SYSTEM INTERCONNECTION

Jan C. Willems

K.U. Leuven, B-3001 Leuven, Belgium
Jan.Willems@esat.kuleuven.be
www.esat.kuleuven.be/∼jwillems

1 Introduction

It is a pleasure to contribute an essay to this volume dedicated to Alberto Isidori on the occasion of his 65-th birthday. As the topic of my article, I chose an issue which is at the core of systems thinking, namely the formalization and the mathematization of system interconnection. This pertains to linear and nonlinear systems alike. In view of Alberto's early interest in foundational aspects of system theory, especially in the context of linear systems, and his later concentration on control problems for nonlinear systems, it is my intention to make this article a fitting tribute to his wide ranging scientific interests and to the influence that his work had in the field of systems and control theory.

Systems, physical and man-made alike, usually consist of interconnections of interacting subsystems. This feature is crucial in modeling, analysis, and synthesis. The notion of a dynamical system that took shape in the field of systems and control throughout the 20-th century is input/output based. This statement ignores the notion of a dynamical system as a 'flow', as used in mathematics, since we consider this setting totally inadequate as a general vantage point for modeling. The statement also ignore developments in computer science, were very subtle types of interactions have been put forward.

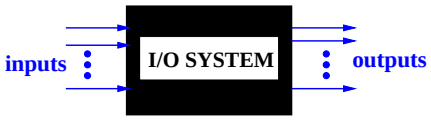


Fig. 1. Input/output system

The central idea in input/output thinking is that the environment acts on a system by imposing inputs, and that the system reacts by imposing outputs on its environment (see figure 1). This mental image also suggests the functioning of interconnec-

tions, by assigning outputs of one system to function as inputs to another system (see figure 2). These are very appealing ideas indeed, but the question should be examined if this is the mechanism by which the environment interacts with a system and if this is the way interconnections function in reality, physical and otherwise.

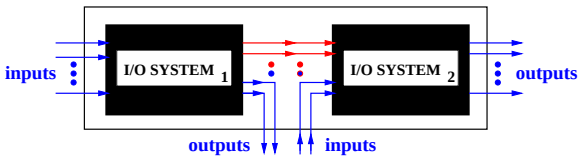


Fig. 2. Input/output interconnection

Early on, a system was regarded as an input/output *map*. This view is especially prevalent in signal processing, in classical control theory, and in Wiener's work. However, in all but the simplest examples, a dynamical input/output system is simply not a *map*. One can go a certain distance with this 'map' idea in the context of linear systems, say by assuming both the input and the output to be zero in the far past, or by restricting to square integrable inputs and outputs. But already this is very awkward, for example in connection with feedback or with unstable systems. However, the 'map' aspect is a totally inappropriate, indeed basically an impossible, starting point for nonlinear or discrete event systems.

Later on, state space systems came in vogue. By taking into consideration initial conditions, the state space point of view gives a much better vantage point to discuss dynamical models. Thus we arrived at

$$\dot{x} = f(x,u,t), \quad y = h(x,u,t)$$

as the mathematical structure on which system and control theory is based since the introduction of state models around 1960.

Both the classical input/output maps, as well as the modern input/state/output version, consider a system as an cause/effect relation. The input/output point of view led to signal flow graphs, and to system interconnection as *output-to-input assignment* (see figure 2). In control and in signal processing (and, but to a lesser extent, in circuit theory), signal flow diagrams combining series, parallel, and feedback connections are viewed as the standard way to deal with interconnections. Since an adder can be viewed as a input/output system, with two inputs and their sum as output, we end up with output-to-input assignment as the basic operation by which systems are interconnected. Since this also fits the classical picture of control as *feedback* so very well, it is this view that came to dominate the field of systems and control.

2 Tearing, zooming, and linking

From an applications point of view, the input/output framework is much more restrictive than one is often led to believe. The architecture formalized by the signal flow arrows is often viewed as essential for describing the interaction of a system with its environment. But, the opposite is actually the case, especially for the description of physical systems and for describing their interconnections. In many situations, signal flow graphs are unphysical, a figment of the imagination, cumbersome, and unnecessary. Sharing common variables is a much more key idea for system interconnection than output-to-intput connection.

