, and $\phi^{\mathrm{r}}_{\delta\delta}$ used in (69)-(70) are shown in Table 2.

Listing 12 is the implementation of the Modelica class WaterIAPWS95. This implementation has been validated with the computer-program verification test values given in [22, Tables 6 and 7].

G. EMPIRICAL MODELS FOR GENERAL SUBSTANCES

We have already seen (Figure 1) that the port-Hamiltonian model of a simple thermodynamic system requires to know the covector field $\mathrm{d}U$ (see (4)). Although U and $\mathrm{d}U$ are not directly within our reach, we can approximate them numerically from the primary set of response functions (Section II-B2). As these properties are measured and tabulated as a function of temperature and pressure, it is more convenient to work with Gibbs potential $G(T, p, m)$ or specific Gibbs potential $G/m = g(T, p)$ rather than internal energy $U(S, V, m)$. Thus, we can write the following equations:

$$\mathrm{d}s = \left.\frac{\partial s}{\partial T}\right|_p \mathrm{d}T + \left.\frac{\partial s}{\partial p}\right|_T \mathrm{d}p = \frac{c_p}{T}dT - v\alpha_p dp,^{3}$$

$$\mathrm{d}v = \left.\frac{\partial v}{\partial T}\right|_p \mathrm{d}T + \left.\frac{\partial v}{\partial p}\right|_T \mathrm{d}p = v\alpha_p\mathrm{d}T - v\kappa_T\mathrm{d}p,$$

$$\mathrm{d}g = \left.\frac{\partial g}{\partial T}\right|_p \mathrm{d}T + \left.\frac{\partial g}{\partial p}\right|_T \mathrm{d}p = -s\mathrm{d}T + v\mathrm{d}p,$$

and in the CIT approach

$$\dot{s} = \frac{c_p}{T}\dot{T} - v\alpha_p\dot{p}, \tag{87}$$

$$\dot{v} = v\alpha_p\dot{T} - v\kappa_T\dot{p}, \tag{88}$$

$$\dot{g} = -s\dot{T} + v\dot{p}. \tag{89}$$

For practical considerations, it is useful to relate α_p and κ_T to the density $\varrho = m/V = 1/v$ because it is easier to tabulate $\varrho(T, p)$ than α_p or κ_T:

$$\alpha_p = \frac{1}{v}\left.\frac{\partial v}{\partial T}\right|_p = \varrho\left.\frac{\partial(1/\varrho)}{\partial T}\right|_p = -\frac{1}{\varrho}\left.\frac{\partial \varrho}{\partial T}\right|_p, \tag{90}$$

$$\kappa_T = -\frac{1}{v}\left.\frac{\partial v}{\partial p}\right|_T = -\varrho\left.\frac{\partial(1/\varrho)}{\partial p}\right|_T = \frac{1}{\varrho}\left.\frac{\partial \varrho}{\partial p}\right|_T. \tag{91}$$

[3]If the potentials satisfy the Maxwell relations, then $\partial^2 U/\partial S\partial V = \partial^2 U/\partial V\partial S$ and we can write $\partial T/\partial V = -\partial p/\partial S$, i.e., $\partial V/\partial T = -\partial S/\partial p$.

```
class WaterIAPWS95 "IAPWS R6-95(2018)"
  extends Partial.SingleSubstance(
    T_min = 235,
    T_max = 1273,
    p_min = 1E-3*MPa,
    p_max = 1E3*MPa);
  constant SI.Temperature T_c = 647.096; // Critical temperature $T_c$.
  constant SI.Density rho_c = 322; // Critical density $\varrho_c$.
  // Specific gas constant $R$ [kJ/(kg·K)].
  constant SpecificHeatCapacity R = 0.46151805;
  // Fundamental set of response functions.
  SI.SpecificHeatCapacity c_v; // Isochoric heat capacity [J/(kg·K)].
  //
  // Independent variables [dimensionless].
  Real delta(start=rho_0/rho_c) = rho/rho_c; // $\delta = \varrho/\varrho_c$.
  Real tau(start=T_c/T_0) = T_c/T; // $\tau = T_c/T$.
  // Ideal-gas part $\phi^o$, and residual $\phi^r$ part [dimensionless].
  Real phi_o, phi_r;
  //
  // Coefficients for $\phi^o$, see  [22, Table 1, p.9].
  constant Real n_o[8] = {-8.3204464837497,..., 0.24873}; // $n^o_{1...8}$
  constant Real gamma_o[5] = {1.28728967,..., 27.5075105}; // $\gamma^o_{4...8}$
  //
  // Coefficients for $\phi^r$, see  [22, Table 2, p.10].
  constant Integer c[44] = {1, 1,..., 6}; // $c_{8...51}$
  constant Integer d[54] = {1, 1,..., 3}; // $d_{1...54}$
  constant Real t[54] = {-0.5, 0.875,..., 4}; // $t_{1...54}$
  constant Real n[56] = {0.12533547935523E-1,...}; // $n_{1...,56}$
  constant Real alpha[3] = {20, 20, 20}; // $\alpha_{52...54}$
  constant Real beta[5] = {150, 150, 250, 0.3, 0.3}; // $\beta_{52...56}$
  constant Real gamma[3] = {1.21, 1.21, 1.25}; // $\gamma_{52...54}$
  constant Real epsilon[3] = {1, 1, 1}; // $\varepsilon_{52...54}$
  constant Real a[2] = {3.5, 3.5}; // $a_{55...56}$
  constant Real b[2] = {0.85, 0.95}; // $b_{55...56}$
  constant Real B[2] = {0.2, 0.2}; // $B_{55...56}$
  constant Real C[2] = {28, 32}; // $C_{55...56}$
  constant Real D[2] = {700, 800}; // $D_{55...56}$
  constant Real A[2] = {0.32, 0.32}; // $A_{55...56}$
  Real Delta[2], Theta[2], Psi[2]; // $\Delta_{55...56}$, $\Theta_{55...56}$, $\Psi_{55...56}$
  //
  // Partial derivatives
  Real phi_o_tau; // $\partial\phi^o/\partial\tau$
  Real phi_o_tau_tau; // $\partial^2\phi^o/\partial\tau^2$
  Real phi_r_tau; // $\partial\phi^r/\partial\tau$
  Real phi_r_tau_tau; // $\partial^2\phi^r/\partial\tau^2$
  Real phi_r_delta_tau; // $\partial^2\phi^r/\partial\delta\partial\tau$
  Real phi_r_delta_delta; // $\partial^2\phi^r/\partial\delta^2$
  Real Delta_bi_tau[2]; // $[\partial\Delta^{b_i}/\partial\tau]_{55...56}$
  Real Psi_tau[2]; // $[\partial\Psi/\partial\tau]_{55...56}$
  //
  Real phi_r_delta; // $\partial\phi^r/\partial\delta$
  Real Delta_delta[2]; // $[\partial^2\Delta/\partial\delta^2]_{55...56}$
  Real Delta_delta_delta[2]; // $[\partial\Delta/\partial\delta]_{55...56}$
  Real Delta_bi_delta[2]; // $[\partial\Delta^{b_i}/\partial\delta]_{55...56}$
  Real Delta_bi_tau_tau[2]; // $[\partial^2\Delta^{b_i}/\partial\tau^2]_{55...56}$
  Real Delta_bi_delta_tau[2]; // $[\partial^2\Delta^{b_i}/\partial\delta\partial\tau]_{55...56}$
  Real Delta_bi_delta_delta[2]; // $[\partial^2\Delta^{b_i}/\partial\delta^2]_{55...56}$
  Real Psi_delta[2]; // $[\partial\Psi/\partial\delta]_{55...56}$
  Real Psi_tau_tau[2]; // $[\partial^2\Psi/\partial\tau^2]_{55...56}$
  Real Psi_delta_delta[2]; // $[\partial^2\Psi/\partial\delta\partial\delta]_{55...56}$
  Real Psi_delta_tau[2]; // $[\partial^2\Psi/\partial\delta\partial\tau]_{55...56}$
equation
  // Specific Helmholtz free energy $f$. See (64).
  f/(R*kJ_kgK*T) = (phi_o + phi_r);
  // Ideal-gas part $\phi^o$ of the dimensionless Helmholtz free energy.
  // See (65).
  phi_o = ln(delta) + n_o[1] + n_o[2]*tau + n_o[3]*ln(tau)
        + sum(n_o[i]*ln(1-exp(-gamma_o[i-3]*tau)) for i in 4:8);
  //
  // Residual part $\phi^r$ of the dimensionless Helmholtz free energy.
  // See (66).
  Delta = {Theta[i]^2+B[i]*((delta-1)^2)^a[i] for i in 1:2};
  Theta = {(1-tau)+A[i]*((delta-1)^2)^(1/(2*beta[i+3])) for i in 1:2};
  Psi = {exp(-C[i]*(delta-1)^2-D[i]*(tau-1)^2) for i in 1:2};
  phi_r = sum(n[i]*delta^d[i]*tau^t[i] for i in 1:7)
        + sum(n[i]*delta^d[i]*tau^t[i]*exp(-delta^c[i-7]) for i in 8:51)
```

