Journal of Computational Science 10 (2015) 108–125

Contents lists available at ScienceDirect

Journal of Computational Science

journal homepage: www.elsevier.com/locate/jocs

Sustainability analysis of complex dynamic systems using embodied energy flows: The eco-bond graphs modeling and simulation framework

Rodrigo D. Castro[a,c,*], François E. Cellier[b], Andreas Fischlin[c]

[a] Computer Science Department, University of Buenos Aires and CONICET, Argentina
[b] Computer Science Department, ETH Zurich, Switzerland
[c] Department of Environmental Systems Sciences, ETH Zurich, Switzerland

ARTICLE INFO

Article history:
Received 25 February 2014
Received in revised form
17 September 2014
Accepted 14 October 2014
Available online 1 November 2014

Keywords:
Energy
Sustainability
Emergy
Bond graphs
Modelica

ABSTRACT

This article presents a general methodology for modeling complex dynamic systems focusing on sustainability properties that emerge from tracking energy flows.

We adopt the embodied energy (*emergy*) concept that traces all energy transformations required for running a process. Thus, energy at any process within a system is studied in terms of all the energy previously invested to support it (up to the primary sources) and therefore sustainability can be analyzed structurally.

These ideas were implemented in the bond graph framework, a modeling paradigm where physical variables are explicitly checked for adherence to energy conservation principles.

The results are a novel Ecological Bond Graphs (EcoBG) modeling paradigm and the new EcoBondLib library, a set of practical ready-to-use graphical models based on EcoBG principles and developed under the Modelica model encoding standard.

EcoBG represents general systems in a three-faceted fashion, describing dynamics at their mass, energy, and emergy facets. EcoBG offers a scalable graphical formalism for the description of emergy dynamic equations, resolving some mathematical difficulties inherited from the original formulation of the equations.

The core elements of EcoBG offer a soundly organized mathematical *skeleton* upon which new custom variables and indexes can be built to extend the modeling power. This can be done safely, without compromising the correctness of the core energy balance calculations. As an example we show how to implement a custom sustainability index at local submodels, for detecting unsustainable phases that are not automatically discovered when using the emergy technique alone.

The fact that we implemented EcoBondLib relying on the Modelica technology opens up powerful possibilities for studying sustainability of systems with interactions between natural and industrial processes. Modelica counts on a vast and reusable knowledge base of industrial-strength models and tools in engineering applications, developed by the Modelica community throughout decades.

1. Introduction

Modern societies rely on complex interactions with natural systems at many spatio-temporal scales. Such interactions often operate at rates exceeding the natural systems capacity to renew [30], leading to unsustainable structures.

As all human-driven processes depend ultimately on natural resources, their depletion, or overexploitation in case of naturally renewable ones, will necessarily shape the intensity, or even feasibility, of these processes in the future.

In order to study feasible future scenarios for human-driven processes, different approaches are required depending on the sustainability of the humans utilization of non-renewable and renewable services and goods, notably those from ecosystems.

For this kind of analysis it may be key to take into account the whole pathway of energy transformations that human-driven processes require (e.g. notably the combustion of fossil fuels).

[*] Corresponding author at: Computer Science Department, University of Buenos Aires and CONICET, Argentina.
E-mail address: rcastro@dc.uba.ar (R.D. Castro).

http://dx.doi.org/10.1016/j.jocs.2014.10.001
1877-7503/© 2014 Elsevier B.V. All rights reserved.

Means are needed to model systems and analyze the sustainability of such energy transformation paths.

A sustainable socio-natural system can be thought of as a "healthy ecosystem" [8]. Quantitative views of ecology [5] help defining, measuring, and interpreting ecosystems' health. In the seminal textbook [23] H.T. Odum (the book author's brother) proposed to quantify also the relations among components of an ecosystem in a systems theoretical manner to enable ecosystem management. He then extended this idea [24,27] to represent related elements of ecological systems in energy equivalents, e.g. as contained in biomass (the energetic content of biomass was used as a unifying measure for universal descriptions across differing ecosystem types).

It was recognized that ecosystems have structures and functions that operate across a broad range of spatial and temporal dimensions [2,29] and the overall integrity of a system, when adding human dimensions, may differ depending on the hierarchical scale at which the ecosystem is being utilized (e.g. an ecosystem supporting an industrial society may be "healthy"; however, an ecosystem receiving extensive waste from industrial processes may become "unhealthy").

A modeling approach known as Energy Systems Language (ESL) [23,26] was proposed to represent and analyze such systems across many spatial and temporal scales and hierarchies of organization (e.g. [1]).

Modeled processes should observe the laws of thermodynamics just like their physical counterparts [27]. H.T. Odum proposed that the emergence of hierarchical organization results from dissipation of the available energy [24]) and that feedback loops are created if energy is available in sufficient amounts [25]. The transfer of energy throughout a hierarchy served Odum as the basis for defining "embodied energy", or emergy.

ESL proposes a modeling approach that represents all conceivable resources in terms of a common accounting unit. As a simple illustration, consider a hypothetical supply chain for a biofuel, where 1000 J of sunlight are needed to produce 10 J of biomass, which in turn are used to produce 1 J worth of fuel [3]. Note all those energy amounts correspond to each other and are equated by introducing some common unit. Say 1 J of biofuel is equal to 1000 solar equivalent Joules (sej). Such an approach allows for adding various further resources in terms of their solar equivalents, and the assumption of substitutability is satisfied. This approach retains information on resource quality, thereby diminishing criticism about the loss of information due to energy path aggregation [17].

An energy quality indicator referred to as "Transformity" (Tr) converts all resources into solar-equivalent joules. It has been proposed that resources with higher transformity values are of higher quality and may be scarcer [27].

Emergy analysis is therefore of great importance as it features the unique capability of quantifying the contribution of diverse ecosystem goods and services under a common and meaningful measure, enabling a comprehensive, yet rigorous sustainability analysis.

Nevertheless, the emergy approach relies on detailed knowledge about complex socio-natural systems, which is likely to be inaccurate and incomplete. As a consequence, the method is considered controversial by some authors [18].

Emergy analysis is predominantly applied to systems at equilibrium, where averaged input/output flows into/from storages match, and even the dynamics for studying small departures from steady state are linearized for the sake of simplicity.

But emergy analysis also permits the modeling of flows of energy by means of highly non-linear dynamic functions, the system being at any type of operating point (stabe/unstable) or phase (steady state/transient). Therefore, very complex behavior can arise even with very simple equations [20].

It becomes then difficult to guarantee that the resulting model is consistent with physics, i.e. that the laws of thermodynamics are not violated. However up to the present, correctness in terms of the adherence of models to physical first principles relies to a large extent on the experience of the modeler, and little assistance is provided by current modeling and simulation technologies.

The usual practice is to perform iterative improvements or other refinements in such models by including the latest insights or by increasing resolution, often implemented over the course of several years. We claim that this approach, when not supported by adequate tools, adds particular risks for the thermodynamic feasibility of the upgraded model. It may well be that a model is not only improved by reducing inaccuracies or removing incompleteness, but also exacerbated by becoming thermodynamically inconsistent.

Therefore, there arises a need for a modeling methodology that supports all of the good features of ESL while guaranteeing thermodynamic feasibility.

1.1. Solutions proposed

Here we propose a new methodology that offers not only means to extend and enhance models incrementally, modularly, and hierarchically, but also provides techniques for tracking flows of matter and/or energy through the system in a systematic and rigorous way.

We present a formal system-theoretic Modeling and Simulation (M&S) framework, named Ecological Bond Graphs (EcoBG), along with a software tool that supports this novel methodology. This methodology is expected to be applicable in a flexible and efficient, yet rigorous and sound manner in M&S of complex natural and socio-economic systems, in particular when studying sustainability.

The framework consists of two pivotal cornerstones: an abstract graphical specification layer to work with system elements and structures (served by bond graph technology) at the top, and a specific equation encoding level (served by Modelica technology) at the bottom. Such an approach offers separation of the model specifications from implementation details while still aiding hierarchical modeling of target systems at all levels in an integrated manner.

2. Background

2.1. Bond graphs

Bond graphs (BG) [4,10] are a multi-physics modeling paradigm intimately concerned with the conservation of energy flows. The interdisciplinary concept of energy flow creates a semantic level that allows BGs to be independent of the modeling domain. Basic concepts of physics, such as the laws of thermodynamics, can be verified in a bond graph independently of their application domain.

Three different Modelica libraries have been created for dealing with different modeling goals embracing the bond graph approach: BondLib, MultiBondLib, and ThermoBondLib.

BondLib [13] makes use of the regular (black) bonds shown below.

$$\xrightharpoonup[f]{e}$$

Regular bonds carry two variables, the effort, e, and the flow, f. They do not carry units in order to make them usable for all application areas. If a bond gets connected, e.g. to an electrical system, the bond inherits units of Volts for the effort variable and units of Amperes for the flow variable and propagates those units across junctions throughout the model topology.

Following this idea, complex models involving interactions among different energy domains (e.g. electrical, mechanical translational, mechanical rotational, hydraulic) can be built under the same paradigm.

MultiBondLib [33] operates on (blue) multi-bonds, consisting of vectors of unit-less scalar bonds. Multi-bonds represent generalizations of regular bonds. This feature is usually needed in applications modeling 2D and 3D mechanical systems, but the concept is completely general. Here, the effort and the flow are vectors of length N that do not carry units by themselves, but inherit those later through connections to elements that belong to a particular energy domain, such as mechanics.

Whereas the dynamics of, e.g. electrical or mechanical phenomena can be fully captured by pairs of power variables, thermodynamic phenomena require three independent variables for their description. ThermoBondLib [11] operates on (red) thermo-bonds. Contrary to the regular and multi-bond, they carry units of their own. Red thermo-bonds carry three effort variables (temperature, pressure, and Gibbs potential), three scalar flow variables (entropy flow, volume flow, and mass flow), and also three state variables (entropy, volume, and mass). The latter are regarded as *information* variables in BG, to stress the fact that they are not part of *power* pairs (i.e. *flow-effort* pairs).

2.2. Bond graphs for sustainability analysis

In this work, we are interested in the ability of tracking flows of emergy, particularly in systems described by their mass flows. The emergy concept presents a fundamental departure from previous existing specializations of bond graphs. Therefore, we introduced the concept of **Ecological Bond Graphs (EcoBG)** that operate on (green) **"eco-bonds"**.

These were implemented in a fourth Modelica library, coined **EcoBondLib**. As shall be discussed in detail, eco-bonds transport a single pair of power variables (just as in regular black bonds) carrying *specific enthalpy* and *mass flow*, but in addition, they also transport state information (just like in the red thermo-bonds), namely the *mass*, and they also carry an information variable representing *specific emergy*. The latter will allow conducting sustainability analysis based on the embodied energy technique.

3. Energy consistency and sustainability analysis in complex dynamic models

Our methodology focuses on models exhibiting complex dynamics. Emphasis is on structurally complex models used to study sustainability. In the following, we shall refer to these models as Complex Dynamics Sustainability Models, or CDSM for short. CDSMs are often derived from observations of the evolution of measurable variables. In the context of sustainability, the energy is key, and we therefore focus on CDSMs that describe processes by means of flows of mass and their associated energy.

CDSMs in socio-natural sciences are often impossible to be deduced from first-principle physical laws (bottom-up approach) in spite of the fact that the latter are invariably dominating any real world process. Following top-down approaches, CDSMs are built from, tuned for, and validated against real world observations, thereby gaining validity. However, the internal structure of a model validated at the level of its observables may sometimes make questionable assumptions from an energetic point of view. Such a CDSM can be misleading in at least two ways. If the goal of the model is to understand the mechanisms behind a system under analysis, it may provide wrong explanations for the observed phenomena. If the goal is to forecast plausible future evolutions of the system, it may fail to offer credible predictions even for previously well-fitted variables.

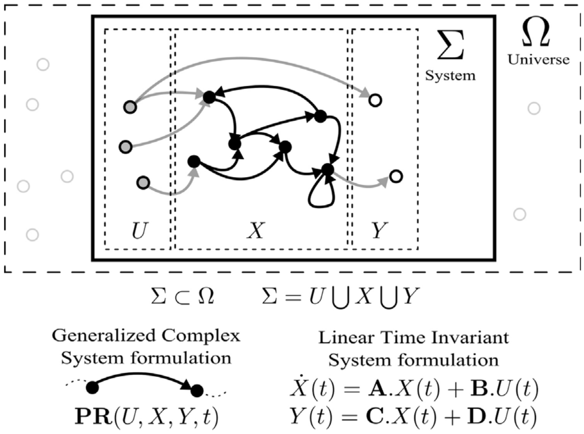

Σ
System
Ω
Universe
U
X
Y
$\Sigma \subset \Omega$
$\Sigma = U \bigcup X \bigcup Y$
Generalized Complex System formulation
$\mathbf{PR}(U, X, Y, t)$
Linear Time Invariant System formulation
$\dot{X}(t) = \mathbf{A}.X(t) + \mathbf{B}.U(t)$
$Y(t) = \mathbf{C}.X(t) + \mathbf{D}.U(t)$

Fig. 1. System-theoretic formulation of dynamic systems. **Nodes:** variables of type Sources (U), States (X), and Sinks (Y). **Edges:** processes of transformation. **Analytic formulations:** *Generalized Complex System* case, with arbitrary processes **PR**. *Linear Time Invariant System* case (state space form), with real-valued state matrix **A**, input matrix **B**, output matrix **C** and feed forward matrix **D**.