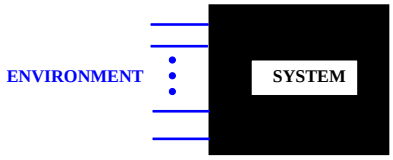


Fig. 3. Blackbox

A typical modeling task can be viewed as follows. Our aim is to model the dynamic behavior of a number of related variables. This is visualized by means of a blackbox (see figure 3) with a number of terminals. One should think of these terminals as 'places' where the variables which we set out to model 'live'. Sometimes one should take these terminals literally, sometimes not. In first instance, this only means that the modeler has declared what the variables of interest are: the terminals are merely a visualization. Often, thou, the terminals are real, and the aim is to model the variables associated with physical terminals through which a system interacts with its environment. When dealing with interconnections, it is natural to assume
(i) that these terminals and their variables are real physical entities, and
(ii) that there are usually *many* physical variables collectively and indivisibly associated with *one and the same* terminal.

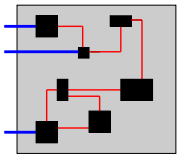


Fig. 4. Greybox

Most systems consist of interacting components. In order to discover these interactions, we look inside the blackbox of figure 3, where we find an interconnection architecture of 'smaller' blackboxes that interact through terminals of their own (see

figure 4). Modeling then proceeds by examining the smaller blackboxes and their interactions.

This modeling process is called *tearing, zooming, and linking* .

1. *Tearing* refers to viewing a system as an interconnection of smaller subsystems.
2. *Zooming* refers to modeling the subsystems.
3. *Linking* refers to modeling the interconnections.

There is an obvious hierarchical structure in this modeling process. Indeed, zooming involves modeling the (dynamic) laws that govern the variables on the terminals of a subsystem. This subsystem may in turn consist of interacting sub-sub-systems. Modeling the subsystem then again involves tearing, zooming, and linking. This goes on until we meet components whose model specification follows from first principles, or a subsystem whose model has been stored in a database, or where system identification is the modeling procedure that is called for.

The question which we examine in this paper is what actually happens when subsystems are interconnected. Our theme is that this does not (usually) imply *input-to-output assignment*, but rather *variable sharing*. However, in order to put these ideas in their proper setting, we briefly backtrack to the mathematical description of dynamical systems outside the input/output setting.

3 Behavioral systems

Over the last two decades, a framework for the study of systems has been developed that does not take the input/output structure as its starting point. The 'behavioral approach', as this has been called, simply identifies the dynamics of a system with a family of trajectories, called the *behavior*, and develops systems theory (including control) from there.

The behavioral framework views modeling as follows. Assume that we have a phenomenon that we wish to describe mathematically. Nature (that is, the reality that governs this phenomenon) can produce certain events (also called outcomes). The totality of possible events (*before* we have modelled the phenomenon) forms a set $\mathbb{U}$, called the *universum*. A *mathematical model* of the phenomenon restricts the outcomes that are declared possible to a subset $\mathscr{B}$ of $\mathbb{U}$; $\mathscr{B}$ is called the *behavior* of the model. We refer to $(\mathbb{U}, \mathscr{B})$ (or to $\mathscr{B}$ by itself, since $\mathbb{U}$ usually follows from the context) as a mathematical model.

As an example, consider the *ideal gas law*, which poses $PV = kNT$ as the relation between the pressure P, the volume V, the number N of moles, and the temperature T of an ideal gas, with k a universal physical constant. The universum $\mathbb{U}$ is $(\mathbb{R}_+)^4$, and the behavior $\mathscr{B} = \{(P,V,N,T) \in (\mathbb{R}_+)^4 \mid PV = kNT\}$.