Listing 12. **Thermodynamics.Substances.WaterIAPWS95.**

```
        + sum(n[i]*delta^d[i]*tau^t[i]*
              exp(-alpha[i-51]*(delta-epsilon[i-51])^2
                  -beta[i-51]*(tau-gamma[i-51])^2) for i in 52:54)
        + sum(n[i]*Delta[i-54]^b[i-54]*delta*Psi[i-54] for i in 55:56);
  //
  // Equations to calculate specific entropy $s$. See (67).
  s/(R*kJ_kgK) = tau*(phi_o_tau+phi_r_tau)-phi_o-phi_r;
  //
  // $\phi^o_\tau = \partial\phi^o/\partial\tau$, (71).
  phi_o_tau = n_o[2] + n_o[3]/tau
    +sum(n_o[i]*gamma_o[i-4+1]
              *((1-exp(-gamma_o[i-3]*tau))^(-1)-1) for i in 4:8);
  //
  // $\partial\Delta^{b_i}/\partial\tau$, (74).
  Delta_bi_tau = {-2*Theta[i]*b[i]*Delta[i]^(b[i]-1) for i in 1:2};
  // $\partial\Psi/\partial\tau$, (77).
  Psi_tau = {-2*D[i]*(tau-1)*Psi[i] for i in 1:2};
  //
  // $\phi^r_\tau = \partial\phi^r/\partial\tau$, (72).
  phi_r_tau = sum(n[i]*t[i]*delta^d[i]*tau^(t[i]-1) for i in 1:7)
    +sum(n[i]*t[i]*delta^d[i]*tau^(t[i]-1)*exp(-delta^c[i-7]) for i in 8:51)
    +sum(n[i]*delta^d[i]*tau^t[i]
             *exp(-alpha[i-51]*(delta-epsilon[i-51])^2
                 -beta[i-51]*(tau-gamma[i-51])^2)
             *(t[i]/tau-2*beta[i-51]*(tau-gamma[i-51])) for i in 52:54)
    +sum(n[i]*delta*(Delta_bi_tau[i-54]*Psi[i-54]+
                    Delta[i-54]^b[i-54]*Psi_tau[i-54]) for i in 55:56);
  //
  // Equations to calculate pressure $p$. See (68).
  p/(rho*R*kJ_kgK*T) = 1+delta*phi_r_delta;
  //
  // $\partial\Delta/\partial\delta$, (76).
  Delta_delta = {
    (delta-1)*(A[i]*Theta[i]*2/beta[i+3]*((delta-1)^2)^(1/(2*beta[i+3])-1)+
              2*B[i]*a[i]*((delta-1)^2)^(a[i]-1)) for i in 1:2};
  // $\partial\Delta^{b_i}/\partial\delta$, (75).
  Delta_bi_delta = {b[i]*Delta[i]^(b[i]-1)*Delta_delta[i] for i in 1:2};
  // $\partial\Psi/\partial\delta$, (77).
  Psi_delta = {-2*C[i]*(delta-1)*Psi[i] for i in 1:2};
  //
  // $\phi^r_\delta = \partial\phi^r_\delta/\partial\delta$, (73).
  phi_r_delta = sum(n[i]*d[i]*delta^(d[i]-1)*tau^t[i] for i in 1:7)
    +sum(n[i]*
         exp(-delta^c[i-7])*
         (delta^(d[i]-1)*tau^t[i]*(d[i]-c[i-7]*delta^c[i-7])) for i in 8:51)
    +sum(n[i]*delta^d[i]*tau^t[i]*
         exp(-alpha[i-51]*(delta-epsilon[i-51])^2
             -beta[i-51]*(tau-gamma[i-51])^2)*
         (d[i]/delta-2*alpha[i-51]*(delta-epsilon[i-51])) for i in 52:54)
    +sum(n[i]*(Delta[i-54]^b[i-54]*(Psi[i-54]+delta*Psi_delta[i-54])+
               Delta_bi_delta[i-54]*delta*Psi[i-54]) for i in 55:56);
  //
  // Equations to calculate the isobaric heat capacity $c_p$. See (69).
  c_p/R = -tau^2*(phi_o_tau_tau+phi_r_tau_tau)
          +(1+delta*phi_r_delta-delta*tau*phi_r_delta_tau)^2
          /(1+2*delta*phi_r_delta+delta^2*phi_r_delta_delta);
  //
  // $\phi^o_{\tau\tau} = \partial^2\phi^o/\partial\tau^2$, (78).
  phi_o_tau_tau = -n_o[3]/tau^2
      -sum(n_o[i]*gamma_o[i-3]^2*exp(-gamma_o[i-3]*tau)*
          (1-exp(-gamma_o[i-3]*tau))^(-2) for i in 4:8);
  //
  // $\phi^r_{\tau\tau} = \partial^2\phi^r/\partial\tau^2$, (79).
  phi_r_tau_tau =
     sum(n[i]*t[i]*(t[i]-1)*delta^d[i]*tau^(t[i]-2) for i in 1:7)
    +sum(n[i]*t[i]*(t[i]-1)*delta^d[i]*tau^(t[i]-2)*exp(-delta^c[i-7])
       for i in 8:51)
    +sum(n[i]*delta^d[i]*tau^t[i]
         *exp(-alpha[i-51]*(delta-epsilon[i-51])^2
             -beta[i-51]*(tau-gamma[i-51])^2)
         *((t[i]/tau-2*beta[i-51]*(tau-gamma[i-51]))^2
           -t[i]/tau^2-2*beta[i-51]) for i in 52:54)
    +sum(n[i]*delta*(Delta_bi_tau_tau[i-54]*Psi[i-54]
         +2*Delta_bi_tau[i-54]*Psi_tau[i-54]
         +Delta[i-54]^b[i-54]*Psi_tau_tau[i-54]) for i in 55:56);
  //
  // $\partial^2\Delta^{b_i}/\partial\tau^2$, (82).
```

