On a Monadic Encoding of Continuous Behaviour (extended abstract)

Renato Neves

INESC TEC (HASLab) & University of Minho, Portugal
rjneves@inescporto.pt

Abstract. The original purpose of component–based development was to provide techniques to master complex software, through composition, reuse, and parametrisation. However, such systems are rapidly moving towards a level in which they become prevalently intertwined with (continuous) physical processes. A possible way to accommodate the latter in component calculi relies on a suitable encoding of continuous behaviour as (yet another) computational effect.
This paper reports such an encoding through a monad which, in the compositional development of hybrid systems, may play a role similar to the one played by the maybe, powerset, and distribution monads in the characterisation of partial, non deterministic and probabilistic components, respectively.

Keywords: Monads, components, hybrid systems, control theory

1 Introduction

Component-based software development is often explained through a visual metaphor: a palette of computational units, and a blank canvas in which they are dropped and interconnected by drawing wires abstracting different composition and synchronisation mechanisms. More and more, however, components are not limited to traditional information processing units, but encapsulate some form of interaction with physical processes. The resulting systems, referred to as *hybrid*, exhibit a complex dynamics in which *loci* of computation, coordination, and control of physical processes interact, become mutually constrained, and cooperate to achieve specific goals.

One way of looking at components, proposed in [1, 2], emphasises an *observational* semantics, through a signature of observers and methods, making them amenable to a *coalgebraic* [3] characterisation as (generalisations of) abstract Mealy machines. The resulting calculus is parametric on whatever behavioural model underlies a component specification. This captures, for example, partial, non deterministic or probabilistic evolution of a component's dynamics by encoding such behavioural effects as *strong monads* [4, 5].

This paper summarises a number of results developed in the context of the author's PhD project [6]. Namely, the introduction of a strong monad $\mathcal{H}$ [7] that subsumes *continuous* behaviour and the study of the corresponding Kleisli category [8] as the mathematical space in which the underlying behaviour can be isolated and its effect over different forms of composition studied. This work may pave the way to the development of a *coalgebraic* calculus of *hybrid components*.

Related work. A few categorial models for hybrid systems have been proposed. For example, document [9] introduced an institution – in essence, a categorial rendering of logic – for hybrid systems and provided basic forms of composition. Around the same time, Jacobs [10] suggested a coalgebraic framework where hybrid systems are viewed as coalgebras equipped with a monoid action. Some years later Haghverdi *et. al* [11] provided a formalisation of hybrid systems using a conceptual framework that is closer to the coalgebraic perspective.

The monad reported in this paper captures the typical continuous behaviour of hybrid systems. Actually, there is a close relationship between the work reported here and Peter Höfner's algebra of hybrid systems [12]: the latter's main operator and its laws are embedded in the (sequential) composition of $Kl\mathcal{H}$, the Kleisli category for monad $\mathcal{H}$.

Since our approach, differently from Höfner's calculus, is structured around a monad that encodes continuous evolution, a number of canonical constructions come for free. Moreover, the integration with other behavioural effects, such as non determinism or probabilistic evolution, becomes more systematic.

Roadmap. After a brief detour on preliminaries and notation in Section 2, monad $\mathcal{H}$ is described in Section 3. Section 4 gives some details about the corresponding Kleisli category $Kl\,\mathcal{H}$, characterising composition and some (co)limits. Finally, conclusions and possible future research directions are discussed in Section 5. In this paper many calculations adopt a pointfree style in the spirit of the Bird-Meertens formalism [13].

2 Preliminaries

2.1 The category of topological spaces

The typical *continuous* behaviour of hybrid systems suggests the category $\mathbb{T}op$ of topological spaces and continuous functions as a suitable working environment for developing the aforementioned results. In the sequel, if the context is clear, a topological space will be denoted just by its underlying set. Also, assume that spaces $X \times Y$, $X + Y$ correspond to the canonical product and coproduct of X, Y, respectively, and that whenever Y is *core-compact*, space X^Y comes with the *exponential* topology [14]. In this context, given a continuous function $f : X \times Y \to Z$ where Y is core–compact, we denote its curried version by