In the study of dynamical systems we are more specifically interested in situations where the events are signals, trajectories, i.e. maps from a set of *independent variables* (time, in the present paper) to a set of *dependent variables* (the values taken on by the signals). In this case the universum is the collection of all maps from the

set of independent variables to the set of dependent variables. It is convenient to distinguish these sets explicitly in the notation: $\mathbb{T}$ for the set of independent variables, and $\mathbb{W}$ for the set of dependent variables. $\mathbb{T}$ suggests 'time', the case of interest in the present article. Whence a (dynamical) *system* is defined as a triple

$$\Sigma = (\mathbb{T}, \mathbb{W}, \mathscr{B})$$

with $\mathscr{B}$, *the behavior*, a subset of $\mathbb{W}^{\mathbb{T}}$ ($\mathbb{W}^{\mathbb{T}}$ is standard mathematical notation for the set of all maps from $\mathbb{T}$ to $\mathbb{W}$). The behavior is the central object in this definition. It formalizes which signals $w : \mathbb{T} \to \mathbb{W}$ are possible, according to the model: those in $\mathscr{B}$, and which are not: those not in $\mathscr{B}$. The behavioral framework treats a model for what it is: an exclusion law. Of course, in applications, the behavior $\mathscr{B}$ must be specified somehow, and it is here that differential equations (and difference equations for discrete-time systems) enter the scene.

In the equations describing a behavior, very often other variables appear in addition to those the model aims at. The origin of these auxiliary variables varies from case to case. They may be state variables (as in flows, automata, and input/state/output systems); they may be potentials (as in the well-known expressions for the solutions of Maxwell's equations); most frequently and most germane for the purposes of the present article, they are interconnection variables. It is important to incorporate these auxiliary variables in our modeling language *ab initio*, and to distinguish clearly between the variables whose behavior the model aims at, and the auxiliary variables introduced in the modeling process. The former are called *manifest* variables and the latter *latent* variables.

A *mathematical model with latent variables* is defined as a triple $(\mathbb{U}, \mathbb{L}, \mathscr{B}_{\text{full}})$, with $\mathbb{U}$ the universum of manifest variables, $\mathbb{L}$ the universum of latent variables, and $\mathscr{B}_{\text{full}} \subseteq \mathbb{U} \times \mathbb{L}$ the *full behavior*. It induces (or *represents*) the *manifest model* $(\mathbb{U}, \mathscr{B})$, with $\mathscr{B} = \{w \in \mathbb{U} \mid \text{there exists } \ell \in \mathbb{L} \text{ such that } (w, \ell) \in \mathscr{B}_{\text{full}}\}$. A (dynamical) *system with latent variables* is defined completely analogously as

$$\Sigma_{\text{full}} = (\mathbb{T}, \mathbb{W}, \mathbb{L}, \mathscr{B}_{\text{full}})$$

with $\mathscr{B}_{\text{full}} \subseteq (\mathbb{W} \times \mathbb{L})^{\mathbb{T}}$. The notion of a system with latent variables is the natural end-point of a modeling process and hence a very natural starting point for the analysis and synthesis of systems. More details and examples of behavioral systems may be found in [3, 4].

The procedure of modeling by *tearing, zooming, and linking* is an excellent illustration of the appropriateness of the behavioral approach. We assume throughout finiteness, i.e., a finite number of subsystems are interconnected, each with a finite number of terminals. Our view of interaction through terminals is certainly not the end point of the development of formalizing the interaction of systems. There are many interactions between subsystems that do not fit this 'terminal' paradigm: actions at a distance (as gravity), rolling and sliding, mixing, components that are interconnected through distributed surfaces, etc. Interconnecting systems through terminals fits very well lumped electrical and mechanical systems, many hydraulic systems, some thermal systems, etc. Interconnection via terminals also serves as a useful paradigm for more complex situations.

4 Formalization

In this section an outline is given of a formal procedure for obtaining a model by viewing a system (a blackbox) as an interconnection of subsystems (smaller blackboxes). The idea is to formalize the picture shown in figure 2: a finite number of systems are interconnected through terminals by to other subsystems. This suggests a graph with the subsystems in the nodes, and the interconnections in the edges. As we shall see, this formalism uses the notions of a behavior and of latent variables in an effective way.

The ingredients are:

1. terminals,
2. (parameterized) modules,
3. the interconnection architecture,
4. the module embedding, and
5. the manifest variable assignment.

4.1 Terminals

A *terminal* is specified by its *type*. Giving the type of a terminal identifies the kind of a physical terminal that we are dealing with. The type of terminal implies a universum of *terminal variables*. These variables are physical quantities that characterize the possible 'signal states' on the terminal, it specifies how the module interacts with the environment through this terminal. Some examples of terminals are given below.