Listing 12. ***(Continued.)*** **Thermodynamics.Substances.WaterIAPWS95.**

```
Delta_bi_tau_tau = {2*b[i]*Delta[i]^(b[i]−1)
                  +4*Theta[i]^2*b[i]*(b[i]−1)*Delta[i]^(b[i]−2) for i in 1:2};
// $\partial^2\Psi/\partial\tau^2$, (86).
Psi_tau_tau = {(2*D[i]*(tau−1)^2−1)*2*D[i]*Psi[i] for i in 1:2};
//
// $\phi^r_{\delta\tau} = \partial^2\phi^r/\partial\delta\partial\tau$, (80).
phi_r_delta_tau =
   sum(n[i]*d[i]*t[i]*delta^(d[i]−1)*tau^(t[i]−1) for i in 1:7)
  +sum(n[i]*t[i]*delta^(d[i]−1)*tau^(t[i]−1)*(d[i]−c[i−7]*delta^c[i−7])
       *exp(−delta^c[i−7]) for i in 8:51)
  +sum(n[i]*delta^d[i]*tau^t[i]
       *exp(−alpha[i−51]*(delta−epsilon[i−51])^2
           −beta[i−51]*(tau−gamma[i−51])^2)
       *(d[i]/delta−2*alpha[i−51]*(delta−epsilon[i−51]))
       *(t[i]/tau−2*beta[i−51]*(tau−gamma[i−51])) for i in 52:54)
  +sum(n[i]*(
      Delta[i−54]^b[i−54]*(Psi_tau[i−54]+delta*Psi_delta_tau[i−54])
    +delta*Delta_bi_delta[i−54]*Psi_tau[i−54]
    +Delta_bi_tau[i−54]*(Psi[i−54]+delta*Psi_delta[i−54])
    +Delta_bi_delta_tau[i−54]*delta*Psi[i−54]) for i in 55:56);
//
// $\partial^2\Delta^{b_i}/\partial\delta\partial\tau$, (84).
Delta_bi_delta_tau = {−A[i]*b[i]*2/beta[i+3]*Delta[i]^(b[i]−1)
   *(delta−1)*((delta−1)^2)^(1/(2*beta[i+3])−1)
  −2*Theta[i]*b[i]*(b[i]−1)*Delta[i]^(b[i]−2)*Delta_delta[i] for i in 1:2};
// $\partial^2\Psi/\partial\delta\partial\tau$, (86).
Psi_delta_tau = {4*C[i]*D[i]*(delta−1)*(tau−1)*Psi[i] for i in 1:2};
//
// $\phi^r_{\delta\delta} = \partial^2\phi^r/\partial\delta^2$, (81).
phi_r_delta_delta =
   sum(n[i]*d[i]*(d[i]−1)*delta^(d[i]−2)*tau^t[i] for i in 1:7)
  +sum(n[i]*exp(−delta^c[i−7])
    *(delta^(d[i]−2)*tau^t[i]
      *((d[i]−c[i−7]*delta^c[i−7])*(d[i]−1−c[i−7]*delta^c[i−7])
        −c[i−7]^2*delta^c[i−7])) for i in 8:51)
  +sum(n[i]*tau^t[i]
    *exp(−alpha[i−51]*(delta−epsilon[i−51])^2
         −beta[i−51]*(tau−gamma[i−51])^2)
    *(−2*alpha[i−51]*delta^d[i]
      +4*alpha[i−51]^2*delta^d[i]*(delta−epsilon[i−51])^2
      −4*d[i]*alpha[i−51]*delta^(d[i]−1)*(delta−epsilon[i−51])
      +d[i]*(d[i]−1)*delta^(d[i]−2)) for i in 52:54)
  +sum(n[i]*(Delta[i−54]^b[i−54]
    *(2*Psi_delta[i−54]+delta*Psi_delta_delta[i−54])
    +2*Delta_bi_delta[i−54]*(Psi[i−54]+delta*Psi_delta[i−54])
    +Delta_bi_delta_delta[i−54]*delta*Psi[i−54]) for i in 55:56);
//
// $\partial^2\Delta^{b_i}/\partial\delta^2$, (83).
Delta_bi_delta_delta = {b[i]*(Delta[i]^(b[i]−1)*Delta_delta_delta[i]
    +(b[i]−1)*Delta[i]^(b[i]−2)*Delta_delta[i]^2) for i in 1:2};
// $\partial^2\Delta/\partial\delta^2$, (85).
Delta_delta_delta = {1/(delta−1)*Delta_delta[i]+(delta−1)^2
    *(4*B[i]*a[i]*(a[i]−1)*((delta−1)^2)^(a[i]−2)
      +2*A[i]^2*(1/beta[i+3])^2*(((delta−1)^2)^(1/(2*beta[i+3])−1))^2
      +A[i]*Theta[i]*4/beta[i+3]*(1/(2*beta[i+3])−1)
         *((delta−1)^2)^(1/(2*beta[i+3])−2)) for i in 1:2};
// $\partial^2\Psi/\partial\delta^2$, (86).
Psi_delta_delta = {(2*C[i]*(delta−1)^2−1)*2*C[i]*Psi[i] for i in 1:2};
//
// Equation to calculate the isochoric heat capacity $c_v$. See (70).
c_v/R = −tau^2*(phi_o_tau_tau+phi_r_tau_tau);
end WaterIAPWS95;
```

Listing 12. ***(Continued.)*** **Thermodynamics.Substances.WaterIAPWS95.**

Therefore, if we can measure c_p and ϱ as a function of temperature and pressure, we only need equations (87) and (89) to model the thermodynamic behavior of a single-substance. Note that (88) is redundant because $v = 1/\varrho$.

Example 5: Syltherm 800 is a silicone fluid manufactured by Dow Corning Corporation used mainly as HTF in facilities like thermal solar power plants. The manufacturer provides tables with the thermodynamic properties of the fluid in the temperature range 233.15 K $\leq T \leq$ 673.15 K [15], where the following polynomial approximations can be obtained for c_p, ϱ, and $\partial\varrho/\partial T$ [34]:

$$c_p = 1.01787 + 1.70736 \times 10^{-3} T \text{ kJ/(kg·K)}, \quad (92)$$

$$\varrho = 1.2691 \times 10^3 - 1.52115T + 1.79133 \times 10^{-3} T^2 - 1.67145 \times 10^{-6} T^3 \text{kg/m}^3, \quad (93)$$

```
class Syltherm800 "SYLTHERM 800 Heat Transfer Fluid"
 extends Partial.SingleSubstance(
   T_min = −40+ZeroCelsius,
   T_max = 400+ZeroCelsius);
 SI.DerDensityByTemperature rho_T; // $\varrho_T = \partial\varrho/\partial T$
 Real alpha_p; // Isobaric expansibility $\alpha_p$.
equation
 // Specific isobaric heat capacity $c_p$ and density $\varrho$
 // for $233.15\,\mathrm{K} \leq T \leq 673.15\,\mathrm{K}$ (see [34]).
 // $c_p$, (92).
 c_p = (1.01787 + 1.70736E−3*T)*kJ_kgK;
 // $\varrho$, (93).
 rho = 1.2691E3 − 1.52115*T + 1.79133E−3*T^2 − 1.67145E−6*T^3;
 // $\varrho_T = \partial/\varrho\partial T$, (94).
 rho_T = −1.52115 + 2*1.79133E−3*T − 3*1.67145E−6*T^2;
 // $\alpha_p$, (90).
 alpha_p = −1/rho*rho_T;
 // $\dot{s}$, (87).
 der(s) = c_p/T*der(T) − v*alpha_p*der(p);
 // $\dot{g}$, (89).
 der(g) = −s*der(T) + v*der(p);
end Syltherm800;
```

Listing 13. **Thermodynamics.Substances.Sylterm800.**

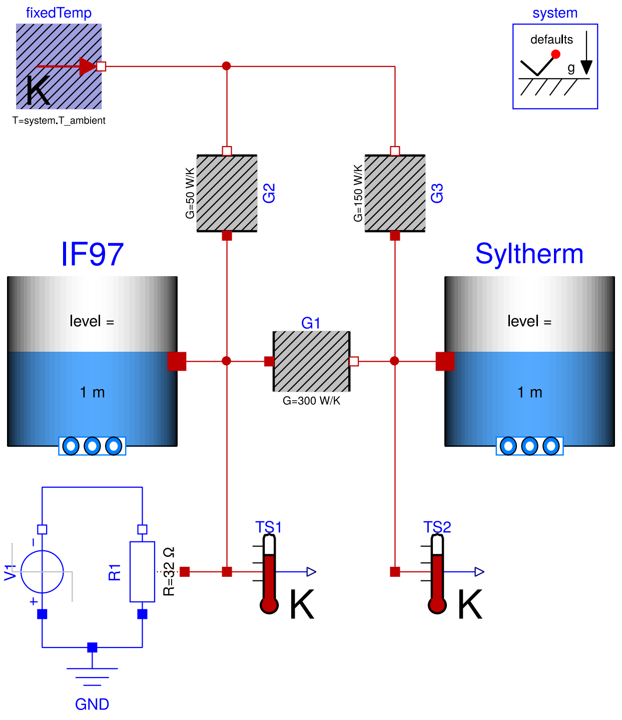


FIGURE 4. **Implementation with MSL of a system consisting of two open tanks at ambient conditions (20 °C, 1 atm) connected through a thermal conductor G_1, and heated with the electrical resistor R_1.**

$$\frac{\partial\varrho}{\partial T} = -1.52115 + 2 \times 1.79133^{-3} T - 3 \times 1.67145 \times 10^{-6} T^2 \text{ kg/(m}^3\text{·K)}. \quad (94)$$

The class Syltherm800 derived from SympleSubstance and models the thermodynamic behavior of Syltherm 800 (see Listing 13).