Improving the internal energy consistency of CDSMs cannot *guarantee*[1] that models will increase their explicatory capabilities, but it can certainly help to rule out models that are energetically unfeasible, thus enhancing their overall reliability.

Moreover, an energy-wise consistent CDSM can provide novel insights about the sustainability of the underlying mechanisms driving the real system. By making explicit the main paths of energy transformation attached to every observable variable – including all energy sources – can point out dependencies on energy renewability constraints that might otherwise remain hidden.

4. Energy and Emergy for CDSM: a system theoretic formulation

In Fig. 1 we abstract and generalize the modeling problem in terms of system theoretic principles, boiling it down to producing a mesh of *nodes* representing source nodes (input vector $U(t)$), accumulator nodes (state vector $X(t)$), and sink nodes (output vector $Y(t)$) that are interlinked by *edges* (generalized physical transformation processes *PR*).

The model is complete once we specify a *system boundary* to delimit our *system* Σ from the rest of the *universe* Ω.

The latter generalized formulation embraces a plethora of particular forms that impose different restrictions to the relations described by *PR*. A well-known example is the linear time invariant family of systems, where linear dynamics are expressed by defining four real-valued time invariant system matrices *A*, *B*, *C*, *D* (state-space form).

Nevertheless, for CDSMs linearity is not the most usual case, and therefore shall resort to the more general process $\dot{X}(t) = PR(U, X, Y, t)$ to describe the (possibly complex) interactions among any two nodes (similarly to what Fischlin introduced as the relational di-graph [15] to define system structure using a 2nd order predicate for such a generalized process).

We now want each node to express explicitly the relation between its mass and the associated energy. We shall denote the latter with *H* (Joule) referring to the enthalpy of the mass *M* (kg). Such a relation must be consistent with the flows of energy (i.e.

[1] Although, improving on energy consistency bears the potential of increasing the model's strengths in representing reality.

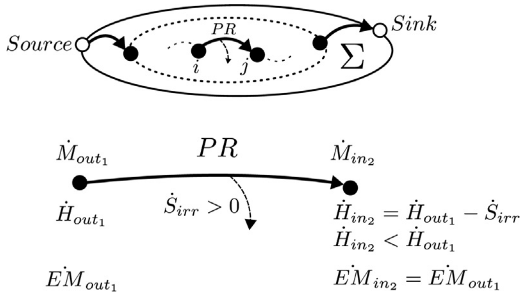

Source
PR
i
j
Sink
Σ
$\dot{M}_{out_1}$
PR
$\dot{M}_{in_2}$
$\dot{H}_{out_1}$
$\dot{S}_{irr} > 0$
$\dot{H}_{in_2} = \dot{H}_{out_1} - \dot{S}_{irr}$
$\dot{H}_{in_2} < \dot{H}_{out_1}$
$\dot{EM}_{out_1}$
$\dot{EM}_{in_2} = \dot{EM}_{out_1}$

Fig. 2. Process relation (PR) as the generalized edge. Production of irreversible entropy is explicitly considered with $\dot{S}_{irr} > 0$ (2nd principle of thermodynamics).

power H [W = J/s]) routed in and out through its connected Edges. See Fig. 2.

The problem is of a "local" nature in the sense that all variables required to formulate a consistent set of mass and energy equations are local to the given Node and its connected Edges. No information about other Nodes is required.

We also want to make explicit the fact that any real physical transformation process outputs an energy flow smaller than the sum of all its input flows, according to the 2nd principle of thermodynamics. Note the extra decoration added in Fig. 2 (bottom) to the *PR* thick arrow (outbound middle-placed thin-dotted arrow) depicting the positive non-zero flow of energy lost in the form of irreversible entropy.

Finally, we also want to track flows of emergy,[2] $\dot{EM}$ [W]. This problem, however, bears characteristics of a "structural" nature. The emergy flowing into a Node must equal the sum of emergies flowing out from its donor Nodes *regardless of the type of processes* connecting them. As stated before, every real process results in dissipation of energy in the form of irreversible entropy $\dot{S}_{irr}$ [W]. The amount of energy actually reaching a "consumer" Node is always smaller than the energy extracted from its "donor" Node. On the contrary, the emergy flow $\dot{EM}$ must account for all the original energy required to "drive" the process, i.e. *all the energy used up* ***before*** *considering losses*.

The bottom part of Fig. 2 represents a general pair of nodes i = 1, j = 2. The emergy flowing into Node 2 is the same emergy flowing out from Node 1, which in turn accounts for all the emergy supplied by Node 2's respective donors,[3] etc.

When this concept is propagated across all connected pairs of Nodes in a system, we realize that a change anywhere in the structure of the mesh can potentially influence the *emergy* of any given Node placed anywhere in the system, even when said Node's local *energy* balance remains intact.

Emergy captures a system property sensitive to its structure by memorizing energy supplies, tracing them backwards until the original sources at the systems boundary are reached.

4.1. A multi-faceted approach

In accordance with the system-theoretic framework developed above, we propose a compact view of Nodes and Edges decomposed into three facets: a "Mass Facet" where the local laws derived from mass observations are encoded, and two energy-oriented facets, namely the "Energy Facet" and the "Emergy Facet" (of local and structural nature, respectively), as depicted in Fig. 3.

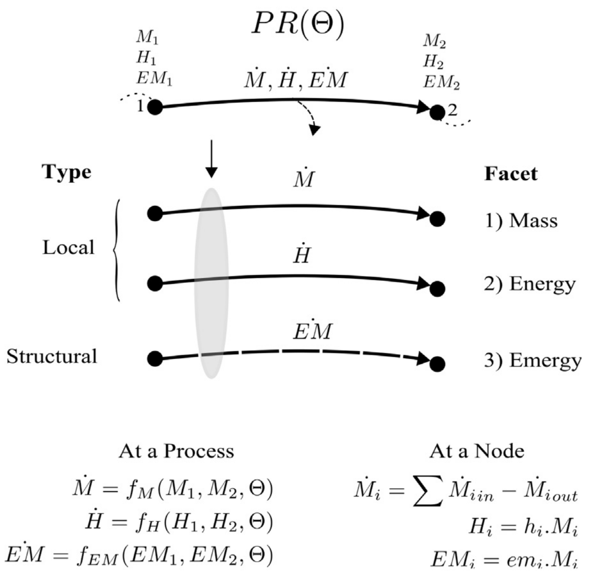

$PR(\Theta)$
M_1
H_1
EM_1
M_2
H_2
EM_2
$\dot{M}, \dot{H}, \dot{EM}$
1
2
Type
Facet
$\dot{M}$
Local
1) Mass
$\dot{H}$
2) Energy
$\dot{EM}$
Structural
3) Emergy
At a Process
At a Node
$\dot{M} = f_M(M_1, M_2, \Theta)$
$\dot{H} = f_H(H_1, H_2, \Theta)$
$\dot{EM} = f_{EM}(EM_1, EM_2, \Theta)$
$\dot{M}_i = \sum \dot{M}_{iin} - \dot{M}_{iout}$
$H_i = h_i.M_i$
$EM_i = em_i.M_i$

Fig. 3. Three-faceted view (mass, energy, and emergy) of the generalized process PR(Θ). Θ : Generalized parameters vector. f_M, f_H, f_{EM} : Functions defining the process transformations at the mass, enthalpy and emergy facets, respectively. h, em : Specific enthalpy and specific emergy (per unit of mass), respectively.

All Nodes and Edges are then equipped with mass-, energy- and emergy-awareness. This separation of concerns helps us to obtain a compact core set of equations for expressing CDSMs that is capable of scaling up for defining complex systems in a robust way. It also fosters easy assimilation of techniques other than emergy tracking to account for past energy transformations while keeping equations in the mass and energy facets encapsulated and self-consistent.

5. Bond graphs as the energy-based practical modeling paradigm

We shall now map the systemic approach of the previous section into a practical modeling formalism.

As illustrated in Section 2.1, BGs offer a highly suitable iconographic modeling framework for the purposes at hand, as they:

- distinguish natively between structure and behavior
- are based on the explicit tracking of energy flows, expressed by their basic pairs of effort and flow variables, and
- enforce explicitly energy conservation laws

As already conveyed in Section 2.2, we shall define a new core set of BG elements suitable for building CDSMs including emergy tracking capabilities. We shall refer to those as **Ecological Bond Graphs** or **EcoBG** for short.

It must be mentioned that H.T. Odum also offered an iconographic library of intercommunicable pre-programmed blocks containing mathematics and energetics adhering to the ESL rules [28]. This responded to a need of using graphical methods to diagram systems visually and get the simulation code generated automatically without writing equations. In [28] a visual ESL library was implemented for the EXTEND simulator[4] [19]. In so doing

[2] Flows of emergy are also referred to as "Empower" in the emergy literature.

[3] H.T. Odum's original formulation for $\dot{EM}$ resorts to a set of differential equations that are switched depending on the dynamics of the energy at a Node: increasing, decreasing or steady state [27] These were termed "differential-logic equations" for "Dynamic Emergy Accounting (DEA)". Strictly speaking, it consisted of a piecewise continuous type of system with discontinuities triggered by state events. As we shall see later, by sticking to our BG-based methods, we are driven to reject this switching idea, obtaining a more physically sound formulation for $\dot{EM}$. Tilley reached the same conclusions following other rationales [31]. The two approaches were developed in parallel and independently of each other [9].

[4] About the same time the Modelica language was created and the Dymola tool already implemented graphical user interfaces for assisting equation-based

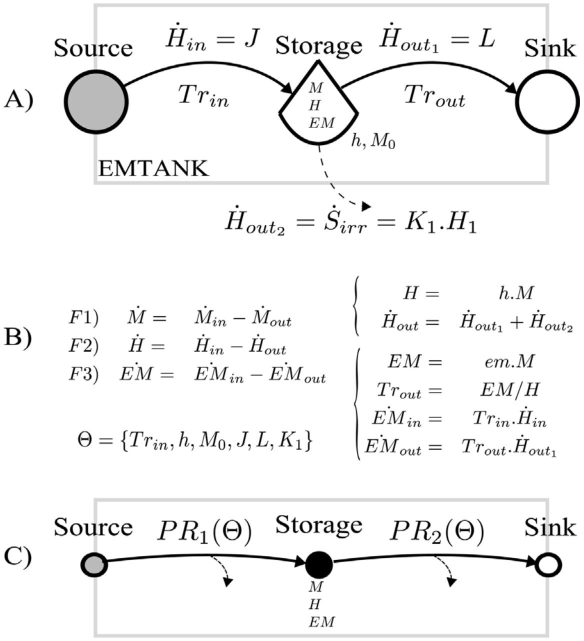

A)
Source
$\dot{H}_{in} = J$
Storage
$\dot{H}_{out_1} = L$
Sink
Tr_{in}
Tr_{out}
M H EM
h, M_0
EMTANK
$\dot{H}_{out_2} = \dot{S}_{irr} = K_1.H_1$
B)
$F1)\ \dot{M} = \dot{M}_{in} - \dot{M}_{out}$
$F2)\ \dot{H} = \dot{H}_{in} - \dot{H}_{out}$
$F3)\ \dot{EM} = \dot{EM}_{in} - \dot{EM}_{out}$
$\Theta = \{Tr_{in}, h, M_0, J, L, K_1\}$
$H = h.M$
$\dot{H}_{out} = \dot{H}_{out_1} + \dot{H}_{out_2}$
$EM = em.M$
$Tr_{out} = EM/H$
$\dot{EM}_{in} = Tr_{in}.\dot{H}_{in}$
$\dot{EM}_{out} = Tr_{out}.\dot{H}_{out_1}$
C)
Source
$PR_1(\Theta)$
Storage
$PR_2(\Theta)$
Sink
M H EM

Fig. 4. The EMTANK system: (A) original ESL formulation, (B) three-faceted view, and (C) system-theoretic formulation.

the authors proposed a standardized vector of values meant to carry Force, Flow and Transformity to communicate them throughout all blocks in a given system. The parallels with the EcoBG approach become evident, but this comes as no surprise. Odum early acknowledged in [24] the suitability of Bond Graphs for expressing his energy systems ideas, but nevertheless opted for developing ESL. One consequence was a departure from a standard framework that kept maturing throughout decades, driven by a growing scientific community that extended it to an ever increasing number of systems domains, but not yet to ecological systems as presented in this work. A partial conclusion we can draw from having taken the "Bond Graph path" is that it was very unlikely that a formulation similar to that of Odum's differential-logic equations would have ever been considered (i.e. the equation switching approach in ESL that led to physical inconsistencies at the heart of the Dynamic Emergy Accounting rules – see discussion by Tilley in [31]).

We shall now proceed with describing the main EcoBG elements, guided by motivating examples of increasing complexity.

5.1. Motivating Example 1: source–storage–sink system

This first example is inspired by a frequently referenced, paradigmatic model proposed by H.T. Odum [27] named EMTANK (Emergy Tank), a classical starting point for the emergy methodology. It represents the most basic (non-trivial) case possible from a structural point of view, as one single Node **Storage** has connections only to system boundaries, and from a dynamical point of view, because processes linked to a **Source** or a **Sink** are simpler to model than processes linking two storages.

Some considerations are in order. In Fig. 4A), the dynamics of the original model are expressed directly in terms of flows of energy ($\dot{H}$) rather than flows of mass ($\dot{M}$). In our approach, we can mimic the latter by setting the specific enthalpy to $h = 1$ J/kg, thus operating on generic masses. The equations of Facet 1 and Facet 2 would then look the same and deliver identical numerical values, albeit using different measurement units.