Type of terminal	**Variables**	**Universum**
electrical	(voltage, current)	$\mathbb{R} \times \mathbb{R}$
1-D mechanical	(force, position)	$\mathbb{R} \times \mathbb{R}$
2-D mechanical	(position, attitude, force, torque)	$\mathbb{R}^2 \times [0, 2\pi) \times \mathbb{R}^2 \times \mathbb{R}$
thermal	(temperature, heat-flow)	$\mathbb{R}_+ \times \mathbb{R}$
fluidic	(pressure, mass-flow)	$\mathbb{R} \times \mathbb{R}$
input	u	$\mathbb{R}$
output	(y)	$\mathbb{R}$
etc.	etc.	etc.

Table 1. Examples of terminals

4.2 Modules

A *module* is specified by its *type*, and its *behavior*. Giving the type of a module identifies the kind of physical system that we are dealing with. Giving a *behavior specification* of a module implies giving a *representation* and the values of the *parameters* associated with a representation. Combined, these specify the behavior of

the variables on the terminals of the module. The type of a module implies an ordered set of terminals. Since each of the terminals comes equipped with a universum of terminal variables, we thus obtain an ordered set of variables associated with that module. The module behavior then specifies what time trajectories are possible for these variables. Thus a module defines a dynamical system $(\mathbb{R}, \mathbb{W}, \mathscr{B})$ with $\mathbb{W}$ the Cartesian product over the terminals of the universa of the terminal variables.

However, there are very many ways to specify a behavior (for example, as the solution set of a differential equation, as the image of a differential operator, through a latent variable model, through a transfer function, and many other ways). The behavioral representation picks out one of these. These representations will in first instance contain unspecified parameters (for example, the coefficients of the differential equation, or the rational functions in a transfer function). Giving the parameter values specifies their numerical values, and completes the specification of the behavior of the signals that are possible on the terminals of a module.

We give two examples. The first is a simple 3 ohm resistor (see figure 5).

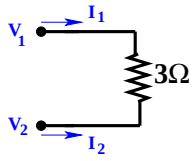


Fig. 5. A resistor

The module type is *ohmic resistor*. This means that it has two terminals, both of 'electrical' type, and that it is parameterized by a non-negative real number (the value of the resistor in ohms). Since the terminals are electrical, there are two variables, a voltage and a current (counted positive when it runs into the device), on each terminal. This yields in total four real variables associated with a resistor: (V_1, I_1) and (V_2, I_2). From the fact that we have an ohmic resistor, we know that the relation between these variables is

$$V_1 - V_2 = RI_1, \;\; I_1 + I_2 = 0.$$

Giving in addition the value of the parameter $R = 3$ of the Ohmic resistor leads to the behavioral equations

$$V_1 - V_2 = 3I_1, \;\; I_1 + I_2 = 0.$$

These equations completely specify the behavior of the terminal variables of a 3 ohm resistor.

Our second example of a module is a transfer function. The module type is *multivariable proper transfer function*. Its parameters are $(\mathtt{m}, \mathtt{p}, G)$, with $\mathtt{m}, \mathtt{p} \in \mathbb{N}$ and G a $\mathtt{p} \times \mathtt{m}$ matrix of proper real rational functions. This means that we have a system with

$\mathtt{m}+\mathtt{p}$ terminals, the first $\mathtt{m}$ of 'input' type, the last $\mathtt{p}$ of 'output' type, and behavior described by the controllable input/state/output system

$$\tfrac{d}{dt}x = Ax + Bu, \;\; y = Cx + Du,$$

with A,B,C,D such that $G(\xi) = D + C(I\xi - A)^{-1}B$. The behavior of this system consists of all $(u,y) : \mathbb{R} \to \mathbb{R}^{\mathtt{m}} \times \mathbb{R}^{\mathtt{p}}$ for which there exists $x : \mathbb{R} \to \mathbb{R}^{\mathtt{n}}$ such that these equations are satisfied: the state serves as a latent variable. Of course, to be precise, we would have to add some smoothness, but we will slide over these technical points, since they are not germane to the purposes of this article.