V. EXAMPLES

The examples in this section have been processed and simulated with Dymola running in a Linux virtual machine,

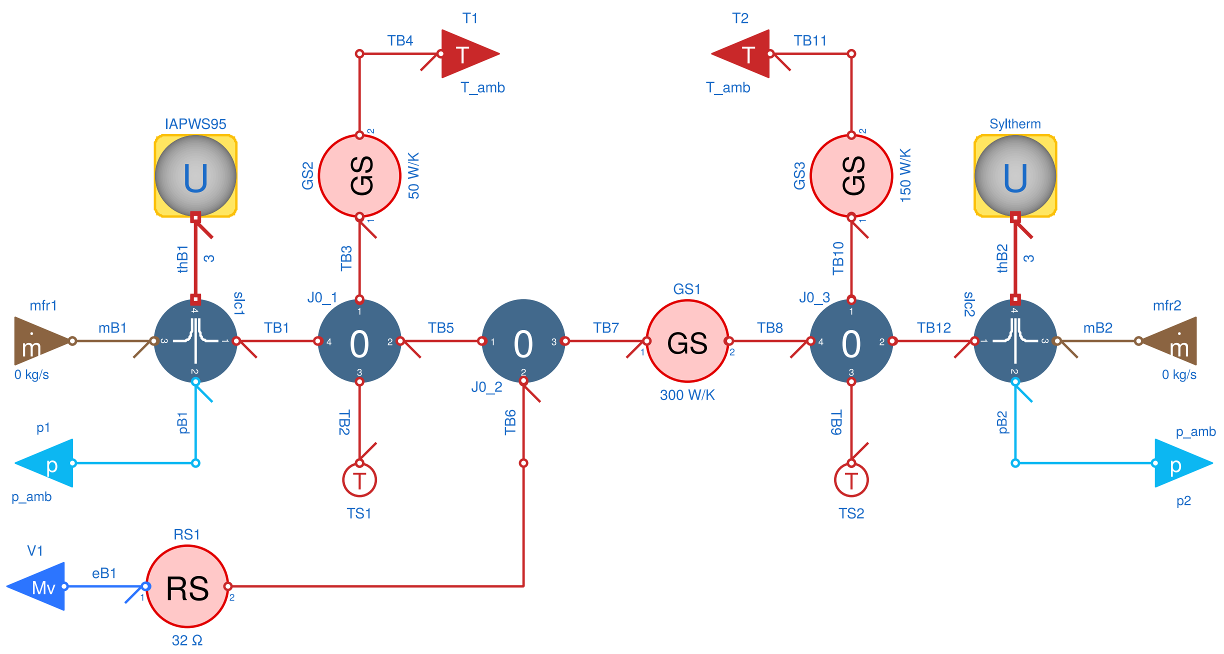


FIGURE 5. Implementation with the library pHlib of the two-tank system previously modeled with MSL (see Figure 4).

which was installed on an iMac computer configured with a 2.66 GHz Intel Core 2 Duo processor and 8 GB of memory.

A. COMPARING pHlib AND MSL FOR ACCURACY

Figure 4 shows the MSL implementation of a system consisting of two open tanks at ambient conditions (20 °C, 1 atm) connected through a thermal conductor $G_1 = 300\,\mathrm{W/K}$. The first tank contains $1\,\mathrm{m}^3$ of water modeled with the MSL IAPWS-IF97 implementation for industrial use [21] and it has thermal losses with its surroundings modeled with the thermal conductor $G_2 = 50\,\mathrm{W/K}$. The second tank also contains $1\,\mathrm{m}^3$, in this case of Syltherm 800, modeled with the MSL library from data supplied by the manufacturer [15] and it has thermal losses with its surroundings modeled with the thermal conductor $G_3 = 150\,\mathrm{W/K}$.

The water tank is heated for 60 hours with the heat generated by the electrical resistor $R_1 = 32\,\Omega$ and then allowed to cool for another 60 hours. For this purpose, we connect a modulated voltage generator $V_1 = 400\,\mathrm{V}$ to R_1 and make the assumptions: a) all the electrical power transformed into heat is transferred to the water tank, b) we do not consider the variations that R_1 may have with temperature. We use thermal conductors and ideal resistors to focus our study on the modeling of the thermodynamic behavior of water and Syltherm substances.

Figure 5 shows the implementation of the same system with the library pHlib where the proposed port-Hamiltonian representation makes it easier to visualize the connection network for energy exchange between the different components of the system.

In this figure we can see the following components (see Figure 3 and [28, Figures 9 and 12] for a detailed explanation of the symbols):

- The tank water modeled by:
  - the storage element IAPWS95,
  - the boundary conditions modeled by the constant temperature source T1, the constant pressure source p1 and the constant mass flow rate source mfr1,
  - the three-port slicer slc1 that separates the 3-dimension thermodynamic port #4 into three 1-dimension ports (thermal port #1, pressure port #2, and mass port #3),
  - the thermal conductance GS2 thermal modeling losses to the environment,
  - the temperature sensor TS1, and
  - the 0-junction J0_1.
- The Syltherm 800 tank, with a similar structure to the water tank, and modeled by the storage element Syltherm.
- The thermal conductance GS1 connecting both tanks.
- The modulated voltage generator V1 connected to the electrical resistor RS1 for heating the water tank.

The thermodynamic behavior of water has been modeled with the class WaterIAPWS95 (see Section IV-F) that implements the equations for general and scientific use [22], and the behavior of Syltherm with the class Sytherm800 (see Section IV-G). The thermal conductors $GS_{1,2,3}$ and the electrical resistor RS_1 have been modeled as entropy generating elements derived from the class pHlib.Dissipative.RS (see [28, Remark 26] and Listing 14).

The implementation of RS satisfaying that the equation $\langle e|f\rangle + \langle T_2|\dot{S}_2\rangle = 0$, guarantees the conservation of energy during the heat conduction process. Thus, the incoming power $P_1 = \langle e|f\rangle$ is equal to the outgoing heat flow rate $\dot{U}_2 = \langle T_2|\dot{S}_2\rangle$. However, in the transformation process, there

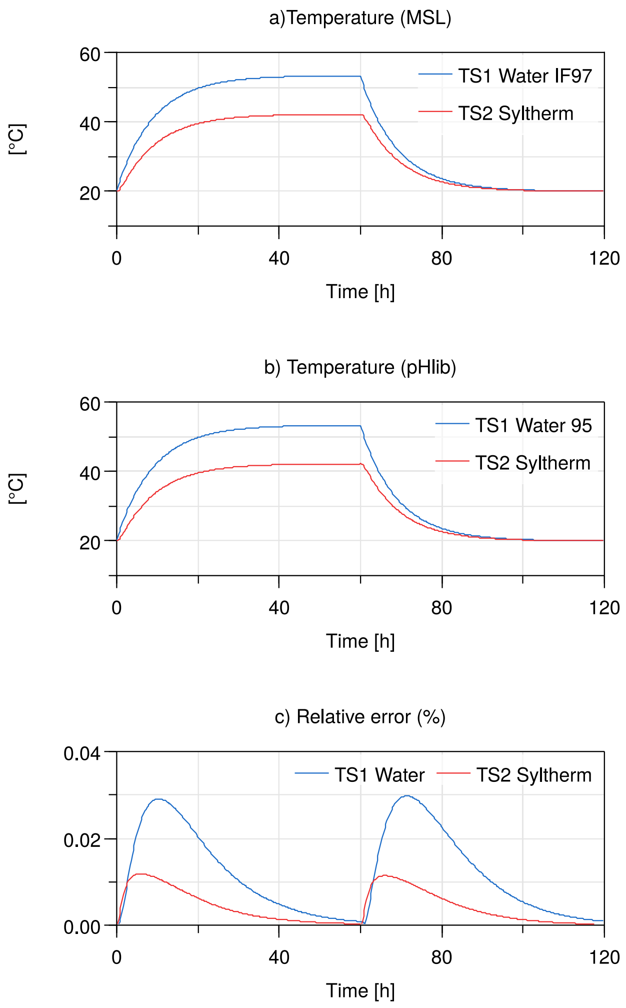


FIGURE 6. Evolution of water and Syltherm temperature during a 120-hour simulation for MSL (a) and pHlib (b) models, and evolution of the relative error (c).

will be an entropy generation to be modeled in each particular class derived from **RS**.

The heat flow rate between the ends of a thermal conductor of conductance G at temperatures T_1 and T_2 is

$$\dot{U}_{\text{cond}} = G \cdot (T_1 - T_2), \tag{95}$$

and the equations for calculating the entropy flow rate $\dot{S}_{\text{cond}}$ generated in the conduction process are [4, Chap.2]

$$\dot{S}_{\text{cond}} = G\frac{(T_1 - T_2)^2}{T_1 \cdot T_2} \geq 0, \tag{96}$$

$$\begin{cases} \dot{S}_2 - \dot{S}_1 = \dot{S}_{\text{cond}} & \text{if } T_1 > T_2, \\ \dot{S}_1 - \dot{S}_2 = \dot{S}_{\text{cond}} & \text{if } T_1 < T_2. \end{cases} \tag{97}$$

These equations are implemented in the classes **MGS** and **GS** (see Listing 15).