$\lambda f : X \to Z^Y$. Moreover, we will use the following isomorphisms in $\mathbb{T}op$:

$$\alpha_l : (X \times Y) \times Z \cong X \times (Y \times Z)$$
$$sw : X \times Y \cong Y \times X$$
$$i : (X \times Y)^{\mathbb{R}_0} \cong X^{\mathbb{R}_0} \times Y^{\mathbb{R}_0}$$

2.2 Notation

Arrows $X \to 1$ to the final object in $\mathbb{T}op$ will be denoted by !, and a function constantly yielding a value x by $\underline{x}$. Given two functions $f, g : X \to Y$, and a predicate p, conditional expression $f \lhd p \rhd g : X \to Y$ is defined by

$$(f \lhd p \rhd g)\, x = (f\, x \lhd p\, x \rhd g\, x) = \begin{cases} f\, x & p\, x \\ g\, x & \text{otherwise} \end{cases}$$

The continuous functions *minimum* $\curlywedge : \mathbb{R} \times (\mathbb{R}+1) \to \mathbb{R}$ and *truncated subtraction* $\ominus : \mathbb{R} \times (\mathbb{R}+1) \to \mathbb{R}$ play a key role in the sequel. They are defined as follows

$$r \curlywedge (i_1\, s) = (\pi_1 \lhd (\leq) \rhd \pi_2)\, (r, s) \qquad r \ominus (i_1\, s) = ((-) \lhd (>) \rhd \underline{0})\, (r, s)$$
$$r \curlywedge (i_2\, \star) = r \qquad r \ominus (i_2\, \star) = 0$$

where $\leq, >$ are the usual ordering relations over the reals, and 1 introduces *infinity*. Set $\mathbb{R}_0$ denotes the non–negative real numbers. Then, we have $(\curlywedge_d)\, r = r \curlywedge d$ and $(\ominus_d)\, r = r \ominus d$. Finally, for any category $\mathbb{C}$, $|\mathbb{C}|$ denotes the corresponding class of objects.

3 A Monad for Continuity

Formally, we see *continuous systems* as arrows of the type

$$I \to O^{\mathbb{R}_0} \times D$$

where $D = \mathbb{R}_0 + 1$ and I, O are the input and output spaces, respectively. The intuition is that outputs of such systems are *continuous evolutions* (also known as *trajectories*) with a specific (possibly infinite) duration $d \in D$.

Definition 1 *$\mathcal{H} : \mathbb{T}op \to \mathbb{T}op$ is a functor such that, for any objects $X, Y \in |\mathbb{T}op|$ and any continuous function $g : X \to Y$,*

$$\mathcal{H}X = \{\, (f, d) \in X^{\mathbb{R}_0} \times D \mid f \cdot \curlywedge_d = f \,\}$$
$$\mathcal{H}g = g \cdot \times id$$

where $(g\, \cdot)\, h = g \cdot h$. Condition $f \cdot \curlywedge_d = f$ tells that function f must become constant after reaching its duration; more formally, for any $r \in \mathbb{R}_0$ such that $r > d$, $f\, r = f\, d$. Hence, continuous systems become arrows of the type

$$I \to \mathcal{H}O$$

also denoted as $I \twoheadrightarrow O$.

The crucial step now is to equip $\mathcal{H}$ with a monad structure, *i.e.* with natural transformations

$$\eta : Id \stackrel{.}{\rightarrow} \mathcal{H}, \;\; \mu : \mathcal{H}\mathcal{H} \stackrel{.}{\rightarrow} \mathcal{H}.$$

First,

Definition 2 *Given any $X \in |\mathbb{T}op|$, define $\eta_X : X \rightarrow \mathcal{H}X$ such that*

$$\eta_X \; x = (\underline{x}, i_1 \, 0)$$

in pointfree notation $\eta_X = \langle \lambda \pi_1, i_1 \cdot \underline{0} \rangle$.

The definition of μ is more demanding.