This representation of the module behavior requires specification of the numerical value of the state space system parameter matrices A,B,C,D. We have identified 'transfer function' with controllable linear time-invariant differential system. In this case there are very many other ways of translating this specification into dynamic equations. For example, by using left or right polynomial co-prime factorizations of the transfer function, we obtain differential equation representations in kernel or image form. By using factorizations with rational functions, we can obtain proper stable rational functions as parametrization. This class of systems, linear time-invariant differential systems, have been dealt with extensively in the literature.

Type of module	**Terminals**	**Type of terminals**
resistor	(terminal1, terminal2)	(electrical, electrical)
transistor	(collector, emitter, base)	(electrical, idem,idem)
mass, 2 applicators	(appl1, appl2)	(3-D mechanical, idem)
2-inlet vessel	(inlet1, inlet2)	(fluidic, fluidic)
heat exchanger	(inlet, outlet)	(fluidic-thermal, idem)
signal processor	(in, out)	(input, output)
etc.	etc.	etc.

Table 2. Examples of modules

Type of module	**Specification**	**Parameter**
resistor	default	R in ohms
$\mathtt{n}$-terminal circuit	transfer impedance	$G \in \mathbb{R}^{\mathtt{n}\times\mathtt{n}}(\xi)$
$\mathtt{n}$-port circuit	i/s/o admittance	(A,B,C,D)
bar, 2 applicators	Lagrangian equations	mass and length
2-inlet vessel	default	geometry
signal processor	kernel representation	$R \in \mathbb{R}[\xi]^{\bullet\times\bullet}$
signal processor	latent variable	(R,M)
etc.	etc.	etc.

Table 3. Examples of module specifications

Some general examples of modules types with their terminals and of behavioral specifications are given in the tables above.

A module Σ of a given type with T terminals yields the signal space $\mathbb{W} = \mathbb{W}_1 \times \mathbb{W}_2 \times \cdots \times \mathbb{W}_T$, with $\mathbb{W}_k$ the universum associated with the k-th terminal. The behavioral specification yields the behavior $\mathscr{B} \subseteq \mathbb{W}^{\mathbb{R}}$. If $(w_1, w_2, \ldots, w_T) \in \mathscr{B}$, then we think of the w_k's as signals $w_k : \mathbb{R} \to \mathbb{W}_k$ that can occur on the k-th terminal.

4.3 The interconnection architecture

The next element in the specification of a model is the *interconnection architecture* (or *interconnection graph*). This is defined as a graph with leaves. Recall that a *graph* is defined as $\mathscr{G} = (\mathbb{V}, \mathbb{E}, \mathscr{A})$, with $\mathbb{V}$ the set of *vertices*, $\mathbb{E}$ the set of *edges*, and $\mathscr{A}$ the *adjacency map*. $\mathscr{A}$ associates with each edge $e \in \mathbb{E}$ an unordered pair $\mathscr{A}(e) = [v_1, v_2]$ with $v_1, v_2 \in \mathbb{V}$, in which case e is said to be *adjacent* to v_1 and v_2. A graph with leaves is a graph in which some of the 'edges' are adjacent to only one vertex. These special 'edges' are called 'leaves'. Formally, a *graph with leaves* is defined as $\mathscr{G} = (\mathbb{V}, \mathbb{E}, \mathbb{L}, \mathscr{A})$, with $\mathbb{V}$ the set of *vertices*, $\mathbb{E}$ the set of *edges*, $\mathbb{L}$ the set of *leaves*, and $\mathscr{A}$ the *adjacency map*. $\mathscr{A}$ associates with each edge $e \in \mathbb{E}$ an unordered pair $\mathscr{A}(e) = [v_1, v_2]$ with $v_1, v_2 \in \mathbb{V}$, and with each leaf $\ell \in \mathbb{L}$ an element $\mathscr{A}(\ell) = v \in \mathbb{V}$, in which case e is said to be *adjacent* to v_1 and v_2, and ℓ to v.

4.4 The module embedding

The *module embedding*
(i) associates with each vertex of the interconnection architecture a module, and
(ii) specifies for every vertex a $1 \leftrightarrow 1$ assignment between the edges and leaves adjacent to the vertex and the terminals of the module that has been associated with this vertex.
This is illustrated in the figure below.