We consider that all the electrical energy consumed by an ideal linear resistor of resistance R is transferred to the thermal port, so the entropy flow rate in this port is

$$\dot{S}_2 = \frac{R \cdot i^2}{T} = \frac{v^2}{R \cdot T} \geq 0. \tag{98}$$

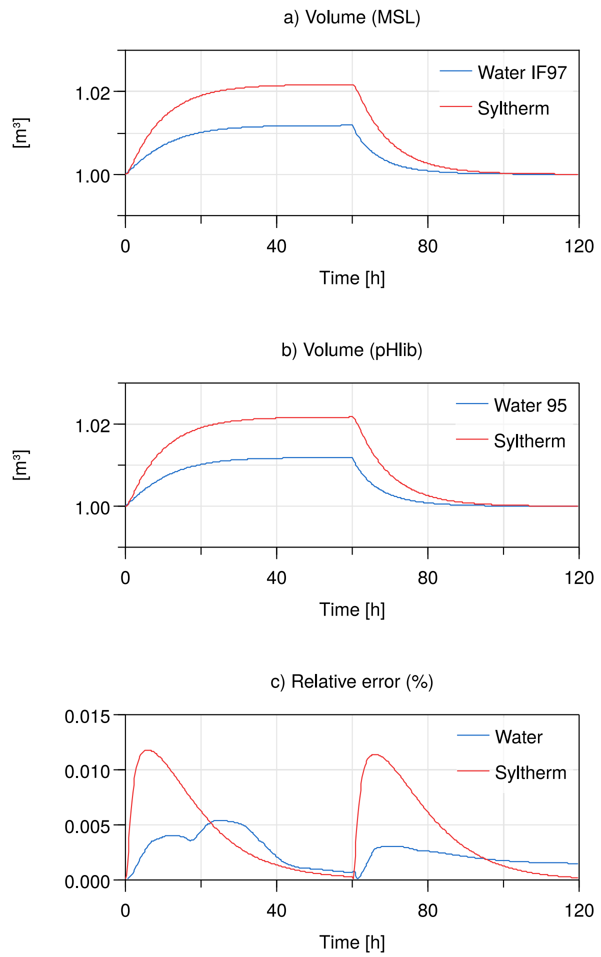


FIGURE 7. Evolution of water and Syltherm volume during a 120-hour simulation for MSL (a) and pHlib (b) models, and evolution of the relative error (c).

In part a) of Figures 6–8, we can see the temperature, volume, and density evolution of water and Syltherm during the 120-hour simulation of the MSL model. In part b) we can see the evolution of the same system modeled with **pHlib**.

We can see that both models show qualitatively similar behavior. For a quantitative comparison, we have taken the MSL model as a reference and calculated the relative errors of the **pHlib** model with the expression

$$\text{error}_{\%}\left(X_{\text{pHlib}}\right) = \frac{\left|X_{\text{MSL}} - X_{\text{pHlib}}\right|}{|X_{\text{MSL}}|} \cdot 100, \quad X \in \{T, V, \varrho\}. \tag{99}$$

In part c) of Figures 6–8, we can see the time evolution of the relative error. For water, the errors are below 0.03% in temperature and below 0.005% in volume and density. For Syltherm, the maximum error is only slightly above 0.01% in all three quantities.

B. COMPARING pHlib AND MSL FOR PERFORMANCE AND SCALABILITY

Figure 9 shows a system consisting of $N = 5$ open water tanks at ambient conditions (20 °C, 1 atm) connected through

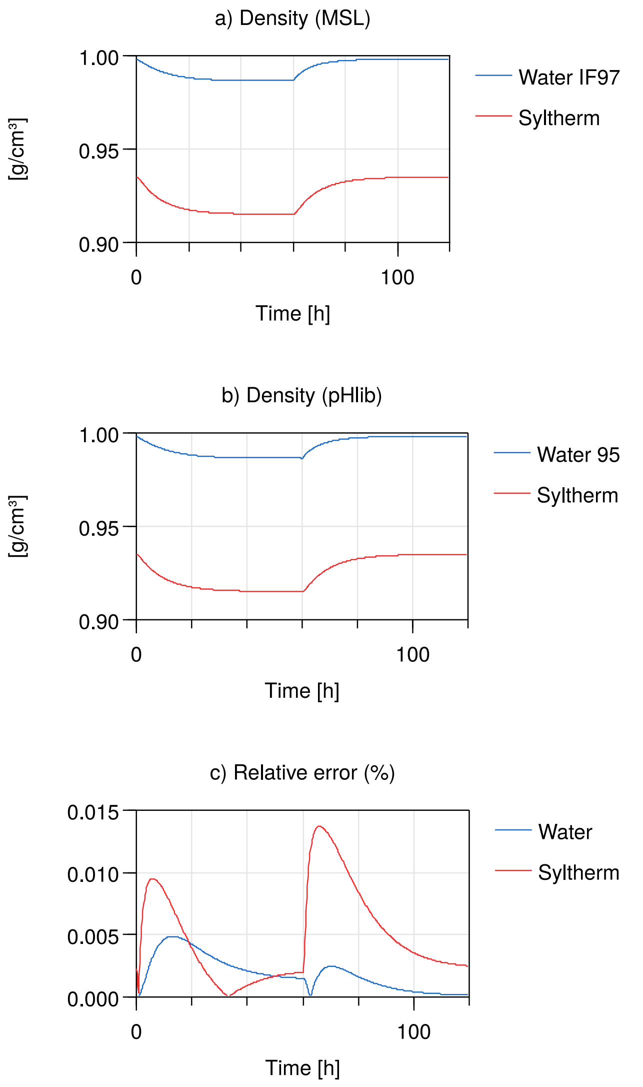


FIGURE 8. Evolution of water and Syltherm density during a 120-hour simulation for MSL (a) and pHlib (b) models, and evolution of the relative error (c).

```
partial class RS "Resistor with entropy generation"
  replaceable Real e[port1.dimPort] = port1.e;
  replaceable Real f[port1.dimPort] = port1.f;
  SI.Temperature T2 = port2.e[1];
  SI.EntropyFlowRate dot_S2 = port2.f[1];
  Real PhiR = 0; // Φ_R(e, f, T_2, Ṡ_2) = 0.
  extends Partial.TwoPorts(
    redeclare Thermodynamics.Interfaces.Ports.ThermalPort port2);
  //
equation
  e*f*port1.dir + T2*dot_S2*port2.dir = 0; // ⟨e|f⟩ + ⟨T_2|Ṡ_2⟩ = 0.
end RS;
```

Listing 14. pHlib.Dissipative.RS.

```
package Dissipative
 package Modulated
  class MGS "Modulated thermal conductor"
   replaceable SI.ThermalConductance G;
   SI.Temperature T1 = port1.e[1];
   SI.EntropyFlowRate dot_S1 = port1.f[1];
   extends pHS.Dissipative.RS (redeclare ThermalPort port1);
  equation
   // Φ_R(T_1, Ṡ_1, T_2, Ṡ_2) = 0. See (97).
   PhiR = dot_S2 − dot_S1 − G*(T1−T2)^2/(T1*T2);
  end MGS;
 end Modulated;

 package Constant
  class GS "Constant thermal conductor"
    extends Modulated.MGS(
    redeclare replaceable parameter SI.ThermalConductance G);
  end GS;
 end Constant;
end Dissipative;
```

Listing 15. Thermodynamics.Dissipative.

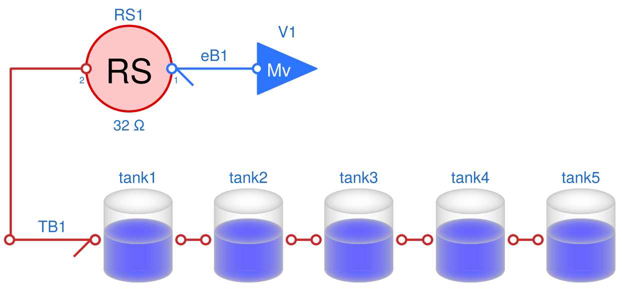


FIGURE 9. System of $N = 5$ open water tanks at ambient conditions (20 °C, 1 atm) connected through thermal conductors.