For the emergy equations (Facet 3), we can assume $\dot{EM}_{in} = \dot{H}_{in}$ by setting $Tr_{in} = 1$. A **Source** Node is a special case as it may import emergy flows from other subsystems, and Tr_{in} is the way of encoding it. Also, different Tr values can be used to scale flows of different types of energy in such a way that they are "converted" to flows of a single chosen type.

In Fig. 4C), we mapped the example to our graph-like system-theoretic formulation.

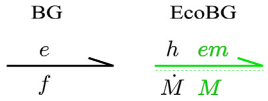
5.2. The EcoBG bond element

The basic BG element that transports energy between elements is the bond.

The most suitable pair of variables for CDSMs is that formed by the specific enthalpy h (an effort variable) and the mass flow $\dot{M}$ (a *flow* variable). Hence, $\dot{H}(J/s) = h(J/kg).\dot{M}(kg/s)$ represents power (W = J/s) (the flow of energy). While h and $\dot{M}$ provide enough information for depicting equations of Facets 1 and 2, we need additional information for Facet 3, i.e. the emergy facet.

We thus add to our bond the specific emergy *em* (J/kg) as an information variable, whereby $\dot{EM}(J/s) = em(J/kg).\dot{M}(kg/s)$ also denotes a flow of energy. The reason for choosing the information type rather than the apparently more suitable effort type for *em* will become evident later on. We shall represent EcoBGs with the classical harpoon with an extra dashed line indicating that information variables are being included, as depicted in the graph below.

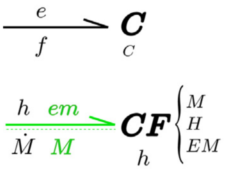
In the Modelica implementation of the EcoBG methodology, i.e. in the EcoBondLib Modelica library, we color-coded the new EcoBG bonds (eco-bonds) using the green color.



5.3. The EcoBG mass storage element

We made it already clear that our approach is particularly concerned with processes transporting and transforming mass, as well as storage elements for accumulating mass. The type of BG element suitable for mass storage is the C-element (or capacitor), as by definition, a C element integrates the flow variable and calculates the effort variable.

As we can see, the parameter C of the mass storage element must be $C = 1/h$. This is consistent with the idea that storages in a model represent one particular type of mass that can be uniquely characterized by its specific enthalpy. The EcoBG C element must also integrate emergy. To that end it resorts to *em*, the specific emergy information attached to the mass flow $\dot{M}$. In this case however, there does not exist a constant specific emergy characterizing the mass, as it is a consequence of chains of processes taking place somewhere else in the topology. Therefore $em = em(t)$ is kept inside the integral.



$$\left. e(t) = \frac{1}{C}\int f(t).dt \right\} \text{BG}$$

$$\left.\begin{aligned} H(t) &= h.\int \dot{M}(t).dt \\ M(t) &= H(t)/h \\ EM(t) &= \int em(t).\dot{M}(t).dt \end{aligned}\right\} \text{EcoBG}$$

modeling. As it will be seen later, these are two of the main practical tools used in this work.

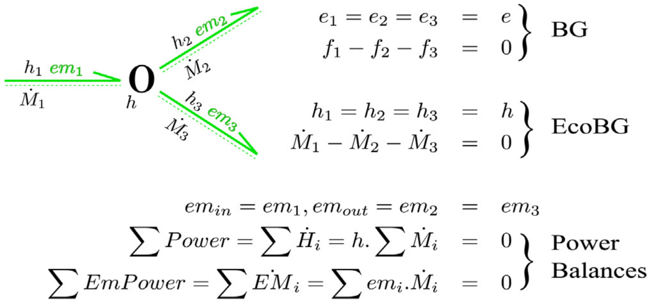

h_1 em_1
$\dot{M}_1$
O
h
h_2 em_2
$\dot{M}_2$
h_3 em_3
$\dot{M}_3$
$e_1 = e_2 = e_3 = e$
$f_1 - f_2 - f_3 = 0$
BG
$h_1 = h_2 = h_3 = h$
$\dot{M}_1 - \dot{M}_2 - \dot{M}_3 = 0$
EcoBG
$em_{in} = em_1, em_{out} = em_2 = em_3$
$\sum Power = \sum \dot{H}_i = h.\sum \dot{M}_i = 0$
$\sum EmPower = \sum \dot{EM}_i = \sum em_i.\dot{M}_i = 0$
Power Balances

Fig. 5. The EcoBG 0-Junction. Balances of mass, power, and EmPower.

5.4. The EcoBG junction element

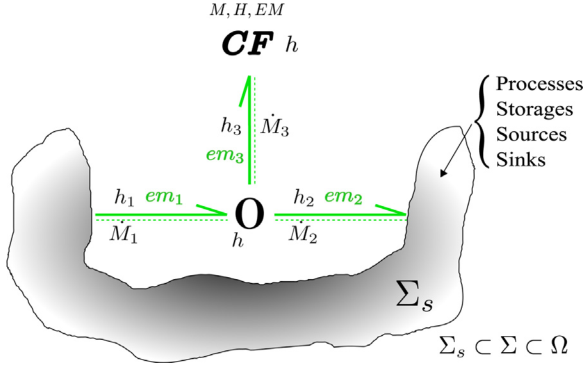
Junctions in BGs are structural elements enforcing energy conservation among those elements interconnected via the junction through their bonds.

- **0-Junctions** achieve that goal by enforcing **flow conservation** while operating at a **common effort**.
- **1-Junctions** enforce **effort conservation** among components operating under a **common flow**.

As systems described by mass flows are the primary objects to be captured by EcoBG models, the 0-Junction is the suitable element to be extended to cope with the concept of emergy.

The EcoBG 0-Junction element distributes a common specific enthalpy among its connected bonds, and enforces mass conservation through $\sum \dot{M} = 0$ (cf. Fig. 5). When a storage is connected to a 0-Junction, C becomes the element imposing its specific enthalpy h to the other connected bonds.

In Fig. 5, the emergy balance looks different from the power balance represented by $\sum \dot{M} = 0$, as there is no *em* common to all flows. According to the emergy principle, incoming flows contribute their own values of emergy per unit mass to build and sustain the storage, while all outgoing flows bear a new and common value of specific emergy imposed by the storage. This explains why we treat *em* as an information variable rather than as an effort variable. We can now take a look at a basic unit composed by the EcoBG elements defined so far, which will serve as an important building block for more complex models.



With the proposed variables, we are now able to describe the storage of mass while adhering to the multi-faceted approach proposed in Section 4.1.

5.5. EcoBG process elements

So far we described the basic tools needed to make Nodes connectable to the rest of the system while observing basic conservation principles. We still lack an element where to encapsulate the mathematical relations connecting Nodes. This will be our process element that populates the Edges webbing our topology. A process shall encapsulate two fundamental pieces of information.

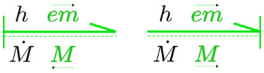
Firstly, at the mass facet, it describes the laws relating masses. The most general dynamic formulation for defining relations among masses looks as follows:

$$\begin{cases} \dot{M}_1 &= f_1(M_1, \ldots, M_n) \\ & \vdots \\ \dot{M}_n &= f_n(M_1, \ldots, M_n) \end{cases} \tag{1}$$

In our motivating example, we have only one type of mass, and assuming $h = 1$ for the sake of simplicity, the Process must express:

$$\begin{cases} \dot{M}_1 = J - K_1 \cdot M - L \\ H = h \cdot M, \;\; h = 1 \end{cases} \tag{2}$$

For a process element to be able to calculate $\dot{M}_1 = -K_1 \cdot M$, it needs to know the value of M, which so far is a variable internal to the C element. Therefore, we need to extend our bond element with one additional information variable carrying M, i.e. the state of the accumulated mass at the storage. Just like h, this variable can only be imposed by a C element.



Secondly, at the energy facet, processes shall be responsible for encoding the loss of energy due to production of irreversible entropy, an inevitable feature of all real processes. The consequence of the latter for the Emergy Facet is evident: the information of the energy used before the discounting of irreversible entropy shall be passed along unaltered for the emergy accounting purposes already discussed.

In Example 1, the two required processes are special ones – in fact, the simplest possible – due to source and sink being special types of Nodes. These Nodes' mass and energy balances are unidirectional as they only impose or accept power flows, respectively. They are placed at the systems border, and they do not accumulate mass or energy. We shall first define the source process and the sink process. At a later example, we shall deal with the more general type of process interconnecting storage nodes among themselves.

5.5.1. The EcoBG source process

We must derive a structure to represent "Process 1" in Fig. 4C).

Facet 1 (mass facet) The EcoBG source process (P_{src}) element provides the flow of mass injected into the storage (after losses).

An EcoBG effort source (S_e) element must feed P_{src} with the energy associated with the demanded mass flow in the form of specific enthalpy (before losses). Eq. (F1) expresses the laws for the mass flows considering that the mass brought in from the source ($\dot{M}_S$) is configured by the user as a parameter ($\dot{M}_{proc}$) of the process, and taking into account the mass lost due to the generation of entropy ($K \cdot M_c$). The mass flow is imposed by P_{src} to S_e ($M_S = M_{proc}$), as we adopted mass flow, rather than energy flow, as the main flow describing the dynamics of the system.

Facet 2 (energy facet) The flow of energy entering P_{src} must equal the flow of energy carried by the mass entering the Storage Node plus the flow of energy dissipated as entropy (S_{irr}).

The specific enthalpy is imposed at the right hand side of P_{src} by the storage element ($h_1 = h_c$). Under normal circumstances, the user-provided parameters h_S and h_C should match, as the type of substance being supplied by S_e and subsequently accumulated in C is of the same type. Yet, these parameter values can be chosen differently. The energy balance would then absorb the disparity with the additional capability of checking whether this generates physical inconsistencies (i.e. $S_{irr} > 0$ must always hold). An h_S greater

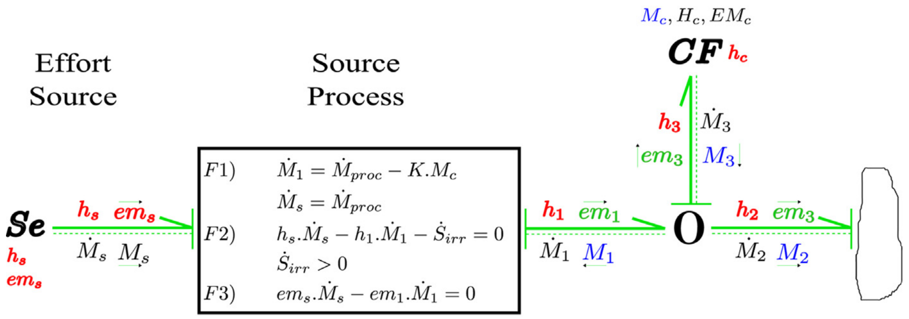


Fig. 6. The source process.

that h_C can be used to represent the energy required to procure the mass from the rest of the system (e.g. to extract a mineral from its ore).

Facet 3 (emergy facet) The specific emergy em_S supplied by the S_e element can be set as a parameter by the user.

Emergy flow reaches P_{src} to the amount of $EM_S = em_S . M_S$. This emergy flow is propagated unaltered to the right hand side of P_{src} as $\dot{EM}_1 = em_1.\dot{M}_1 = \dot{EM}_S$, ignoring the irreversible loss of energy via S_{irr} (in accordance with the principle of emergy accounting).

5.5.2. The EcoBG sink process element

Following a similar approach, we shall derive the structure to represent “Process 2” in Fig. 4C). In fact, we can already take advantage of the systematic formulations made so far and make the Sink Process be just a mirror version of the source process.

Conceptually, we should be able to simply invert the signs of flow equations in Facets 1, 2 and 3, thus transforming inputs to outputs, and vice-versa. In BGs, such an idea is achieved by inverting the bonds' harpoons that indicate the direction of positive flows (cf. Fig. 6).

In Fig. 7, we see the schematic representation of this idea. The sink process (P_{snk}) element defines through its phenomenological law, how mass is extracted from C. The S_e element (a source of effort now operating as a sink) can state the energy required for taking the mass out of the C element (h_S) together with the emergy associated with that energy (em_S). However, these parameters will not have any influence on the rest of the model: the energy contained per unit mass in C is defined by h_C, and its emergy per unit mass is calculated as $em_C = EM_C/M_C$ also at the storage element. Yet, specifying h_S and em_S can serve for the comprehensiveness of the model as a whole.

5.6. The EcoBG EMTANK model

We already defined all of the EcoBG elements required for representing the model of Fig. 4. The next figure shows the coupling of elements achieving that goal.

In the lower part of Fig. 8, we can see the equations that are automatically extracted from the graphical model by the Modelica compiler. We set the specific enthalpies and emergies to 1 for the sake of simplicity while analyzing the first results.

At Facet 2, we see that parameter K_{snk} of process P_{snk} plays no role, as we set $K_{snk} = 0$, implying that the process of mass/energy consumption from the Storage considers no losses.

On purpose, we decided to assign all losses due to entropy generation in the original model to the process P_{src}. In Odum's ESL language, dissipation losses can be modeled at storages and/or processes, with the former option being the one usually adopted. In ESL, dissipation losses represent emergy-less processes on their own. Nevertheless in the EcoBG approach, we consider that losses exist only as consequences of mass *transformations* (i.e. processes), and therefore the correct place to model them is at the EcoBG PROC elements.