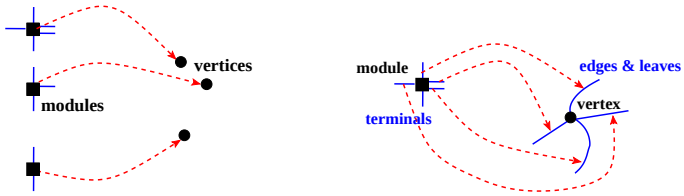


Fig. 6. Terminal assignment

Since each edge is adjacent to two vertices, each edge is associated by the module embedding with 2 terminals. It is assumed that this assignment results in terminals that are of the same type if the type is physical (both electrical, or mechanical, or hydraulic, or thermal, etc.), or of opposite type (one input, one output) if the terminals

are of logical type. In other words, if the edge e is adjacent to vertices v_1 and v_2, then the module embedding makes v_1 and v_2 either of the same physical type, or of opposite logical type. In this way, each edge and leaf is labelled by a terminal type, and each vertex is labelled as a module.

Consider again a few examples. The first is the electrical circuit shown in figure 7. The goal is to model the behavior of the voltage and current in the external port.

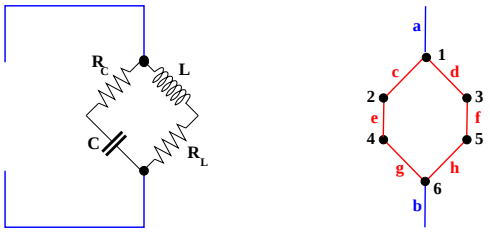


Fig. 7. RLC circuit

This circuit has 6 modules. Two resistors with parameter values R_C and R_L respectively, one capacitor with parameter value C, one inductor with parameter value L, and two connectors with parameter value 3 (meaning that it connects 3 terminals). All terminals are of electrical type, the resistors, capacitor, and inductor each have 2 terminals, and the connectors both have 3. The interconnection architecture is shown on the right side of figure 7. There are 6 vertices, labelled 1 to 6, 6 edges, labelled c,d,e,f,g,h, and 2 leaves labelled a,b. The module embedding consists of

$$R_C \mapsto 2, R_L \mapsto 5, C \mapsto 4, L \mapsto 3, \text{connector}_1 \mapsto 1, \text{connector}_2 \mapsto 6.$$

Because of the special symmetries that are valid for the electrical elements used, we need not specify how the terminals of the modules are exactly associated with the edges. If, for example, there would have been a diode in edge 2, we would have had to specify if its current blocking direction is associated with edge c or with edge e.

The second example is the classical feedback system shown in figure 8. The interconnection architecture is the graph with vertices A_1, A_2, G_1, and G_2, edges $3,4,5,6$, and leaves $1,2$, shown on the right side of the figure. The modules are two

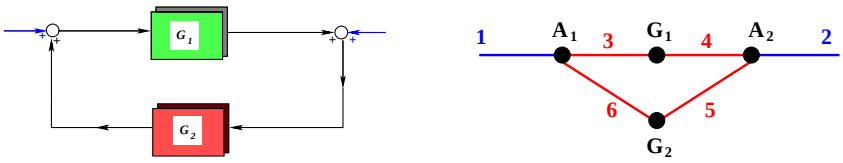


Fig. 8. Feedback system

adders, associated with vertices A_1 and A_2, each with 2 inputs and 1 output, and two input/output systems, associated with vertices G_1 and G_2. The module embedding requires that the appropriate input-to-output directions are respected.

4.5 Interconnection equations

The edges of the interconnection architecture specify how terminals of modules are linked. *Assume that there are universal rules that specify relations among the variables on the terminals that are linked.* Pairing of terminals by the edges of the interconnection architecture implies an *interconnection law*. Some examples of interconnection laws are shown in the table below.

Pair of terminals	**Variables terminal 1**	**Variables terminal 2**	**Interconnection constraints**
electrical	(V_1, I_1)	(V_2, I_2)	$V_1 = V_2, I_1 + I_2 = 0$
1-D mechanical	(F_1, q_1)	(F_2, q_2)	$F_1 + F_2 = 0, q_1 = q_2$
thermal	(Q_1, T_1)	(Q_2, T_2)	$Q_1 + Q_2 = 0, T_1 = T_2$
fluidic	$(p_1, f_1$	(p_2, f_2)	$p_1 = p_2, f_1 + f_2 = 0$
information processing	input u	output y	$u = y$
etc.	etc.	etc.	etc.