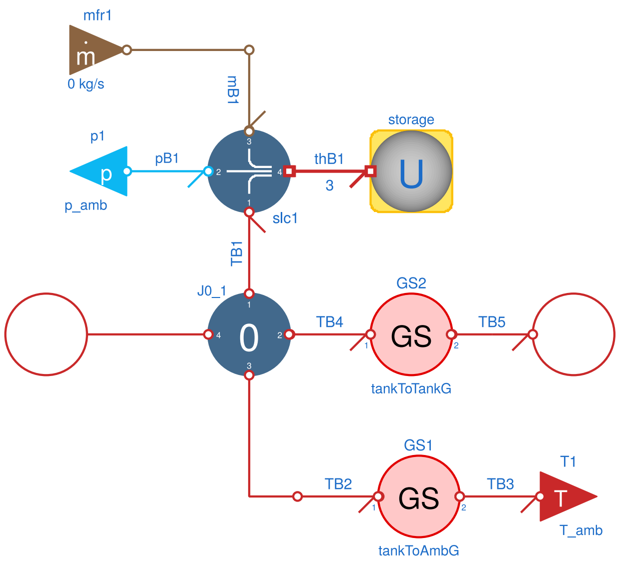


FIGURE 10. Model of each tank in Figure 9.

thermal conductors. Figure 10 shows the model of each tank. The system parameters are the number N of tanks, the mass of water stored in each tank, its initial volume, the thermal conductance between tanks tankToTankG, and the thermal conductance with the environment tankToAmbG. The model used for water is IAPWS-95. Resistor R1 connected to the modulated voltage generator V1 heats the first tank for 60 hours, and then we let the tanks cool for another 60 hours. This system has also been implemented with MSL and the IAPWS-IF97 water model. Both models generate $2 \times N$ state variables.

Table 3 shows the execution times when simulating each model for some values of N. The results show that simulations

TABLE 3. Performance comparison pHlib (IAPWS-95)–MSL (IAPWS-IF97).

N	t_{pHlib}	t_{MSL}	$t_{\text{pHlib}}/t_{\text{MSL}}$
5	0.71	0.087	8.2
10	1.48	0.156	9.5
20	2.72	0.245	11.1
30	4.32	0.253	17.1
40	7.04	0.378	18.6
50	9.35	0.411	22.7

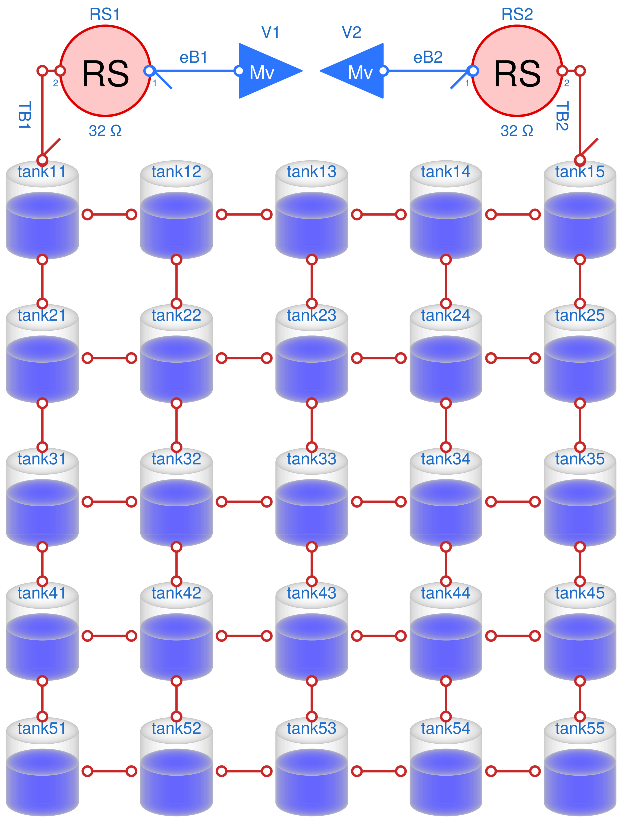


FIGURE 11. System of 5 × 5 open water tanks at ambient conditions (20 °C, 1 atm) connected through thermal conductors.

with the IAPWS-IF97 model are faster than simulations with the IAPWS-95 model. These results are expected because the industrial formulation simplifies the scientific formulation to increase computational speed in a tradeoff for lower accuracy (for more details on this subject see [21], [52]).

These results might seem to suggest that IAPWS-95 models are not very suitable for real-time applications. However, note that the 120-hour simulation for the 50 tanks system took only 9.35 seconds. This system produces a model with 100 state variables, which is common in modeling complex industrial systems. For this reason, we think that improvements in hardware performance over the last two decades make it possible to use the IAPWS-95 formulation in real-time applications. The IAPWS-95 model also has the advantage of formulating the fundamental equation of water with a single continuous and differentiable expression, unlike IAPWS-IF97, which is forced to split the formulation into

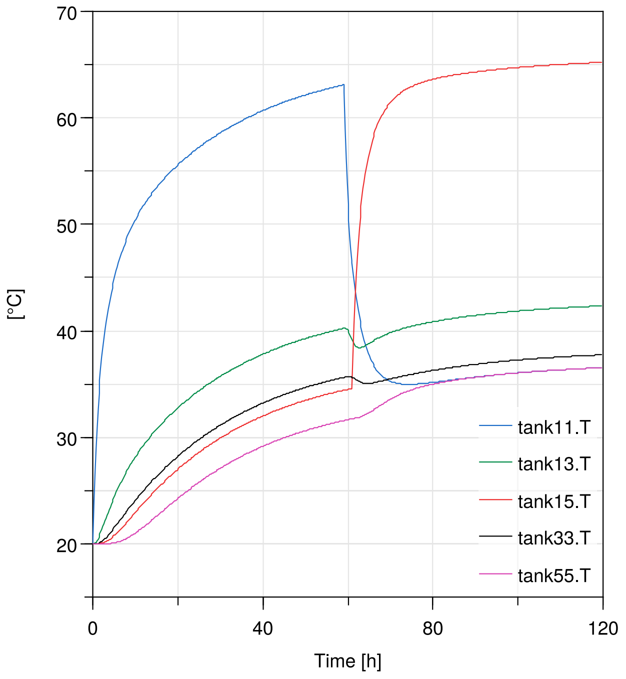


FIGURE 12. Evolution of water temperature in tanks 11, 13, 15, 33 and 55 during a 120-hour simulation (see Figure 11).

five regions, each with its corresponding fundamental equation. This partitioning complicates the construction of models with region changes and it can even lead to convergence and stability problems at the region boundaries due to chattering phenomena (see [3] for details).

C. ARRAY OF 5 × 5 TANKS

Figure 11 shows a system consisting of an array with 5 open water tanks at ambient conditions (20 °C, 1 atm) connected through thermal conductors. Each tank has 250 kg of water, 10 W/k of thermal conductance with the environment, and 250 W/K of thermal conductance with each of its neighbors. Resistors **R1** and **R2** heat the system for 120 hours. **V1** is a voltage generator that generates 400 V during the first 59 hours, and **V2** generates 400 V during the last 59 hours. Therefore, during the two central hours, the resistors do not supply heat to the system.

Figure 12 shows the temperature evolution of some tanks that have been chosen to check the consistency of the model. Given the symmetry of the system, some temperatures have to converge to the same value as is the case for the temperatures of tanks 11 and 55. Other temperatures have to maintain a relationship of order, e.g., the temperature of tank 13 has to be higher than that of tank 33 because it is closer to the heat sources.

VI. CONCLUSION AND FUTURE WORKS

This work is a continuation of [28] where we presented a general purpose Modelica library for modeling multiphysics systems with a port-Hamiltonian approach. In the conclusions of that article, we proposed the extension of this library as a future research line for ‘‘modeling of thermofluid systems and its assessment in the development of models for large-scale solar thermal power plants''.

In the current article, we have extended the library with the proposed line of research and presented the added components for modeling thermodynamic systems. This is a first step in approaching the modeling of thermofluid systems. We will complete the modeling and the library in a future paper studying the transport phenomena in flowing substances.

The object-oriented design of the library (taking advantage of the resources offered by the Modelica language) has allowed us to build general substance models and to refine them in a simple way to model specific substances such as water or Syltherm 800.

The proposed symbols which accompany each component allow the construction of complex system models that are easy to read for any designer familiar with bond graphs.

When contrasting the simulation results of the two-tank model with the results obtained for the same model implemented with MSL, we have found that they coincide with relative errors lower than 0.005% in some variables, even though both libraries are designed with different methodologies. This result validates the choice of the port-Hamiltonian approach and Classical Irreversible Thermodynamics to model the behavior of thermodynamic systems. The suitability of the port-Hamiltonian formulation as a multiphysics modeling tool of great theoretical and practical value has been illustrated, according to the following aspects: the mathematical rigor of the port-Hamiltonian formalism built on differential geometry, the fundamental role played by energy in the definition of port-Hamiltonian systems, and the ease of modeling complex systems by connecting simpler ones that exchange energy with each other.