At Facet 3, we recognize that the term $K_{src} \cdot M_C$, present at Facets 1 and 2, takes no effect; due to in EMTANK this represents the flux meant to describe irreversible losses, therefore carrying no emergy. For this reason, it must not be subtracted for emergy accounting.

5.6.1. Model implementation

We implemented the EcoBG elements introduced so far as individual models encoded in the Modelica language [21].

In particular, we used the Dymola visual modeling and simulation environment [14]. We shall delve more into the implications of this approach in Section 7. Fig. 9 shows EMTANK implemented in the Model View of Dymola. Besides implementing the EcoBG version of EMTANK, we encoded a subsystem capturing the original set of equations defining EMTANK for comparison purposes (cf. the blue box in Fig. 9). Parameter values were set for each element according to those presented in Fig. 8.

5.6.2. Model simulation

We simulated the model for tf = 350 units of time, i.e. until steady state is reached. Fig. 10 shows the results for the main variables of interest: *M*, *H*, *EM* (state variables) and *Tr*, simulated at the storage

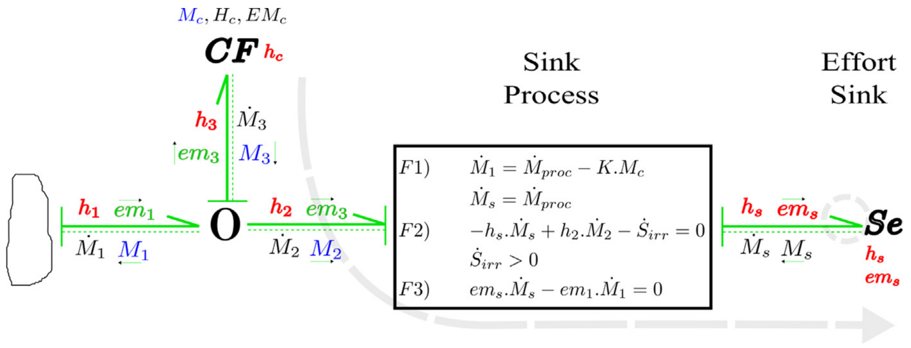
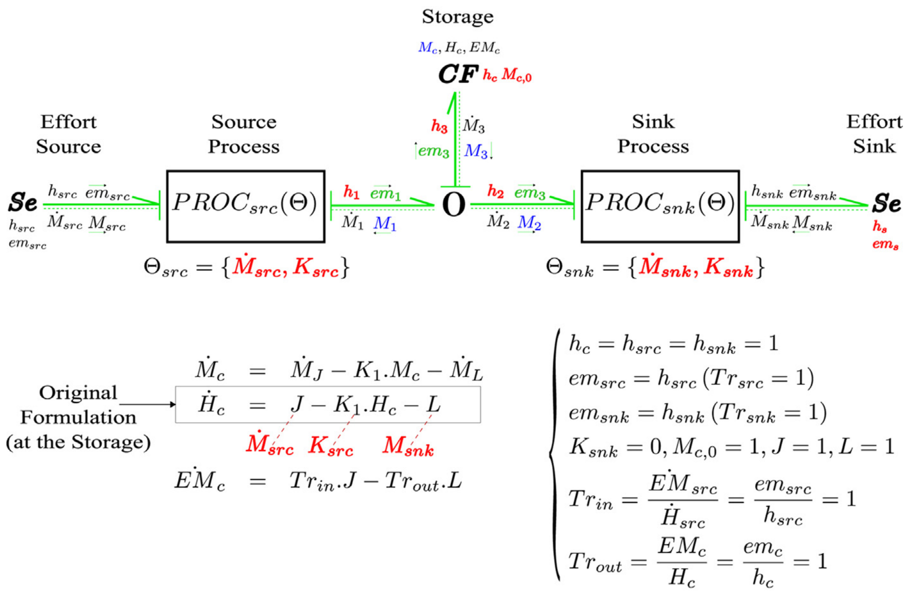


Fig. 7. The sink process.



$$\dot M_c = \dot M_J - K_1.M_c - \dot M_L$$

$$\dot H_c = J - K_1.H_c - L$$

$$\dot{EM}_c = Tr_{in}.J - Tr_{out}.L$$

$$\begin{cases} h_c = h_{src} = h_{snk} = 1 \\ em_{src} = h_{src}\,(Tr_{src} = 1) \\ em_{snk} = h_{snk}\,(Tr_{snk} = 1) \\ K_{snk} = 0, M_{c,0} = 1, J = 1, L = 1 \\ Tr_{in} = \dfrac{\dot{EM}_{src}}{\dot H_{src}} = \dfrac{em_{src}}{h_{src}} = 1 \\ Tr_{out} = \dfrac{EM_c}{H_c} = \dfrac{em_c}{h_c} = 1 \end{cases}$$

Fig. 8. The source–storage–sink system (EMTANK).

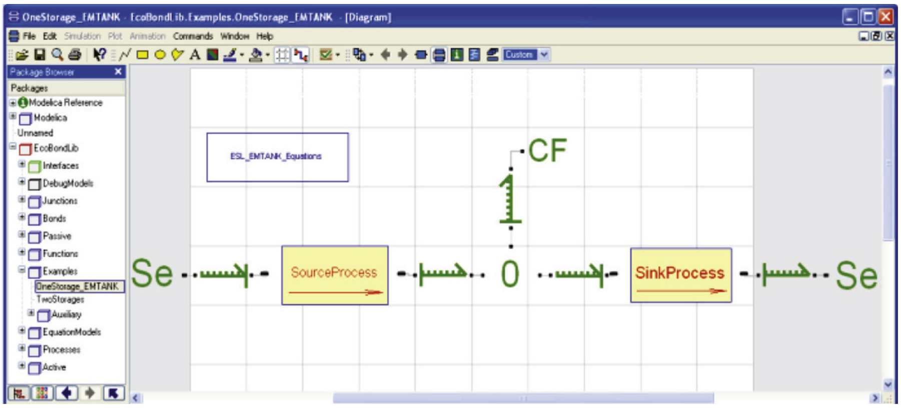


Fig. 9. The EcoBG implementation of the source–storage–sink system in Dymola.

element in the EcoBG model and also at the EMTANK block that implements the standard equations proposed by Odum (legends squared with dotted lines).

We verify that the EcoBG model mimics exactly the original EMTANK behavior (the trajectories are indistinguishable for *M*, *EM*, and *Tr*). We also display *H* together with *EM*. Since all specific enthalpy values were set to 1, the curve of *H* matches that of *M*. Yet, *H* differs from *EM* as the latter remembers all of the energy that was used at the source before discounting losses. The ratio *Tr* = *EM*/*H* is verified to be consistent throughout the simulation.

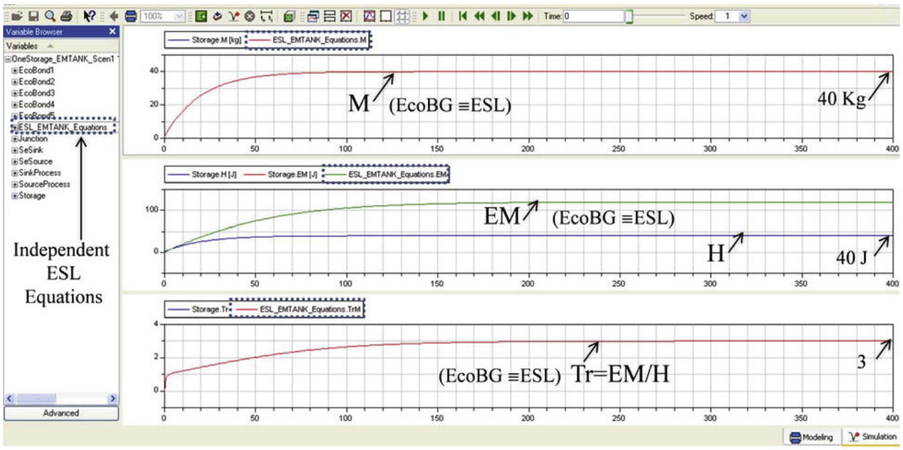


Fig. 10. Baseline simulation of EMTANK with EcoBG and standard ESL equations. EcoBG mimics ESL results.

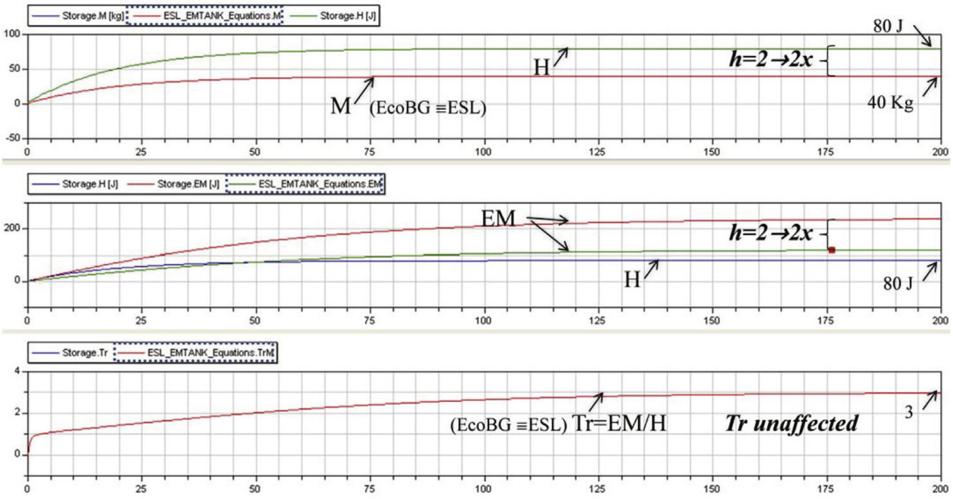


Fig. 11. Scenario 1: baseline with doubled specific enthalpy.

With the given parameter values, the “quality” of each unit of mass available for consumption from the Storage at steady state is 3. In less trivial networks of processes, this number helps to identify which Storages bear higher energetic quality, indicating that their overall energetic cost is higher.

We can now experiment with different parameter values. E.g. if we change the specific enthalpy of the mass, we should see no changes in the mass dynamics, but we should notice changes in the energy dynamics. To do so, we set $h_C = h_S = 2$, and also $em_S = h_S$ so as not to incur changes in the transformity. Results are shown in Fig. 11.

All results are as expected. If we now change only the specific emergy at the source, making it, e.g. $em_S = 2 \cdot h_S = 4$, we should see the dynamics of mass and energy unchanged, whereas now the emergy flows change by a factor of 2. This is shown and verified in Fig. 12.

Before coding a less trivial example, an important remark is in order. As discussed before, we use source elements at the systems’ borders. So far we used *effort* sources, which state how much energy and emergy the requested mass flow shall carry. Yet we know that physical processes are always limited in terms of the maximum power they can deliver. Apparently our ideal sources are able to withstand flows of infinite energy, which is certainly non-physical. To remedy this situation, we equip sources with a $P_{\max}$ parameter. Should the condition $P_{src}(t) = \dot{H}(t) > P_{\max}$ become true, the simulation should stop indicating that the system is requesting more power from the source than can be delivered.

$P_{\max}$ represents a coarse-grained abstraction, an assumption made for representing limitations that arise from more complex dynamics at parts of the system that have been lumped together in a single source element. In the next example we shall see richer dynamics by replacing the source of effort at the supply side with a submodel that accounts for flow, power, and storage limitations, thus representing limitations more realistically as they are found in nature.

6. Example 2: two storages. Replacing idealized sources. Checking for sustainability

In Fig. 13 we show an example that considers a natural process that produces a primary renewable reservoir of matter (C_a), which serves as the actual repository feeding the supply process.

Now, we use a source of flow (S_f) element. Such a source imposes $\dot{M}_r$ while accepting the specific enthalpy imposed by the system it is connected to (in our example h_a, a parameter of C_a). For S_f, the contributed mass flow is not a consequence of a

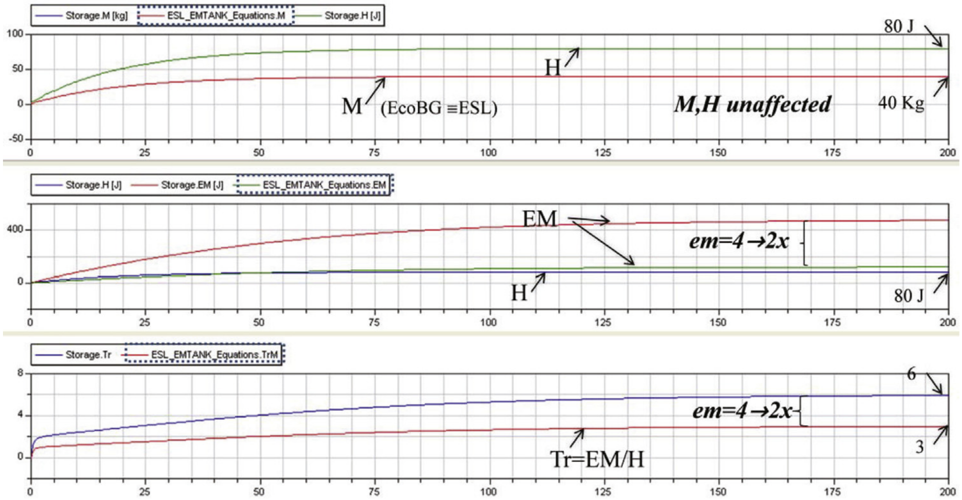


Fig. 12. Scenario 2: scenario 1 with doubled specific emergy *em*.