Table 4. Examples of interconnection laws

Proceeding this way leads to a complete set of behavioral equations:

(i) For each vertex of the interconnection architecture, we obtain a behavior relating the variables that 'live' on the terminals of the module that is associated with this vertex. These behavioral equations are called the *module equations*.

(ii) For each edge of the interconnection architecture, we obtain behavioral equations relating the variables that 'live' on the terminals and that are linked by this edge. These behavioral equations are called the *interconnection equations*. Note that no interconnection equations result from the leaves, but the associated terminal variables do enter in the module equations.

These equations together specify the behavior of all the variables on all the terminals involved. Each vertex of the interconnection graph is in the end labelled as a module, and each edge as a terminal type: we have systems in the vertices, and interconnections in the edges. This stands in contrast to conventional electrical circuit theory, which has the elements (i.e. modules) in the edges, and the interconnections in the vertices. The interconnection equations are usually very simple (see the table above). Typically they equate certain variables and put the sum of other variables to zero. We therefore think of interconnection as *variable sharing*.

For the examples discussed earlier, and with the obvious notation, we obtain the following specification of the behavior.
1. For the circuit, we obtain the module equations

$$\begin{aligned}
V_{c''} - V_{e'} &= R_C I_{c''},\ I_{c''} = I_{e'};\\
V_{f''} - V_{h'} &= R_L I_{f''},\ I_{f''} = I_{h'};\\
C\tfrac{d}{dt}(V_{c''} - V_{e'}) &= I_{c''}, I_{c''} = I_{e'};\\
L\tfrac{d}{dt} I_{d''} &= \left(V_{d''} - V_{f'}\right),\ I_{d''} = I_{f'};\\
V_a = V_{c'} &= V_{d'},\ I_a + I_{c'} + I_{d'} = 0;\\
V_b = V_{g''} &= V_{h''},\ \ I_b + I_{g''} + I_{h''} = 0,
\end{aligned}$$

and the interconnection equations:

$$\begin{aligned}
V_{c'} &= V_{c''},\ I_{c'} + I_{c''} = 0;\\
V_{d'} &= V_{d''},\ I_{d'} + I_{d''} = 0;\\
V_{e'} &= V_{e''},\ I_{e'} + I_{e''} = 0;\\
V_{f'} &= V_{f''},\ I_{f'} + I_{f''} = 0;\\
V_{g'} &= V_{g''},\ I_{g'} + I_{g''} = 0;\\
V_{h'} &= V_{h''},\ I_{h'} + I_{h''} = 0.
\end{aligned}$$

2. For the feedback system, we obtain the module equations

$$\begin{aligned}
(u_3, y_4) \in \mathscr{B}_{\mathrm{G}_1}\ ;\ (u_5, y_6) &\in \mathscr{B}_{\mathrm{G}_2};\\
y_2 = u_1 + u_6\ ;\ y_4 &= u_2 + u_4,
\end{aligned}$$

and the interconnection equations

$$u_3 = y_3;\ u_4 = y_4;\ u_5 = y_6;\ u_6 = y_6.$$

Here $\mathscr{B}_{\mathrm{G}_1}$ and $\mathscr{B}_{\mathrm{G}_2}$ denote the behavior of the input/output systems in respectively the forward loop and the feedback loop of the feedback system.

4.6 The manifest variable assignment

The final step consists of the *manifest variable assignment*. This is a mapping that assigns the manifest variables as a function of the terminal variables. *The terminal variables are henceforth considered as latent variables.*

For the circuit example the manifest variable assignment consists of the maps that defines the port voltage and port current as a function of the terminal voltages and currents:

$$V_{\text{external}} = V_a - V_b,\ \ I_{\text{external}} = I_a.$$

For the feedback system, the manifest variable assignment consists of

$$u_{\text{external}} = (u_1, u_2),\ \ y_{\text{external}} = (y_6, y_5).$$

The behavioral equations (the combination of the module equations and the interconnection equations), combined with the manifest variable assignment, define the full behavior of the system that is being modelled. It is the end result of the modeling process based on tearing ($\cong$ the interconnection architecture), zooming ($\cong$ obtaining the module equations and the manifest variable assignment), and linking ($\cong$ setting up the interconnection equations).