The performance measurements evaluated on the N tanks example show that the system modeled with IAPWS-IF97 (MSL library) is faster than the one modeled with IAPWS-95 (pHLib library). The pHlib library, due to its more academic port-Hamiltonian approach, will be mainly useful in scientific research. However, the response times measured for different values of N indicate that modern computers can cope with real-time simulations of large systems in industrial applications. This result encourages us to continue with the research initiated in [28] to complete the library pHlib with the modeling of transport phenomena in thermofluid systems and to study its application to the modeling of complex systems such as solar thermal power plants.

APPENDIX

ACRONYMS AND NOMENCLATURE

A. ACRONYMS

CEA	Chemical Equilibrium with Applications
HTF	Heat Transfer Fluid
IAPWS	International Association for the Properties of Water and Steam
ISO	Input State Output
MSL	Modelica Standard Library

B. NOMENCLATURE

Quantity	Name	SI units
c_p	Isobaric heat capacity	J/(kg·K)
c_v	Isochoric heat capacity	J/(kg·K)
$\langle e\|f\rangle$	Dual product ⟨effort|flow⟩	W
F	Helmholtz free energy	J
f	Specific Helmholtz free energy	J/kg
G	Gibbs free energy	J
g	Specific Gibbs free energy	J/kg
g_j	Chemical potential	J/kg
H	Enthalpy	J
H	Hamiltonian (Section III)	J
h	Specific enthalpy	J/kg
m	Mass	kg
$M_{\mathrm{mol},j}$	Molar mass	kg/mol
n_j	Mole number	mol
p	Pressure	Pa
R	Specific gas constant	J/(kg·K)
R_{mol}	Molar gas constant	J/(mol·K)
S	Entropy	J/K
s	Specific entropy	J/(kg·K)
T	Temperature	K
$T\mathcal{X}$	Tangent bundle of $\mathcal{X}$	—
$T^*\mathcal{X}$	Cotangent bundle of $\mathcal{X}$	—
U	Internal energy	J
u	Specific internal energy	J/kg
V	Volume	m^3
$v = 1/\varrho$	Specific volume	m^3/kg
$w_j = m_j/m$	Mass fraction	1
$\mathcal{X}$	Smooth manifold $\mathcal{X}$	—
α_p	Isobaric expansibity	K^{-1}
α_s	Adiabatic expansibity	K^{-1}
$\gamma = c_p/c_v$	Heat capacity ratio	1
κ_s	Adiabatic compressibility	Pa^{-1}
κ_T	Isothermal compressibility	Pa^{-1}
μ_j	Chemical potential	J/mol
$\varrho = 1/v$	Density	kg/m^3

REFERENCES

[1] R. Abraham and J. E. Marsden, *Foundations of Mechanics*, 2nd ed. Reading, MA, USA: The Benjamin/Cummings, 1978.

[2] R. A. Alberty, "Use of Legendre transforms in chemical thermodynamics (IUPAC technical report)," *Pure Appl. Chem.*, vol. 73, no. 8, pp. 1349–1380, Aug. 2001.

[3] J. Bonilla, L. J. Yebra, and S. Dormido, "A heuristic method to minimise the chattering problem in dynamic mathematical two-phase flow models," *Math. Comput. Model.*, vol. 54, nos. 5–6, pp. 1549–1560, Sep. 2011.

[4] W. Borutzky, *Bond Graph Methodology–Development and Analysis of Multidisciplinary Dynamic System Models*. London, U.K.: Springer-Verlag, 2010.

[5] W. Borutzky, Ed., *Bond Graph Modelling of Engineering Systems. Theory, Applications and Software Support*. New York, NY, USA: Springer-Verlag, 2011.

[6] H. B. Callen, *Thermodynamics*. Hoboken, NJ, USA: Wiley, 1960.

[7] H. B. Callen, *Thermodynamics Introduction to Thermostatistics*, 2nd ed. Hoboken, NJ, USA: Wiley, 1985.

[8] F. E. Cellier, *Continuous System Modeling*. New York, NY, USA: Springer-Verlag, 1991.

[9] F. E. Cellier and J. Greifeneder, "ThermoBondLib—A new modelica library for modeling convective flows," in *Proc. 6th Int. Modelica Conf.* Bielefeld, Germany: Univ. of Applied Sciences, 2008, pp. 163–172.

[10] F. E. Cellier and A. Nebot, "The Modelica bond braph library," in *Proc. 4th Int. Modelica Conf.*, G. Schmitz, Ed. Hamburg, Germany: Hamburg Univ. of Technology, 2005, pp. 57–65.

[11] T. J. Courant, "Dirac manifolds," *Trans. Amer. Math. Soc.*, vol. 319, no. 2, pp. 631–661, 1990.

[12] S. R. de Groot and P. Mazur, *Non-Equilibrium Thermodynamics*. Amsterdam, The Netherlands: North-Holland, 1962.

[13] A. de la Calle, F. E. Cellier, L. J. Yebra, and S. Dormido, "Improvements in BondLib, the modelica bond graph library," in *Proc. 8th Congr. Modeling Simulation (EUROSIM)*, Sep. 2013, pp. 282–287.

[14] A. Donaire and S. Junco, "Derivation of input-state-output Port-Hamiltonian systems from bond graphs," *Simul. Model. Pract. Theory*, vol. 17, no. 1, pp. 137–151, Jan. 2009.

[15] *SYLTHERM 800 Heat Transfer Fluid*, Dow Chemical Company, Lansing, MI, USA, 1997.

[16] V. Duindam, A. Macchelli, S. Stramigioli, and H. Bruyninckx, Eds., *Modeling and Control of Complex Physical Systems. The Port Hamiltonian Approach*. Berlin, Germany: Springer, 2009.

[17] H. Elmqvist, H. Tummescheit, and M. Otter, "Object-oriented modeling of thermo-fluid systems," in *Proc. 3rd Int. Modelica Conf.* Stockholm, Sweden: Linköping Univ., 2003, pp. 269–286.

[18] P. Fritzson, *Principles of Object-Oriented Modeling and Simulation With Modelica 3.3: A Cyber-Physical Approach*, 2nd ed. Hoboken, NJ, USA: Wiley, 2015.

[19] P. Fritzson, A. Pop, K. Abdelhak, A. Asghar, B. Bachmann, W. Braun, D. Bouskela, R. Braun, L. Buffoni, F. Casella, and R. Castro, "The Open-Modelica integrated environment for modeling, simulation, and model-based development," *Model., Identificat. Control A, Norwegian Res. Bull.*, vol. 41, no. 4, pp. 241–295, 2020.

[20] G. Golo, A. van der Schaft, P. C. Breedveld, and B. C. Maschke, "Hamiltonian formulation of bond graphs," in *Nonlinear and Hybrid Systems in Automotive Control*, R. Johansson and A. Rantzer, Eds. London, U.K.: Springer-Verlag, 2003, ch. 16, pp. 351–372.

[21] IAPWS, "Revised release on the IAPWS industrial formulation 1997 for the thermodynamic properties of water and steam," Int. Assoc. Properties Water Steam, Lucerne, Switzerland, Tech. Rep. IAPWS R7-97 (2012), 2007.

[22] IAPWS, "Revised release on the IAPWS formulation 1995 for the thermodynamic properties of ordinary water substance for general and scientific use," Int. Assoc. Properties Water Steam, Prague, Czech Republic, Tech. Rep. IAPWS R6-95, 2018.

[23] D. Jou, J. Casas-Vázquez, and G. Lebon, *Extended Irreversible Thermodynamics*, Dordrecht, The Netherlands: Springer, 2010.

[24] D. Karnopp, D. Margolis, and R. Rosenberg, *System Dynamics: Modeling, Simulation, and Control of Mechatronic Systems*, 5th ed. Hoboken, NJ, USA: Wiley, 2012.

[25] A. N. Kaufman, "Dissipative Hamiltonian systems: A unifying principle," *Phys. Lett. A*, vol. 100, no. 8, pp. 419–422, Feb. 1984.

[26] Y. Koga, *Solution Thermodynamics and its Application to Aqueous Solutions. A Differential Approach*. Amsterdam, The Netherlands: Elsevier, 2007.

[27] L. D. Landau and E. M. Lifshitz, *Statistical Physics. Part 1 of Course of Theoretical Physics*, 3rd ed., vol. 5. Oxford, U.K.: Butterworth, 1980.

[28] F. M. Marquez, P. J. Zufiria, and L. J. Yebra, "Port-Hamiltonian modeling of multiphysics systems and object-oriented implementation with the modelica language," *IEEE Access*, vol. 8, pp. 105980–105996, 2020.

[29] M. Materassi, "Entropy as a metric generator of dissipation in complete metriplectic systems," *Entropy*, vol. 18, no. 8, p. 304, Aug. 2016.