Definition 3 *Define the continuous functions $g : \mathcal{H}\mathcal{H}X \times \mathbb{R}_0 \rightarrow X^{\mathbb{R}_0}$, $h : \mathcal{H}\mathcal{H}X \times \mathbb{R}_0 \rightarrow \mathbb{R}_0$ such that*

$$g((f,d),r) = (\pi_1 \cdot f) \; (r \curlywedge d),$$
$$h((f,d),r) = r \ominus d$$

Next, we have $fl_1 : \mathcal{H}\mathcal{H}X \rightarrow X^{\mathbb{R}_0}$ where $fl_1 = \lambda(ev \cdot \langle g, h \rangle)$. In pointwise notation, fl_1 is defined as

$$fl_1(f,d) = ev \cdot \langle \pi_1 \cdot f \cdot \curlywedge_d, \ominus_d \rangle$$

Then, define function $fl_2 : \mathcal{H}^2 X \rightarrow D$ such that

$$fl_2 \; (f,d) = ((\pi_2 \cdot f) \; d \; \lhd (d \not\in 1) \; \rhd \; i_2 \star) + d$$

Finally, we define for any $X \in |\mathbb{T}op|$, $\mu_X = \langle fl_1, fl_2 \rangle$.

Intuitively, operation μ_X 'concatenates' functions: given a pair $(f,d) \in \mathcal{H}\mathcal{H}X$, μ_X concatenates function $(\pi_1 \cdot f) \text{ - } 0 : [0,d] \rightarrow X$ with $(\pi_1 \cdot f) \; d \text{ - } : [0,d'] \rightarrow X$, and sums the corresponding durations.

Theorem 1 *The triple $\langle \mathcal{H}, \eta, \mu \rangle$ forms a monad.*

Proof. In document [7].

4 The Category of Continuous Behaviours

4.1 Kleisli Composition

The Kleisli category for $\mathcal{H}$ $(Kl\mathcal{H})$ provides an interesting setting to study the requirements placed by continuity over different forms of composition; actually, the envisaged component calculus for hybrid systems is essentially its calculus. This motivates the study of $Kl \, \mathcal{H}$, summmarised in the current section.

The definition of Kleisli composition in $Kl \, \mathcal{H}$ suggests a relevant distinction between continuous systems.

Definition 4 *A continuous system $c : I \to \mathcal{H}I$ is* passive *if the following diagram commutes*

$$\begin{tikzcd}
I \arrow[r, "f_c"] \arrow[dr, "id"'] & I^{\mathbb{R}_0} \arrow[d, "ev \cdot \langle id, \underline{0} \rangle"] \\
& I
\end{tikzcd}$$

where $f_c = \pi_1 \cdot c$. It is active *otherwise.*

Intuitively, the diagram tells that any evolution triggered by c 'starts' at the point given as input. To see why such a distinction is relevant, let us consider two continuous systems $c_1 : I \to \mathcal{H}K$, $c_2 : K \to \mathcal{H}O$. Through Kleisli composition we obtain component $c_2 \bullet c_1 : I \to \mathcal{H}O$ whose behaviour is computed as follows:

$$\begin{aligned}
& \pi_1 \cdot (c_2 \bullet c_1)\, x \\
= \quad & \{ \text{Kleisli composition} \} \\
& \pi_1 \cdot \mu \cdot \mathcal{H}c_2 \cdot c_1\, x \\
= \quad & \{ \text{Cancellation} \times \} \\
& fl_1 \cdot \mathcal{H}c_2 \cdot c_1\, x \\
= \quad & \{ \text{Definition of } \mathcal{H} \text{ (taking } d = (\pi_2 \cdot c_1)\, x \text{) } \} \\
& fl_1\, (c_2 \cdot (f_{c_1}\, x), d) \\
= \quad & \{ \text{Application} \} \\
& ev \cdot \langle \pi_1 \cdot c_2 \cdot (f_{c_1}\, x) \cdot \curlywedge_d, \ominus_d \rangle \\
= \quad & \{ \text{Notation} \} \\
& ev \cdot \langle f_{c_2} \cdot (f_{c_1}\, x) \cdot \curlywedge_d, \ominus_d \rangle
\end{aligned}$$