This tearing-zooming-linking modeling methodology has many virtues: it is *systematic* and *modular*, it is *adapted to computer assisted modeling* (with module equations in parametric form stored in a database, and with the interconnection equations also stored in a database), it is *hierarchical* (once a model of a system has been obtained, it can be used as a subsystem-module on a higher level). A good model library will have items that are *re-useable, extendable, modifiable, flexible*, etc.).

Disadvantages are that it immediately involves many variables. This can be alleviated by the (partial) elimination of latent variables when possible. For example, the interconnection equations often immediately lead to elimination of half of the interconnection variables. There are situations where the special structure of the modules and the interconnections allow a more direct elimination of some of the variables. For example, modeling of mechanical systems using Lagrangians, modeling of electrical circuits using ports instead of terminals, etc.

Our philosophy is to keep the interconnections highly standardized and simple, and to deal with complex models in the modules. For example, in the electrical circuit, a multi-terminal connector was viewed as a module, not as a connection. Also, in mechanical systems, joints, hooks and hinges should be viewed as modules, not as connections.

As a caveat, we should emphasize that not all interconnections or interactions of subsystems fit the framework described above. In particular, distributed interconnections were not considered, for example mechanical systems that are interconnected by sharing a surface, or heat conduction along a surface. Also, terminals do not capture interconnections along virtual terminals, as action at a distance. Finally, interactions as rolling, sliding, bouncing, mixing, etc., also require a different setting.

The resulting graph structure of an interconnected system has the modules in the nodes and the interconnections as the branches. This is faithful to the physics, and should be contrasted with the graph structure pursued in electrical circuit theory, which has the modules in the branches and the connectors in the nodes. This works fine with 2-terminal elements, but is awkward otherwise, and is difficult to generalize to other, non-electrical, domains.

5 Interconnected behavior

We now formalize the interconnected system. The most effective way to proceed is to specify it as a latent variable system, with as manifest variables the variables specified in the manifest variable assignment, and as latent variables all the terminal variables associated with all the modules. Its full behavior behavior then consists of the behavior as specified by each of the modules, combined by the interconnection

laws obtained by the interconnection architecture. A first principles model of an interconnected system obtained this way always contains many latent variables. This is one of the main motivations to introduce latent variables in our modeling language *ab initio*. It also underscores the importance of the *elimimation theorem* [1, 2, 4].

The modeling procedure described above has some similarity with modeling using *bondgraphs*. However, there are important differences, especially that in our setting it is not required that the interconnection variables have any relation to the enrgy that is transmitted along the interconnection terminals. The comparison with bondgraphs is taken up in detail in [5].

6 Conclusions

Modeling interconnected via the above method of *tearing, zooming, and linking* provides the prime example of the usefulness of behaviors and the inadequacy of input/output thinking. Even if our system, after interconnection, allows for a natural input/output representation, it is unlikely that this will be the case of the subsystem and of the interconnection architecture. It is only when considering the more detailed signal flow graph structure of a system that input/output thinking becomes useful. Signal flow graphs are useful building blocks for interpreting information processing systems, but physical systems need a more flexible framework.

Acknowledgments

This research is supported by the Belgian Federal Government under the DWTC program Interuniversity Attraction Poles, Phase V, 2002–2006, Dynamical Systems and Control: Computation, Identification and Modelling, by the KUL Concerted Research Action (GOA) MEFISTO–666, and by several grants en projects from IWT-Flanders and the Flemish Fund for Scientific Research.

References

1. T. Cotroneo, *Algorithms in Behavioral Systems Theory*, Doctoral dissertation, Faculty of Mathematics and Physical Sciences, University of Groningen, 2001.
2. T. Cotroneo and J.C. Willems, The simulation problem for high order differential equations, *Applied Mathematics and Computation*, volume 145, pages 821-851, 2003.
3. J.W. Polderman and J.C. Willems, *Introduction to Mathematical Systems Theory: A Behavioral Approach*. Springer-Verlag, 1998.
4. J.C. Willems, Paradigms and puzzles in the theory of dynamical systems, *IEEE Transactions on Automatic Control*, volume 36, pages 259–294, 1991.
5. J.C. Willems, Modeling open and interconnected systems, submitted.