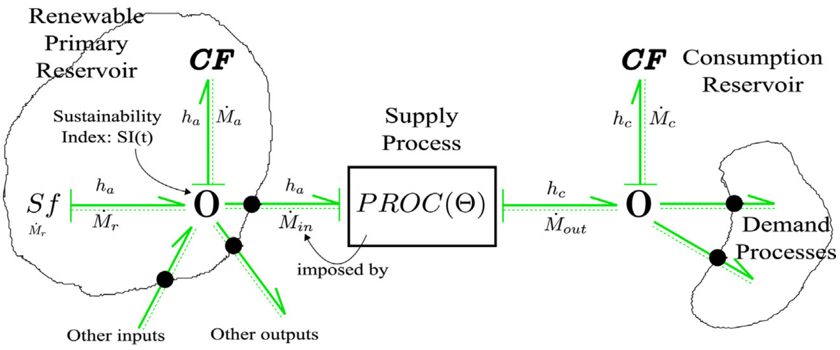


Fig. 13. System with two storages: source deposit (left) and a consumption reservoir (right).

process demanding material, but rather a known supply parameter. It represents a natural process, whose mass flow can be directly measured (such as the flow of rain filling an underground aquifer). In this new model, the power limitation will be given by the balance between incoming and outgoing flows at the junction element. Consuming processes that consistently draw more power than that supplied by the donor processes are not sustainable, and checks must be performed for those cases in the same way that we used to check against $P_S > P_{\max}$ in our previous example. We now talk about checking for sustainability[5] rather than for instantaneous maximum power, because processes can potentially – though only temporarily – draw more power than is being instantaneously provided, by means of consuming from the storage the required deficit energy (thereby lowering the storage level).

Parameterization for the system in Figs. 13 and 14 is as follows:

$$\begin{array}{ll}
\text{Rain}(Sf_r) & : \dot{M}_r = 2, h_r = em_r = 1 \\
\text{Aquiferdeposit}(C_a) & : h_a = 1, M_{a,0} = 150 \\
\text{Supply}(PR_S) & : K_s = 0.001, K_S^{irr} = 0.1 \\
\text{Consumption } \textit{deposit}(C_c) & : h_c = 1, M_{c,0} = 1 \\
\text{Demand}(Se_d) & : K_d = 0.05, K_d^{irr} = 0
\end{array} \tag{3}$$

The equivalent set of equations expressed with the ESL method (and implemented within the standalone model named `ESL_EMTANK_Equations`, upper right corner) is as follows (see the notation correspondence with in Fig. 4).

Taking subindexes 1 for the aquifer storage, and 2 for the consumption storage, and retaining subindexes r for rain and d for demand:

$$\begin{cases} \dot{M}_1 = \dot{M}_{1in} & -K_1 \cdot M_1 \cdot M_2 \\ \dot{M}_2 = +K_1 \cdot M_1 \cdot M_2 & -K_2 \cdot M_2 - \dot{M}_{2out} \end{cases} \tag{4}$$

with $\dot{M}_{1in} = \dot{M}_r$ and $\dot{M}_{2out} = \dot{M}_d$.

$$\begin{cases} \dot{H}_1 = J & -h_1 \cdot K_1 \cdot M_1 \cdot M_2 \\ \dot{H}_2 = & +h_2 \cdot K_1 \cdot M_1 \cdot M_2 - h_2 \cdot K_2 \cdot M_2 - L \end{cases} \tag{5}$$

with $J = h_r.\dot{M}_r$ and $L = h_2.\dot{M}_d$.

$$\begin{cases} E\dot{M}_1 = Tr_j \cdot J & -Tr_{M_1} \cdot h_1 \cdot K_1 \cdot M_1 \cdot M_2 \\ E\dot{M}_2 = & +Tr_{M_1} \cdot h_2 \cdot K_1 \cdot M_1 \cdot M_2 - Tr_{M_2} \cdot L \end{cases} \tag{6}$$

with $J = h_r \cdot \dot{M}_r$ and $L = h_2 \cdot \dot{M}_d$. Note that the term $-h_2 \cdot K_2 \cdot M_2$ of $\dot{H}_2$ in Eq. (5) represents the loss due to irreversible entropy generation.

The parameterization for Eqs. (4)–(6) is as follows: $J = 2$, $K_1 = 0.001$, $K_2 = 0.1$, $L = 0.05$, $Tr_J = Tr_{M_2} = 1$ with initial conditions $M_{1,0} = 150$, $M_{2,0} = 1$, $EM_{1,0} = EM_{2,0} = 0$.

6.1. Simulations

Fig. 15 below shows simulation results for a system configuration yielding oscillatory behavior (a mode possible only with two or more storages).

All curves of the EcoBG model overlap those produced by the simulation of the independent ESL Eqs. (4)–(6).

For both storages, the upper graph shows M, the center graph depicts H and EM, and the lower graph displays Tr. All variables reach steady state in the long run (the oscillations die out). We can see how the non-intuitive evolution of H and EM at the second storage produces an oscillatory trajectory for its transformity, indicating that the “quality” measure as defined in the emergy methodology is indeed a dynamic concept. Emergy analysis uses flows of emergy and their transformities along with labels assigned to flows (e.g. “renewable” or “non-renewable”), and then produces several sustainability-related performance indicators. EcoBondLib provides us with all the elements required to model and simulate the dynamics of arbitrarily interlinked storage/mass flow systems. Special indicators can be programmed easily to warn us when the simulation enters an operational mode that is not physically feasible.

6.2. Augmenting the expressive power of EcoBG with custom variables: the case for a sustainability index

The definitions used so far for the basic constitutive elements of EcoBG represent only a well organized *skeleton* upon which new variables and indicators can be added to convenience and safely, without compromising the correctness of the core energy balance calculations.

As an example, we present the case for a custom sustainability index (*SI*) indicator for EcoBG, along the lines of similar indexes well established for a long time in natural resources management, as e.g. the maximum sustainable yield in the areas of fisheries and forestry.

As mentioned at the beginning of the section, a sustainable balance of power must be observed locally at every source or storage, otherwise the whole system is unsustainable.

[5] Consequently, it must be remarked that we give the qualifiers *sustainable* and *unsustainable* a time-scale dependent interpretation. Specifically, the sustainable or unsustainable condition at a given point in time implies the assumption that the energy flow balance remains under the current conditions. Because the power balance can change its sign at given instants (switching from sustainable to unsustainable or vice versa) we shall preferably talk about sustainable or unsustainable *phases*.

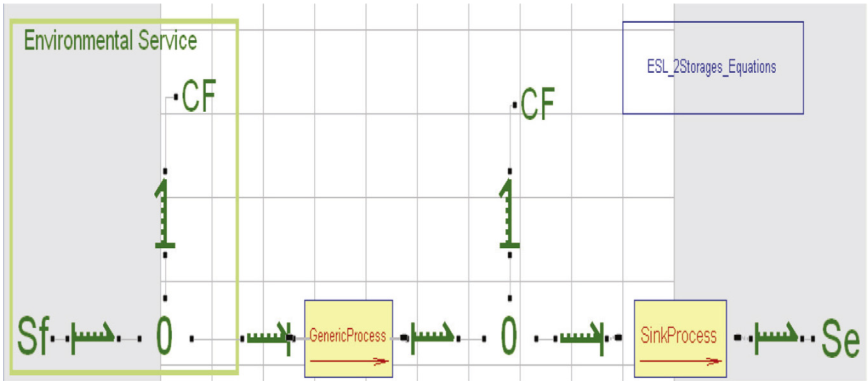


Fig. 14. The EcoBG implementation of the two storages system in Dymola.

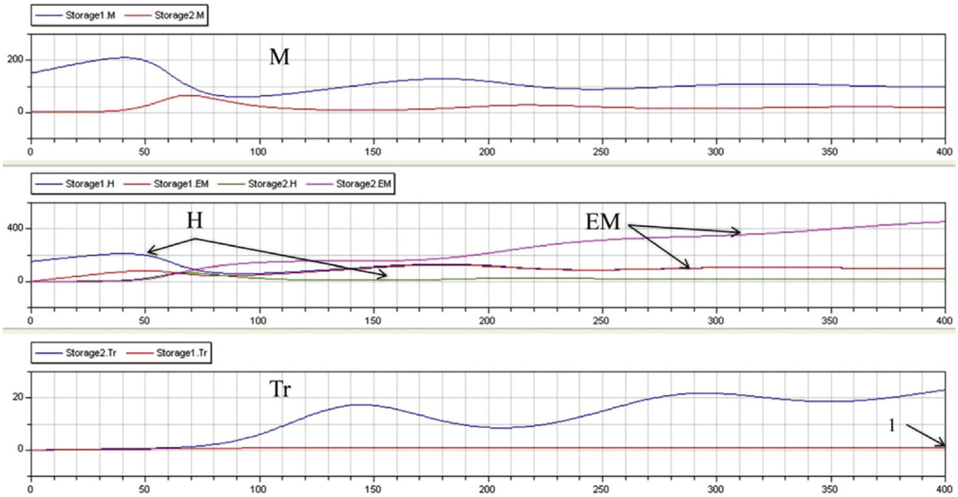


Fig. 15. Baseline simulation of the two storages model with EcoBG.

For EcoBG junctions we define

$$SI(t) = \frac{Power_{in}(t)}{Power_{out}(t)} - 1 \tag{7}$$

$SI(t) < 0$ indicates an unsustainable phase, whereas $SI(t) \gg 0$ denotes a sustainable phase, $SI(t)$ being the consequence of the (possibly many) simultaneous flows of energy entering and leaving the Junction. The power balance mandates that:

$$Power_{in}(t) - Power_{storage}(t) - Power_{out}(t) = 0 \tag{8}$$

For $SI(t)$, we ignore $Power_{Storage}(t)$, i.e. how much power is being accumulated by or consumed from the storage, because the condition $Power_{out}(t) > Power_{in}(t)$ is unsustainable regardless of the behavior of the storage. At a source, we set $Power_{in}(t) = P_{\max}$ (a parameter of the element). Therefore, $SI(t)$ will depend solely on the $Power_{out}(t)$ imposed from the connected processes. This sustainability concept, local to nodes in a system, is not present in ESL and is not a matter to be captured at the emergy facet.[6]

In Fig. 16, we show the results of a scenario where we reduced by a factor of four the constant inflow of mass supplied by nature (rain) into the renewable storage (aquifer, C_a). The figure presents the impact of this change on the second storage (consumption, C_c).

Contrary to the baseline scenario where we decided to finish the simulation at tf = 400, the simulation stops at tf = 210 in this second scenario, because C_c gets emptied and cannot supply anymore the demanded flow of energy. In the bottom panel of Fig. 16, we plotted $SI(t)$. We can see that for times $tf > 107$ s, $SI(t)$ assumes negative values indicating the beginning of an unsustainable phase.

The simulation never recovers from this unsustainable operational mode leading to the total depletion of C_c before its donor process can replenish it drawing from C_a. Of course, it can happen that a system experiences temporary phases of unsustainable operation from which it subsequently recovers.

Such temporary unsustainable regions should be a reason for concern, but may be acceptable if no better alternative can be found.

$SI(t) < 0$ is an indication of an instantaneous mode of operation that cannot be sustained forever. This information is not intuitive and cannot be immediately derived from observing independently any of the main variables analyzed so far in any of the three facets considered.

A potential advantage of this indicator in the context of the EcoBG approach is that it could be propagated from a given Junction to all its connected processes, opening up the possibility of designing rules of decisions to be made by processes reacting to this augmented knowledge. For example, the process drawing from storage C_a could shut down for a while when the storage level of C_a becomes too low, thereby granting the source enough time to replenish the storage.

[6] An Emergy Sustainability Index (ESI) exists that depends in turn on other emergy-based indices relying on qualitative tagging [7]. While emergy-based indices are a useful tool, they are of a relative nature: they depend on arbitrarily made choices for e.g. the tagging procedure or for the system boundary definition. In contrast, our SI indicator is based on energy, and is locally unambiguous at each junction where it is evaluated, regardless of assumptions made elsewhere.

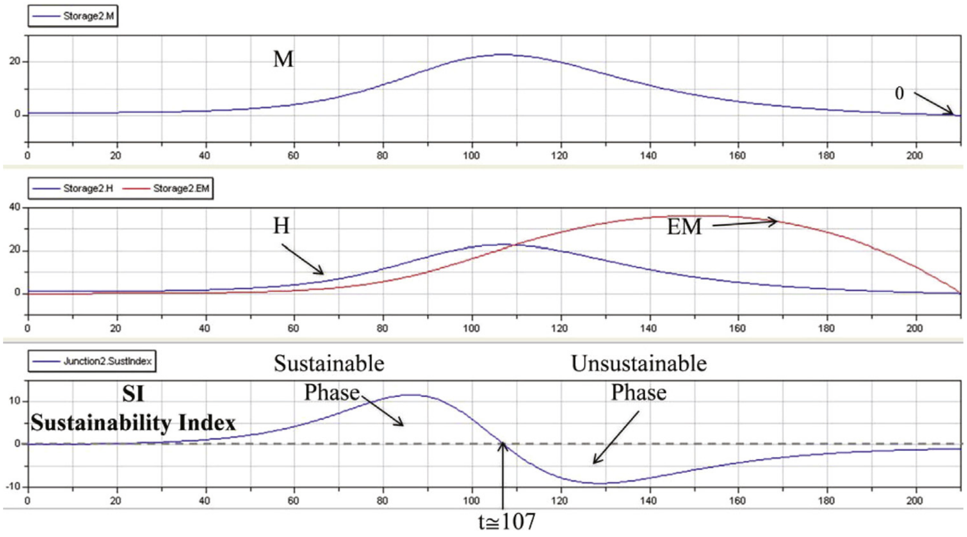


Fig. 16. Scenario 3: baseline with natural inflow reduced by four. Curves are for the second storage.