Going pointwise,

$$\begin{aligned}
& ev \cdot \langle f_{c_2} \cdot (f_{c_1}\, x) \cdot \curlywedge_d, \ominus_d \rangle\, t \\
= \quad & \{ \text{ Application } \} \\
& f_{c_2}\, \big(\, f_{c_1}\, x\, (t \curlywedge d) \,\big)\, (t \ominus d) \\
= \quad & \{ \text{ Notation } \} \\
& f_{c_2}\, \big(\, f_{c_1}\, x\, t \,\big)\, 0 \lhd (\, t \leq d \,) \rhd\, f_{c_2}\, \big(\, f_{c_1}\, x\, d \,\big)\, (t - d) \\
= \quad & \{ \text{ If } c_2 \text{ is passive } \} \\
& f_{c_1}\, x\, t \lhd (\, t \leq d \,) \rhd\, f_{c_2}\, \big(\, f_{c_1}\, x\, d \,\big)\, (t - d)
\end{aligned}$$

Assuming that c_2 is passive, the last expression tells that given an input $i \in I$ the resulting evolution corresponds to the evolution of the first component

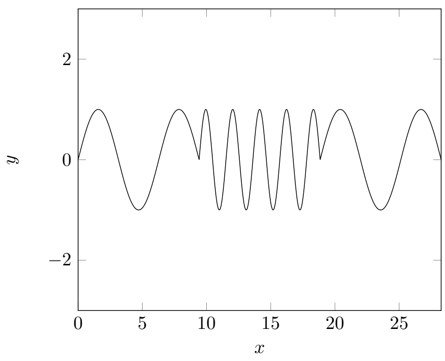
$f_{c_1}\, i$ ensued by the evolution of the second, which receives as input the 'last' point of evolution $f_{c_1}\, i$. Therefore, when c_2 is passive Kleisli composition may be alternatively called *sequential* composition or concatenation. On the other hand if c_2 is active, Kleisli composition tells that c_2 can alter the evolution of c_1 and then proceed with its own evolution. This is illustrated in the following examples.

Example 1. *Given two signal generators* $c_1, c_2 : \mathbb{R} \to \mathcal{H}\mathbb{R}$ *defined as*

$$\begin{aligned} c_1\, r &= (r + sin \text{ - }, 3\pi), \\ c_2\, r &= (r + sin\,(3 \times \text{ - }), 3\pi) \end{aligned}$$

the signal given by $c_1 \bullet (c_2 \bullet c_1)\; 0$ *yields the plot below*

$c_1 \bullet (c_2 \bullet c_1)\; 0$

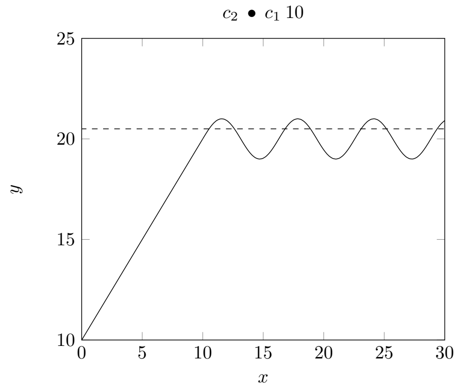


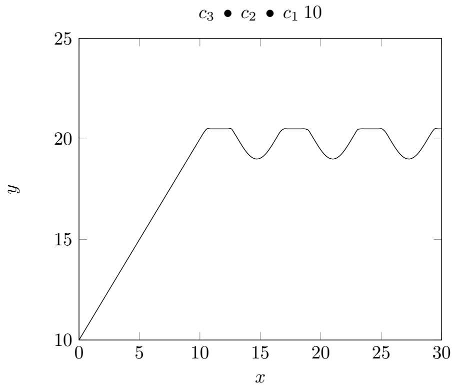
This type of signal is common in the domain of *frequency modulation*, where the varying frequency is used to encode information for electromagnetic transmission.

Example 2. *Suppose the temperature of a room is to be regulated according to the following discipline: starting at* 10 °C, *seek to reach and maintain* 20 °C, *but in no case surpass* 20.5 °C. *To realise such a system, three components have to work together:* c_1 *to raise the temperature to* 20 °C, *component* c_2 *to maintain a given temperature, and component* c_3 *to ensure the temperature never goes over* 20.5 °C. *Formally,*

$$\begin{aligned} c_1\, x &= (\; (x + \text{ - }),\; 20 \ominus x \;), \\ c_2\, x &= (\; x + (\sin \text{ - }),\; \infty \;), \\ c_3\, x &= (\; \underline{x} \;\lhd\; (x \leq 20.5) \;\rhd\; \underline{20.5}\, , 0 \;) \end{aligned}$$

One may then compose c_2, c_1 into $c_2 \bullet c_1$, which results in a component able to read the current temperature, raise it to 20 °C, and then keep it stable, as shown by the plot below on the left. If, however, temperatures over 20.5 °C occur, composition $c_3 \bullet c_2 \bullet c_1$ puts the system back into the right track as the plot below on the right illustrates.