[30] B. J. McBride, M. J. Zehe, and S. Gordon, "NASA Glenn coefficients for calculating thermodynamic properties of individual species," Nat. Aeronaut. Space Admin., Glenn Res. Center, Cleveland, OH, USA, Tech. Rep. NASA/TP-2002-211556, 2002.

[31] P. J. Mohr and B. N. Taylor, "CODATA recommended values of the fundamental physical constants: 1998," *Rev. Modern Phys.*, vol. 72, no. 2, pp. 351–495, Apr. 2000.

[32] P. J. Morrison, "Bracket formulation for irreversible classical fields," *Phys. Lett. A*, vol. 100, no. 8, pp. 423–427, Feb. 1984.

[33] P. J. Morrison, "A paradigm for joined Hamiltonian and dissipative systems," *Phys. D, Nonlinear Phenomena*, vol. 18, nos. 1–3, pp. 410–419, Jan. 1986.

[34] A. Mwesigye, T. Bello-Ochende, and J. P. Meyer, "Numerical investigation of entropy generation in a parabolic trough receiver at different concentration ratios," *Energy*, vol. 53, pp. 114–127, May 2013.

[35] H. Olsson, *Modelica—A Unified Object-Oriented Language for Systems Modeling. Language Specification. Version 3.5*. Linköping, Sweden: Modelica Association, 2021. [Online]. Available: https://www.modelica.org/documents/MLS.pdf

[36] L. Onsager, "Reciprocal relations in irreversible processes. I," *Phys. Rev.*, vol. 37, no. 4, pp. 405–426, 1931.

[37] L. Onsager, "Reciprocal relations in irreversible processes. II," *Phys. Rev.*, vol. 38, no. 12, pp. 2265–2279, Dec. 1931.

[38] G. Oster, A. Perelson, and A. Katchalsky, "Network thermodynamics," *Nature*, vol. 234, no. 5329, pp. 393–399, Dec. 1971.

[39] H. C. Öttinger, *Beyond Equilibrium Thermodynamics*. Hoboken, NJ, USA: Wiley, 2005.

[40] H. M. Paynter, *Analysis and Design of Engineering Systems*. Cambridge, MA, USA: MIT Press, 1961.

[41] A. S. Perelson, "Network thermodynamics. An overwiew," *Biophysical J.*, vol. 15, pp. 667–685, Jul. 1975.

[42] M. Pfeifer, S. Caspart, S. Hampel, C. Müller, S. Krebs, and S. Hohmann, "Explicit Port-Hamiltonian formulation of multi-bond graphs for an automated model generation," *Automatica*, vol. 120, Oct. 2020, Art. no. 109121.

[43] I. Prigogine, *Introduction to Thermodynamics of Irreversible Processes*, 3rd ed. New York, NY, USA: InterScience, 1967.

[44] R. Rashad, F. Califano, A. J. van der Schaft, and S. Stramigioli, "Twenty years of distributed Port-Hamiltonian systems: A literature review," *IMA J. Math. Control Inf.*, vol. 37, no. 4, pp. 1400–1422, Dec. 2020.

[45] J. Thoma and B. O. Bouamama, *Modelling and Simulation in Thermal and Chemical Engineering. A Bond Graph Approach*. Berlin, Germany: Springer-Verlag, 2000.

[46] A. Thompson and B. N. Taylor, "Guide for the use of the international system of units (SI)," Nat. Inst. Standards Technol., Gaithersburg, MD, USA, NIST Special Publication 811, 2008.

[47] C. Truesdell, *Rational Thermodynamics*, 2nd ed. New York, NY, USA: Springer–Verlag, 1984.

[48] N. W. Tschoegl, *Fundamentals Equilibrium Steady-State Thermodynamics*. Amsterdam, The Netherlands: Elsevier, 2000.

[49] A. van der Schaft, *L_2-Gain and Passivity Techniques in Nonlinear Control*. London, U.K.: Springer-Verlag, 2000.

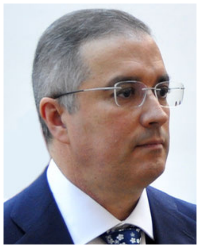
[50] A. van der Schaft, *L_2-Gain and Passivity Techniques in Nonlinear Control*, 3rd ed., vol. 11. Cham, Switzerland: Springer-Verlag, 2017.

[51] A. van der Schaft and B. Maschke, "Geometry of thermodynamic processes," *Entropy*, vol. 20, no. 12, p. 925, Dec. 2018.

[52] W. Wagner, J. R. Cooper, A. Dittmann, J. Kijima, H.-J. Kretzschmar, A. Kruse, R. Mareš, K. Oguchi, H. Sato, I. Stöcker, O. Šifner, Y. Takaishi, I. Tanishita, J. Trübenbach, and T. Willkommen, "The IAPWS industrial formulation 1997 for the thermodynamic properties of water and steam," *J. Eng. Gas Turbines Power*, vol. 122, no. 1, pp. 150–184, Jan. 2000.

[53] W. Wagner and A. Pruß, "The IAPWS formulation 1995 for the thermodynamic properties of ordinary water substance for general and scientific use," *J. Phys. Chem. Ref. Data*, vol. 31, no. 2, pp. 387–535, 2002.

[54] D. Zimmer and F. E. Cellier, "The Modelica multi-bond graph library," in *Proc. 5th Int. Modelica Conf.*, Vienna, Austria, 2006, pp. 559–568.



FRANCISCO M. MÁRQUEZ received the B.S. degree in electronics engineering from the Universidad de Alcalá, Alcalá de Henares, Spain, in 1990, and the M.S. degree in telecommunication engineering and the M.S. degree in statistical and computational information processing from the Universidad Politécnica de Madrid, Madrid, Spain, in 2010 and 2011, respectively.

From 1989 to 1991, he was the Head of the laboratory at INDRA, Spain. Since 1991, he has been a Lecturer with the Departamento de Automática, UAH.

PEDRO J. ZUFIRIA received the Ingeniero de Telecomunicación degree from the Universidad Politécnica de Madrid (UPM), in 1986, the M.Sc. degree in ME, the M.Sc. degree in EE, the Ph.D. degree from the University of Southern California (USC), in 1989, the Doctor Ingeniero de Telecomunicación degree from MEC, in 1991, and the Licenciado en Ciencias Matemáticas degree from the Universidad Complutense de Madrid, in 1997.

From 1987 to 1990, he was a Teaching and Research Assistant with USC, in collaboration with different projects of the National Science Foundation and TRW/NASA. Since 1991, he has been holding a teaching position (currently as a Full Professor) at the Escuela Técnica Superior de Ingenieros de Telecomunicación (ETSIT), UPM, where he was the Chairman of the Departamento de Matemática Aplicada a las Tecnologías de la Información y las Comunicaciones, from 1999 to 2004. In the ETSIT, he was responsible for the Office of International Relations with other academic institutions, from 1993 to 1996. He was the Vice-Dean of the research and graduate studies, in 1999, and the Co-Director of the Orange Chair, from 2009 to 2019. Since 2019, he has been the Co-Director of Cabify Chair. He also has been the Founder and a Scientist responsible for the Neural Networks Group and currently the Dynamical Systems, Learning, and Control Research Group, where he has been involved in research projects of the European Union as well as several Spanish official and private institutions. He has authored over 100 international publications in his research field. His research interests include analysis, control, and fault diagnosis of dynamical systems, and the applications of statistics, machine learning paradigms, and complex network theory on system modeling and data processing.

LUIS J. YEBRA was born in Almería, Spain, in 1971. He received the degree in telecommunications technical engineering from the Universidad de Alcalá, Alcalá de Henares, Spain, in June 1993, the degree in physics from the Universidad Nacional de Educación a Distancia (UNED), Madrid, Spain, in June 1997, and the Ph.D. degree in automatic control and industrial computing from the Department of Automatic Control and Computer Science, UNED, in May 2006.

From 1999 to 2007, he was a Predoctoral Fellow with the National Research Center, CIEMAT, and a Researcher with the Department of Energy, CIEMAT in Plataforma Solar de Almería (PSA-CIEMAT). Since 2007, he has been a Tenured Scientist in dynamic modeling and automatic control with PSA-CIEMAT, where he has been mainly focused on concentrated solar thermal plants (CST). He has coauthored four books, more than 30 scientific articles published in international journals (JCR), and more than 70 international conference proceeding papers. He has supervised four Ph.D. degree students. He has participated in more than 30 publicly funded research projects, two technology transfer agreements, and the creation of a spin-off company.

Dr. Yebra has received three prizes to the best academic curriculum for his Physics degree.