7. Modelica as the model encoding language

We took advantage of the BG modeling framework as the foundation onto which to build and extend ideas of energy- and emergy-oriented modeling of CDSMs. We implemented the core set of EcoBG models in Dymola, a Modelica-based modeling and simulation environment. There are further relevant implications of having chosen this path.

7.1. Modeling aspects

Rather than being a traditional sequential programming language, Modelica[7] is a mathematical modeling language that expresses mathematical relations using an object-oriented declarative textual (eventually also graphically rich) interface. A Modelica compiler checks not only for syntax errors or code completeness, but most importantly also for soundness and solvability of the set of mathematical equations encoded in the model.

Modelica features also model inheritance capabilities that make model development efficient and robust. For example, the Process models used in our examples differ in their laws governing mass and energy, but they all follow the same rule at the emergy facet (namely, they all pass along the emergy flow from input to output). This facet can then be coded once in a base "template" process model, so that every new specialized process will inherit this functionality. Changes in the template process will then be automatically inherited by all models that are based on that template, while retaining their specializations.

Benefitting from the latter, full subsystems can also be declared as new self-contained classes hiding away inner intricacies, exposing only selected input and output interfaces, and assigning them with convenient graphical decorators. This feature would make it straightforward to build a library of emergy models following Odums ESL iconography while actually building them using EcoBG as the underlying plumbing technology.

Also very important for productivity, teamwork, and scientific communication, Modelica models are self documentable.

7.2. Simulation aspects

Modelica models can be run by any tool adhering to its language specification, which is open and standardized. Advanced Modelica implementations, such as Dymola, offer robust integration algorithms for simulating models that are potentially difficult to deal with.

For instance, certain models call for adaptive-step size algorithms to offer simulation efficiency while retaining the accuracy and stability requirements. Other models combine variables with very low and very strong dynamics (known as stiff models), requiring a particular family of techniques (implicit methods) in order to preserve efficiency and numerical stability. Yet another class of models can present very frequent and heavy discontinuities, challenging the integration methods regarding their efficiency and correctness at both detecting and handling said discontinuities (For a comprehensive treatment of different types of continuous models and their suitable numerical integration methods see [12]).

Most of the classical ESL implementations are rather weak in this respect, offering e.g. only a fixed-step explicit Forward Euler algorithm as their numerical integration method for simulation.

The dangers of relying on only one (or a reduced set of entry-level) numerical solvers can be high. Dynamic models of even low complexity can already present serious numerical difficulties.

Hence, drawing conclusions about sustainability issues in CDSMs by trusting outcomes of simulations tested with only one primitive numerical method is not recommended. In this same context, having a mathematical encoding language that allows a piece of code to be interpreted by different tools with different strengths is a considerable advantage, as models can gain more credibility and acceptance in a potentially wider community of users. The clean separation between the modeling aspects and the simulation aspects embraced by Modelica offers an important advantage of our approach in the analysis of complex systems.

7.3. Modelica implementations of selected EcoBG components and systems

In this section we will present an overview of some relevant implementation details for a selected set of EcoBG models.

The goal is to provide the reader with a grasp of the relation between the theoretical concepts discussed so far and their practical implementation with the Modelica language.

[7] For an quick entry level reference to Modelica properties and features we suggest the following reading aimed at a non-programmer audience: https://www.modelica.org/education/educational-material/lecture-material/english/ModelicaOverview.pdf

As it shall be seen, similar concepts are present both in the Modelica encoding language and in the bond graph modeling methodology. Even though they are independent technologies, concepts match smoothly provided both methodologies look at physical systems through the glass of modularization and energy conservation principles.

7.3.1. EcoBG connectors

Connectors are the means for making different models compatible with each other at their "interface" (or "port") level. Accordingly, they are placed at the `EcoBondLib.Interfaces` sub-library. They represent the most basic EcoBG wrapping level. While EcoBG models can have internal dynamics of any complexity, their communication with other EcoBG components shall be reduced down to the information conveyable by their constitutive connectors.

The flow connector (Modelica `connector fEcoBondCon` class) and the effort connector (Modelica `connector eEcoBondCon` class) are uni-directional types of connectors. The `EcoBondCon` is a direction-less type of connector within which variables shall adopt the direction being enforced by a directional connector linked to it.

The code in the Listing 1 below corresponds to the effort connector `eEcoBondCon`.

Listing 1. The eEcoBondCon connector model.

```
connector eEcoBondCon "Uni-directional eco-bond graph with
   state information connector"
 ////// Mass Facet //////
 // Power
 flow output Modelica.SIunits.MassFlowRate Mdot "Mass flow";
 // Information
 input Modelica.SIunits.Mass M "Mass";
 ////// Energy Facet //////
 // Power
 input Modelica.SIunits.SpecificEnthalpy h "Specific
    enthalpy";
 // Information
 output Modelica.SIunits.SpecificEnthalpy h_partial "Maximum
     specific enthalpy to render the partial
 mass-to-mass phenomena thermodynamically feasible";
 ////// Emergy Facet //////
 // Information
 input Modelica.SIunits.SpecificEnergy em
   "Specific total emergy (embodied energy)";
 output Modelica.SIunits.SpecificEnergy em_partial
   "Specific partial mass-to-mass emergy (embodied energy)";
 ////// Auxiliary //////
 output Integer d "Directional variable";
 output Integer c "Causality variable";
end eEcoBondCon;
```

We use a special class of variable in Modelica language: the `connector` class. It is used in `connect` statements to link models that are wrapped by compatible types of connectors, thus building more complex models by the interconnection of simpler ones.

No equations can be defined within connectors, as they are meant only to expose variables, not to change them.

Inside `eEcoBondCon` we split the declaration of variables according to the three-faceted paradigm presented in Section 4.1: The mass, energy and Emergy facets (this approach will in fact be followed to organize internally the programming of all EcoBG models). Within each facet, two types of variables are defined: power and information variables. This distinction is entirely up to the user. Information variables can take values without supervision by the Modelica compiler. Nevertheless, some Power variables are checked for consistency by the compiler. This is the case of variables declared as flows using the Modelica `flow` prefix (in this case `Mdot`, the flow of mass), which are checked at connections to sum up to zero in order to guarantee conservation principles (conservation of mass in this case).

Auxiliary variables are intended to track the positive direction of flow (`d`, incoming or outgoing from the connector) and the causality (`c`, imposed to or by the connector). Even when these could be regarded also as Information variables, they do not bear a standalone meaning as they complement the flow variable. We thus decided to keep them apart from the stronger organizational categories of facets (Power and Information), and leave them as auxiliary.

The *power pair* of variables, mandatory in any bond graph model, is present at connectors through the effort (across) variable `h` and the flow (through) variable `Mdot` (Power is `h*Mdot`).

Finally the `input` and `output` variable constraints are used to enforce causality consistency when linking connectors with each other.

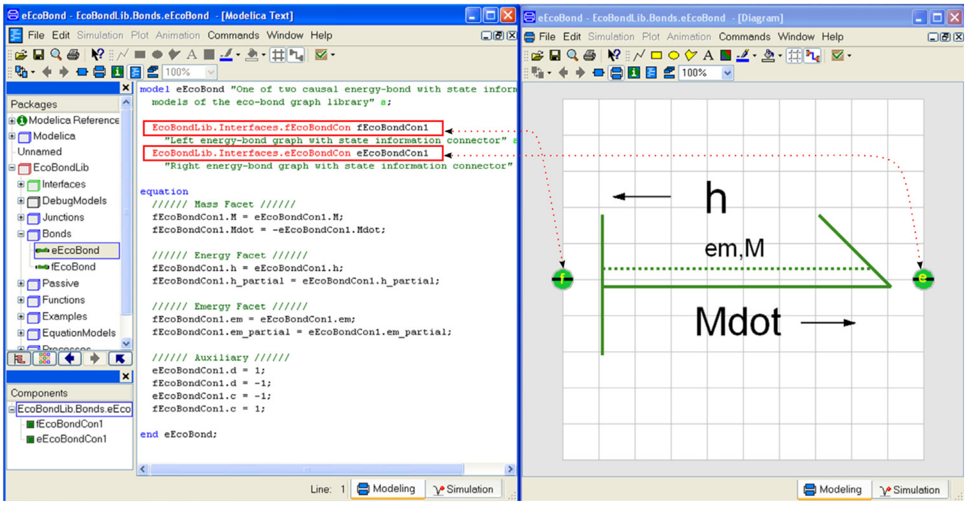
Note that whenever possible we declare variables using ISO-standard units provided by the `Modelica.SIunits` library. As a consequence, when performing operations at equations inside models, the units of the results are automatically managed by the Modelica compiler yielding results also in SI units.

The differences between `fEcoBondCon` and `eEcoBondCon` rely exclusively on the inversion of the `input` and `output` prefixes at the facet variables.

7.3.2. EcoBG bonds

Bonds are the simplest EcoBG elements using Connectors. They transport power and information variables' magnitudes unchanged, while setting the directional and causality variables according to the bond type.

The directional variable assumes a value of +1 at the side of the harpoon, and a value of −1 at the opposite end of the bond. The directional variable is used by the junction elements to sum either the flows or the efforts up correctly. The causality variable assumes a value of +1 at the side of the stroke, and a value of −1 at the opposite end of the bond.



```
model eEcoBond "One of two causal energy-bond with state inforn
  models of the eco-bond graph library" a;

  EcoBondLib.Interfaces.fEcoBondCon fEcoBondCon1
    "Left energy-bond graph with state information connector" a
  EcoBondLib.Interfaces.eEcoBondCon eEcoBondCon1
    "Right energy-bond graph with state information connector"

equation
  ////// Mass Facet //////
  fEcoBondCon1.M = eEcoBondCon1.M;
  fEcoBondCon1.Mdot = -eEcoBondCon1.Mdot;

  ////// Energy Facet //////
  fEcoBondCon1.h = eEcoBondCon1.h;
  fEcoBondCon1.h_partial = eEcoBondCon1.h_partial;

  ////// Emergy Facet //////
  fEcoBondCon1.em = eEcoBondCon1.em;
  fEcoBondCon1.em_partial = eEcoBondCon1.em_partial;

  ////// Auxiliary //////
  eEcoBondCon1.d = 1;
  fEcoBondCon1.d = -1;
  eEcoBondCon1.c = -1;
  fEcoBondCon1.c = 1;

end eEcoBond;
```

Fig. 17. The eEcoBond model of the EcoBG library. Dymola user interface: modelica text view (left) and diagram view (right).

Thus, two types of EcoBG bonds are available: the effort bond (Modelica `model eEcoBond` class) and the flow bond (Modelica `model fEcoBond` class).

In Fig. 17 the `eEcoBond` model is defined at the `EcoBondLib.Bonds` sub-library. The equations are shown in the Modelica text view (left) and the graphical representation is shown in the diagram view (right).

Bonds use one effort connector and one flow connector as their interfaces. The variables and icons for connectors are inherited by invoking the corresponding definitions of `eEcoBondCon` and `fEcoBondCon` classes (at the `EcoBondLib.Interfaces` sub-library), instantiating them locally as connectors `eEcoBondCon1` and `fEcoBondCon1`.

The code is provided in the Listing 2 below.

Listing 2. The eEcoBond bond model.

```
model eEcoBond
 "One of two causal bonds with state information models of
    the eco-bond graph library";
 EcoBondLib.Interfaces.fEcoBondCon fEcoBondCon1
 "Left energy-bond graph with state information connector";
 EcoBondLib.Interfaces.eEcoBondCon eEcoBondCon1
 "Right energy-bond graph with state information connector";
equation
  ////// Mass Facet //////
 fEcoBondCon1.M = eEcoBondCon1.M;
 fEcoBondCon1.Mdot = -eEcoBondCon1.Mdot;
 ////// Energy Facet //////
 fEcoBondCon1.h = eEcoBondCon1.h;
 fEcoBondCon1.h_partial = eEcoBondCon1.h_partial;
 ////// Emergy Facet //////
 fEcoBondCon1.em = eEcoBondCon1.em;
 fEcoBondCon1.em_partial = eEcoBondCon1.em_partial;
 ////// Auxiliary //////
 eEcoBondCon1.d = 1;
 fEcoBondCon1.d = -1;
 eEcoBondCon1.c = -1;
 fEcoBondCon1.c = 1;
end eEcoBond;
```

At the `equation` section the relation among variables are defined, basically letting facet variables pass from one connector to the other with their absolute values unchanged and inverting the sign of the flow variable (`Mdot`).

The auxiliary variables are set to represent the structure of an effort bond: (a) the energy is positive when it flows in the direction of the harpoon (from the flow connector into the effort connector: `fEcoBondCon1.d = −1` and `eEcoBondCon1.d = 1`) and (b) the flow connector imposes the causality of the flow variable `Mdot` (`fEcoBondCon1.c = 1`, and consequently `eEcoBondCon1.c = −1`).

The flow bond `fEcoBond` is defined identically to the effort bond excepting that the flow of energy is considered positive when it flows from the effort connector into the flow connector (`fEcoBondCon1.d = −1` and `eEcoBondCon1.d = 1`).

A comment is in order regarding the two information variables `em_partial` and `h_partial` that have not been referred to so far in this work. These are the "partial" versions of the power variables `em` and `h`, respectively. They are partial in the sense that are used at certain models as partial information required to calculate their "final" (or "total") power counterparts. In a nutshell, `h_partial` is used at O-Junctions to support a special feature by which, when `h` is left undefined by the modeler at a Storage element, an `h` is automatically (and dynamically) selected by the model to make the system thermodynamically locally viable. In the case of `em_partial`, it is used at 0-Junctions to collect emergy flows from process elements (only incoming emergy), pass them along to a storage element, and calculate there the total `em` which will be in turn imposed back to the 0-Junction and from there to the outgoing bonds. These mechanisms will be further detailed in a more technical, Modelica-specific publication currently under preparation. These mechanisms adhere at all times to the energy conservation principles of the power pair `Mdot*h` and its associated emergy `em`.

7.3.3. *EcoBG storage*

In the bond graph methodology, a *capacity* element stores the quantity transported by the flow variable, and is represented as a "C" element. Depending on the energy domain of the flow variable, the C element accumulates a given type of energy (potential, kinetic, electrical, etc.)

An EcoBG Storage is modeled as a Capacity Field, represented as a "CF" element. The "Field" extra descriptor expresses the fact that the EcoBG Storage accumulates simultaneously *several* energy domains, thus defining a multidimensional field. Indeed, EcoBG CF elements accumulate three dimensions of a given substance: mass, enthalpy and emergy.

In Fig. 18 the `EcoStorage` model is defined at the `EcoBondLib.Passive` sub-library (distinguishing it from elements at the `EcoBondLib.Active` sub-library, containing "Source" models not storing but bringing/consuming energy into/from the system, acting the system's boundary).

The code of the Modelica implementation for EcoBG Storage is provided in the Listing 3 below.

Listing 3. The EcoStorage storage model.

```
model EcoStorage "Capacitive Field representing substance
    storage"
extends EcoBondLib.Interfaces.EcoPassiveOnePort;
  parameter Modelica.SIunits.SpecificEnthalpy h_int "
      Specific enthalpy";
  parameter Modelica.SIunits.Mass M0 "Initial mass in the
      storage";
  parameter Modelica.SIunits.Energy EM0 "Initial emergy in
      the storage";
  Modelica.SIunits.Energy EM(start=0) "Emergy of stock";
  Modelica.SIunits.Energy H(start=0) "Enthalpy of stock";
  Modelica.SIunits.EnergyFlowRate EMdot "Flow of emergy";
  Modelica.SIunits.EnthalpyFlowRate Hdot "Flow of enthalpy";
  Real Tr(start=0) "Transformity";
equation
  if (initial()) then
      M = M0;
      EM = EM0;
  end if;
  ////// Mass Facet //////
  Mdot = der(M);
  ////// Energy Facet //////
  h = if (h_int==0) then h_partial else h_int;
  H = h * M;
  Hdot = h * Mdot;
  ////// Emergy Facet //////
  EM = em * M;
  EM = H  * Tr;
  EMdot = der(EM);
  EMdot = em_partial * Mdot;
end EcoStorage;
```

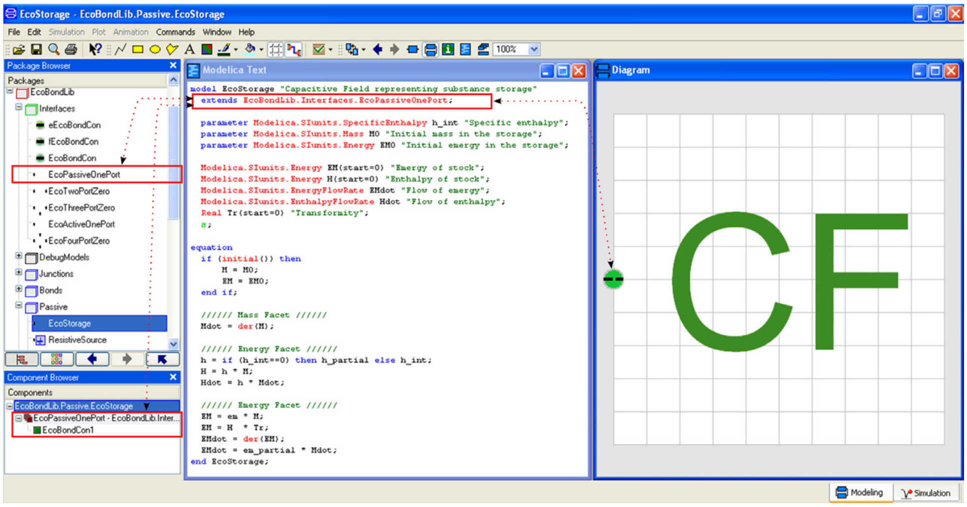
EcoBG Storage begins its definition with the object-oriented inheritance mechanism provided by `Modelica` through the `extends` command. This way, Storage reuses structure and properties defined in the model `EcoPassiveOnePort`. The latter is part of the `EcoBondLib.Interfaces` sub-library that provides models consisting of different sets of EcoBG `connectors`. In this case, `EcoPassiveOnePort` offers a single `EcoBondCon` connector (cf. the placement of these elements in the Text view, the Package Browser and the Component Browser in Fig. 18).

EcoBG storage has three `parameter` variables: the specific enthalpy per unit of mass accumulated (`h_int`, or "internal" enthalpy), the initial state of the mass (`M0`) and the initial state of the emergy (`EM0`). The latter two are set at the `equation` section when the `initial()` function evaluates to true (signaling the instant just before the commencement of the simulation).

At the mass facet, `Mdot` is set as the time derivative of the mass `M`: `Mdot=der(M)`. At the energy facet, `h` is set to the internal characteristic enthalpy of the accumulated substance: `h=h_int`.

At the energy facet, `H=h*M` states the relation between the specific and the total enthalpy, while `Hdot=h*Mdot` states that `Hdot` is statically related to `Mdot` through the specific enthalpy `h` (i.e. the integration of `Hdot` is achieved indirectly through the integration of `Mdot`). Besides, in the `if` statement at line 19, `h` can assume its value by means of either a fixed user parameterization (`h_int`) or a dynamic information coming from the rest of the system (`h_partial`).

At the Emergy facet, similar concepts apply. Emergy `EM` is integrated from its flow `EMdot` by means of `EMdot=der(EM)`. Additionally, the Transformity `Tr` is calculated as the emergy-to-enthalpy ratio in line 24.



```
model EcoStorage "Capacitive Field representing substance storage"
  extends EcoBondLib.Interfaces.EcoPassiveOnePort;

  parameter Modelica.SIunits.SpecificEnthalpy h_int "Specific enthalpy";
  parameter Modelica.SIunits.Mass M0 "Initial mass in the storage";
  parameter Modelica.SIunits.Energy EM0 "Initial emergy in the storage";

  Modelica.SIunits.Energy EM(start=0) "Emergy of stock";
  Modelica.SIunits.Energy H(start=0) "Enthalpy of stock";
  Modelica.SIunits.EnergyFlowRate EMdot "Flow of emergy";
  Modelica.SIunits.EnthalpyFlowRate Hdot "Flow of enthalpy";
  Real Tr(start=0) "Transformity";
  a;

equation
  if (initial()) then
     M = M0;
     EM = EM0;
  end if;

  ////// Mass Facet //////
  Mdot = der(M);

  ////// Energy Facet //////
  h = if (h_int==0) then h_partial else h_int;
  H = h * M;
  Hdot = h * Mdot;

  ////// Emergy Facet //////
  EM = em * M;
  EM = H  * Tr;
  EMdot = der(EM);
  EMdot = em_partial * Mdot;
end EcoStorage;
```

Fig. 18. The EcoStorage model of the EcoBG library. Dymola user interface: modelica text view (left) and diagram view (right).

7.3.4. *A full system: the case of OneStorage_EMTANK*

So far we have described models made up of interfaces and internal behavior, reaching the "component level" in the process of incrementally and hierarchically building systems.

The next level is the "system level" where components are interconnected to build more complex constructs out of simpler units.

We will show the Modelica implementation of the EcoBG EMTANK system presented before in Section 5.6 as an interconnection of EcoBG components.

Fig. 19 shows the Dymola user interface for the OneStorage_EMTANK model.

Note that all the definitions before the equation section are instantiations components previously defined and organized in different sub-libraries of the EcoBondLib main library. Meanwhile, all commands following equation are connect directives with their arguments consisting on pairs of connectors of the aforedeclared components (cf. Listing 4).

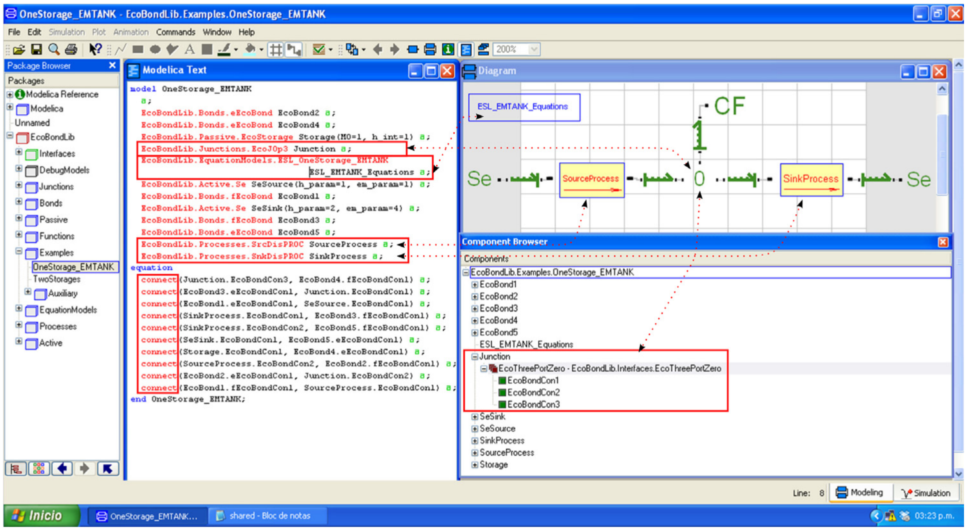
As we did in previous figures, some parts of the model have been highlighted (red rectangles and dotted arrows) that are worth commenting.

An EcoBG 0-Junction with three ports (EcoBondLib.Junctions.EcoJ0p3) has been used, and instantiated with name Junction. Note that each of its three connectors (Junction's EcoBondCon1, EcoBondCon2 and EcoBondCon3) are used in different connect commands. These connectors are available for use as soon as its "father" model EcoJ0p3 is declared, for it has been constructed extending the EcoThreePortZero interface (see the Component Browser at the bottom right of Fig. 19).

Just below, a totally non-connected model ESL_EMTANK_Equations is instantiated. This consists of an isolated set of mathematical equations that describe the EMTANK system using Odum's original approach to emergy modeling (i.e. not using EcoBG models, but the ESL language). Then we compare ESL variables against those calculated by the EcoBG elements (as it was done in Section 5.6.2) for model verification purposes.

Two process models take part of this system: SourceProcess (model EcoBondLib.Processes.SrcDisPROC) and SinkProcess (EcoBondLib.Processes.SnkDisPROC). They implement the equations as described in Sections 5.5.1 and 5.5.2, respectively.

The Modelica implementation of OneStorage_EMTANK is provided in the Listing 4.



```
model OneStorage_EMTANK
  a;
  EcoBondLib.Bonds.eEcoBond EcoBond2 a;
  EcoBondLib.Bonds.eEcoBond EcoBond4 a;
  EcoBondLib.Passive.EcoStorage Storage(M0=1, h_int=1) a;
  EcoBondLib.Junctions.EcoJ0p3 Junction a;
  EcoBondLib.EquationModels.ESL_OneStorage_EMTANK
                                     ESL_EMTANK_Equations a;
  EcoBondLib.Active.Se SeSource(h_param=1, em_param=1) a;
  EcoBondLib.Bonds.fEcoBond EcoBond1 a;
  EcoBondLib.Active.Se SeSink(h_param=2, em_param=4) a;
  EcoBondLib.Bonds.fEcoBond EcoBond3 a;
  EcoBondLib.Bonds.eEcoBond EcoBond5 a;
  EcoBondLib.Processes.SrcDisPROC SourceProcess a;
  EcoBondLib.Processes.SnkDisPROC SinkProcess a;
equation
  connect(Junction.EcoBondCon3, EcoBond4.fEcoBondCon1) a;
  connect(EcoBond3.eEcoBondCon1, Junction.EcoBondCon1) a;
  connect(EcoBond1.eEcoBondCon1, SeSource.EcoBondCon1) a;
  connect(SinkProcess.EcoBondCon1, EcoBond3.fEcoBondCon1) a;
  connect(SinkProcess.EcoBondCon2, EcoBond5.fEcoBondCon1) a;
  connect(SeSink.EcoBondCon1, EcoBond5.eEcoBondCon1) a;
  connect(Storage.EcoBondCon1, EcoBond4.eEcoBondCon1) a;
  connect(SourceProcess.EcoBondCon2, EcoBond2.fEcoBondCon1) a;
  connect(EcoBond2.eEcoBondCon1, Junction.EcoBondCon2) a;
  connect(EcoBond1.fEcoBondCon1, SourceProcess.EcoBondCon1) a;
end OneStorage_EMTANK;
```

Fig. 19. OneStorage_EMTANK. Dymola user interface: modelica text view (left), diagram view (top right) and component browser (bottom right).

Listing 4. The OneStorage_EMTANK system model.

```
model OneStorage_EMTANK
 EcoBondLib.Bonds.eEcoBond EcoBond2;
 EcoBondLib.Bonds.eEcoBond EcoBond4;
 EcoBondLib.Passive.EcoStorage Storage(M0=1, h_int=1);
 EcoBondLib.Junctions.EcoJ0p3 Junction;
 EcoBondLib.EquationModels.ESL_OneStorage_EMTANK
                                    ESL_EMTANK_Equations;
 EcoBondLib.Active.Se SeSource(h_param=1, em_param=1);
 EcoBondLib.Bonds.fEcoBond EcoBond1;
 EcoBondLib.Active.Se SeSink(h_param=2, em_param=4);
 EcoBondLib.Bonds.fEcoBond EcoBond3;
 EcoBondLib.Bonds.eEcoBond EcoBond5;
 EcoBondLib.Processes.SrcDisPROC SourceProcess;
 EcoBondLib.Processes.SnkDisPROC SinkProcess;
equation
 connect(Junction.EcoBondCon3, EcoBond4.fEcoBondCon1);
 connect(EcoBond3.eEcoBondCon1, Junction.EcoBondCon1);
 connect(EcoBond1.eEcoBondCon1, SeSource.EcoBondCon1);
 connect(SinkProcess.EcoBondCon1, EcoBond3.fEcoBondCon1);
 connect(SinkProcess.EcoBondCon2, EcoBond5.fEcoBondCon1);
 connect(SeSink.EcoBondCon1, EcoBond5.eEcoBondCon1);
 connect(Storage.EcoBondCon1, EcoBond4.eEcoBondCon1);
 connect(SourceProcess.EcoBondCon2, EcoBond2.fEcoBondCon1);
 connect(EcoBond2.eEcoBondCon1, Junction.EcoBondCon2);
 connect(EcoBond1.fEcoBondCon1, SourceProcess.EcoBondCon1);
end OneStorage_EMTANK;
```

As it can be seen, the listing above does not present equations within the `equation` section, but only `connect` statements, i.e. purely structural directives.

All EcoBG systems are composed in a similar way, instantiating and connecting EcoBG components. In these cases, the user composes the system model entirely in the graphical window (diagram view) while the textual model (text view) gets generated in a completely automated fashion, encompassing the graphical definition.

This is the case also for the system `Examples.TwoStorages` studied in Section 6.

8. Conclusions

We presented EcoBG, a novel energy-based modeling framework for complex dynamic systems with a focus on sustainability and emergy flows. By tracking simultaneously flows of mass and their attached specific enthalpy and emergy, EcoBG can check for the thermodynamic feasibility of the models, pinpointing physically unfeasible phases that cannot be detected in advance via static inspections of the models definition. EcoBG offers a low-level, domain-independent plumbing technology useful for building higher-level components tailored for specific application domains or modeling communities. EcoBG was specifically designed with generality and scalability in mind. Models of increasing complexity can be built hierarchically while preserving all energy self-checking features. With EcoBG we offer a sound framework into which to embed the basic ideas behind emergy tracking, naturally lending to a mathematical formulation of emergy dynamics that circumvents the physically awkward original proposition of switching differential equations.

Although not demonstrated in this article, also available in EcoBG is the ability to simulate models with only partial knowledge about the specific enthalpy values associated with flows of masses. By only stating specific enthalpy values at sources, EcoBG will automatically choose values for unknown specific enthalpy values throughout the inner system that do not violate energy conservation principles. This can be very helpful at the early exploratory stages of a modeling process.

An EcoBG model can also be connected with other types of Modelica models. This makes it possible to mix sustainability models with other model types based on first principles in different physical domains (e.g. mechanic, electric, hydraulic, etc.) drawing from the Modelica Standard Library (MSL) [22], which offers a rich palette of sophisticated models covering many physical phenomena from a wide range of energy domains.

Also benefitting from the Modelica underlying technologies, a wealth of optimization/automatic control techniques becomes easily accessible and applicable in EcoBG. An example of this could be designing processes that self-adapt their consumption/production rates according to (possibly sophisticated combinations of) dynamically calculated sustainability indices.

This is very appealing for studying sustainability of systems involving interactions between natural and industrial processes, as for the latter we can readily inherit a vast knowledge base of models developed within the Modelica community over many years [16,22].

9. Next steps

We are still one important step away from claiming EcoBG to be a full-fledged technology in terms of its emergy accounting capabilities.

There exist definite rules, called Emergy Algebra [6] to assign emergy to flows of energy in special situations, such as e.g. a process generating more than one output (by-products), an output being split after it left a process (flow splits), etc. Among these, the most relevant in our view is the rule stating that emergy shall not be counted twice when recirculating through a process that it had passed through already once after completing a feedback loop, e.g. material recycling. In EcoBG, we are currently forced to manually deactivate the emergy carried by flows passing through a feedback loop. This implies that the user needs to detect in advance all looping flows, an approach that is clearly error prone (or even unfeasible when the size of the system grows). Yet, human-assisted approaches to feedback loops are the de facto norm in most ESL implementations today (with the EmSim simulator [32] being an exception to the rule). We shall make EcoBG capable of dealing with automated loop detection rules in the next version of the formalism. We shall also supply a richer set of higher-level customized models, in particular PROC elements implementing well known functional relationships in sustainability science.

References

[1] T.F.H. Allen, T.W. Hoekstra, The confusion between scale-defined levels and conventional levels of organization in ecology, J. Veg. Sci. 1 (1990) 5–12.
[2] T.F.H. Allen, T.B. Starr, Hierarchy: Perspectives for Ecological Complexity, University of Chicago Press, Chicago, 1982.
[3] A. Baral, B.R. Bakshi, Thermodynamic metrics for aggregation of natural resources in life cycle analysis: insight via application to some transportation fuels, Environ. Sci. Technol. 44 (2009) 800–807.
[4] W. Borutzky, Bond Graph Methodology: Development and Analysis of Multidisciplinary Dynamic System Models, Springer Verlag, London, 2010.
[5] B. Breckling, F. Jopp, H. Reuter, Historical background of ecological modelling and its importance for modern ecology, in: F. Jopp, H. Reuter, B. Breckling (Eds.), Modelling Complex Ecological Dynamics, Springer, Berlin, Heidelberg, 2011, pp. 29–40.
[6] M. Brown, R.A. Herendeen, Embodied energy analysis and emergy analysis: a comparative view, Ecol. Econ. 19 (1996) 219–235.
[7] M. Brown, S. Ulgiati, Emergy-based indices and ratios to evaluate sustainability: monitoring economies and technology toward environmentally sound innovation, Ecol. Eng. 9 (1997) 51–69.
[8] D. Campbell, Using energy systems theory to define, measure, and interpret ecological integrity and ecosystem health, Ecosyst. Health 6 (2000) 181–204.
[9] R. Castro, D. Tilley, Personal communications, 2012.
[10] F.E. Cellier, Continuous System Modeling, Springer, New York, 1991.
[11] F.E. Cellier, J. Greifeneder, Thermobondlib, a new modelica library for modeling convective flows, in: Proceedings of the Sixth International Modelica Conference, Vol. 1, Bielefeld, Germany, 2008, pp. 163–172.
[12] F.E. Cellier, E. Kofman, Continuous System Simulation, Springer, New York, 2006.
[13] F.E. Cellier, A. Nebot, The modelica bond graph library, in: Proceedings of the Fourth International Modelica Conference, Vol. 1, Hamburg, Germany, 2005, pp. 57–65.
[14] Dynasim, Dymola, 2014, URL http://www.dynasim.se (accessed February 2014).
[15] A. Fischlin, Interactive modeling and simulation of environmental systems on workstations, in: D.P.F. Möller, O. Richter (Eds.), Analysis of Dynamic Systems in Medicine, Biology, and Ecology, Vol. 275 of Informatik-Fachberichte, Springer, Berlin, 1991, pp. 131–145.
[16] P. Fritzson, Principles of Object Oriented Modeling and Simulation with Modelica 3.3: A Cyber-Physical Approach, second ed., Wiley IEEE Press, 2014.
[17] H. Haberl, H. Weisz, C. Amann, A. Bondeau, N. Eisenmenger, K.H. Erb, M. Fischer-Kowalski, F. Krausmann, The energetic metabolism of the European union and the united states: decadal energy input time series with an emphasis on biomass, J. Ind. Ecol. 10 (2006) 151–171.
[18] J.L. Hau, B.R. Bakshi, Promise and problems of emergy analysis., Ecol. Model. 178 (2004) 215–225.
[19] D. Krahl, Extend: an interactive simulation tool, in: Simulation Conference, 2003. Proceedings of the 2003 Winter, Vol. 1, IEEE, 2003, pp. 188–196.
[20] R.M. May, et al., Simple mathematical models with very complicated dynamics, Nature 261 (5560) (1976) 459–467.
[21] Modelica Association, Modelica homepage, 2014, URL https://www.modelica.org (accessed February 2014).
[22] Modelica Association, The Modelica Standard Library, 2014, URL https://www.modelica.org/libraries (accessed February 2014).
[23] E.P. Odum, Fundamentals of Ecology, first ed., W.B. Saunders Comp., Philadelphia, 1959.
[24] H.T. Odum, Systems Ecology: An Introduction, John Wiley & Sons, New York, 1983.
[25] H.T. Odum, Energy, Environment and Public Policy: A Guide to the Analysis of Systems. UNEP Regional Seas Reports and Studies, The Hague, UNEP, 1988.
[26] H.T. Odum, Ecological and General Systems: An Introduction to Systems Ecology, third ed., University Press of Colorado, Niwot, CO, 1994.
[27] H.T. Odum, Environmental Accounting: Emergy and Environmental Decision Making, John Wiley & Sons, New York, 1996.
[28] H.T. Odum, N. Peterson, Simulation and evaluation with energy systems blocks, Ecol. Modell. 93 (1) (1996) 155–173.
[29] R.V. O'Neill, D.L. Angelis de, J.B. Waide, T.F.H. Allen, A Hierarchical Concept of Ecosystems, vol. 23, Princeton University Press, Princeton, 1986.
[30] J. Rockström, W. Steffen, K. Noone, A. Persson, F.S. Chapin, E.F. Lambin, T.M. Lenton, M. Scheffer, C. Folke, H.J. Schellnhuber, B. Nykvist, C.A. de Wit, T. Hughes, R.H. van der Leeuw, S.S. Sörlin, P.K. Snyder, R. Costanza, U. Svedin, K.L. Falkenmark, M.R.W. Corell, H.J.W.B.L.D. Fabry, V.J.K. Richardson, P. Crutzen, J.A. Foley, A safe operating space for humanity, Nature 461 (7263) (2009) 472–475, URL http://www.nature.com/nature/journal/v461/n7263/full/461472a.html
[31] D.R. Tilley, Exploration of Odum's dynamic emergy accounting rules for suggested refinements, Ecol. Model. 279 (2014) 36–44.
[32] R. Valyi, E. Ortega, Emergy simulator, an open source simulation platform dedicated to systems ecology and emergy studies, 2004, pp. 349–360.
[33] D. Zimmer, F.E. Cellier, The modelica multi-bond graph library, in: Proceedings of the Fifth International Modelica Conference, Vol. 2, Vienna, Austria, 2006, pp. 559–568.

Rodrigo Castro received his Electronic Engineer (MAScEE) (2004) and Ph.D. (2010) degrees from the National University of Rosario (UNR), Argentina. Since 2007 he is a Lecturer at the Computer Science Department, School of Exact and Natural Sciences, University of Buenos Aires (UBA), Argentina. In 2011, he was appointed an Assistant Researcher at the National Scientific and Technical Research Council (CONICET), Argentina. Since 2008 he visits regularly the Carleton University in Ottawa, Canada, and since 2012 he is a postdoc visiting researcher at ETH Zurich, Switzerland. He received an Emerging Leaders in the Americas award in 2008 from the Government of Canada and in 2013, and he was awarded the DEVS Best Ph.D. Dissertation Award by the Society for Modeling and Simulation International (SCS).

François E. Cellier received his B.S. degree in electrical engineering in 1972 and his Ph.D. degree in technical sciences in 1979 from the Swiss Federal Institute of Technology (ETH) Zurich. He worked at the University of Arizona as professor of Electrical and Computer Engineering from 1984 until 2005. He then returned to his home country of Switzerland and his alma mater. He has authored more than 200 technical publications. He published a textbook on Continuous System Modeling in 1991 and a second textbook on Continuous System Simulation in 2006, both with Springer-Verlag, New York. He is a fellow of the Society for Modeling and Simulation International (SCS).

Andreas Fischlin studied Biology and Systems Theory and did ecological research in Canada. He returned to Switzerland ETH Zurich to teach systems ecology and computer science. He played a leading role in the design and formation of the novel curriculum and department Environmental Sciences at ETH Zurich. He was presented the Peace Nobel Prize 2007 as awarded to IPCC for his work as coordinating lead author in the 2nd and 4th Assessment Report of IPCC. He represents Swiss scientists in the Swiss delegation in UNFCCC negotiations and has published numerous scientific works, notably on climate change impacts on ecosystems.