On a different note, for any $X \in |\mathbb{T}op|$, arrow η_X is a *trivial* system in the sense that its evolutions always have duration zero and the only point in the trajectories is the input given. For this reason we will refer to η_X by $copy_X$, and often omit the subscript. Setting up $Kl\, \mathcal{H}$ yields the following laws

$$copy \bullet c_1 = c_1 \tag{1}$$

$$c_1 \bullet copy = c_1 \tag{2}$$

$$(c_3 \bullet c_2) \bullet c_1 = c_3 \bullet (c_2 \bullet c_1) \tag{3}$$

for any arrows c_1, c_2, c_3 in $Kl\, \mathcal{H}$.

4.2 (Co)limits and Tensorial Strength

(Co)limits are a main tool to build 'new' arrows from 'old' ones, which in the case of $Kl\, \mathcal{H}$ translates to new forms of (continuous) component composition. One important colimit is the *coproduct*, which provides the *choice* operator:

Definition 5 *Given two components $c_1 : I_1 \to \mathcal{H}O$, $c_2 : I_2 \to \mathcal{H}O$ component*

$$[c_1, c_2] : I_1 + I_2 \to \mathcal{H}O$$

behaves as c_1 if input I_1 is chosen, *and as c_2 otherwise. Diagrammatically,*

$$\begin{tikzcd}
I_1 \arrow[r, "\ulcorner i_1 \urcorner"] \arrow[dr, "c_1"'] & I_1 + I_2 \arrow[d, dotted, "{[c_1, c_2]}"] & I_2 \arrow[l, "\ulcorner i_2 \urcorner"'] \arrow[dl, "c_2"] \\
& O &
\end{tikzcd}$$

where, for any continuous function $f : X \to Y$, symbol $\ulcorner f \urcorner$ denotes copy $\cdot f$.

Since the choice operator comes from a colimit, a number of laws are given; one example is the following equation

$$c_3 \bullet [c_1, c_2] = [c_3 \bullet c_1, c_3 \bullet c_2] \tag{4}$$

An important limit of $Kl\ \mathcal{H}$ is the *pullback* below, which brings parallelism up front.

$$\begin{tikzcd}
I \arrow[rrd, bend left, "c_2"] \arrow[rd, dotted, "\gamma \cdot \langle c_1, c_2 \rangle"] \arrow[rdd, bend right, "c_1"'] & & \\
& K \times O \arrow[r, "\ulcorner \pi_2 \urcorner"] \arrow[d, "\ulcorner \pi_1 \urcorner"'] & O \arrow[d, "\ulcorner ! \urcorner"] \\
& K \arrow[r, "\ulcorner ! \urcorner"'] & 1
\end{tikzcd}$$

for $\gamma((f_1, d), (f_2, d)) = (\langle f_1, f_2 \rangle, d)$.

Intuitively, the diagrams states that whenever two components c_1, c_2 are compatible – in the sense that for any input the duration of their evolutions coincide (commutativity of the outer square) – we can define component $\gamma \cdot \langle c_1, c_2 \rangle$ whose output corresponds to the (paired) evolutions of c_1 and c_2.

Note that functions $\ulcorner \pi_1 \urcorner, \ulcorner \pi_2 \urcorner$ introduce trajectory elimination, due to their ability to remove one side of the paired evolution. Note also that $\ulcorner \triangle \urcorner : X \to \mathcal{H}(X \times X)$ duplicates trajectories, for $\triangle : X \to (X \times X)$ the diagonal function, and $\ulcorner sw \urcorner$ swaps evolutions.

Definition 6 *Given two compatible components $c_1 : I \to \mathcal{H}O_1$, $c_2 : I \to \mathcal{H}O_2$ component*

$$\langle\langle c_1, c_2 \rangle\rangle = \gamma \cdot \langle c_1, c_2 \rangle : I \to \mathcal{H}(O_1 \times O_2)$$

is the parallel execution of c_1, c_2.

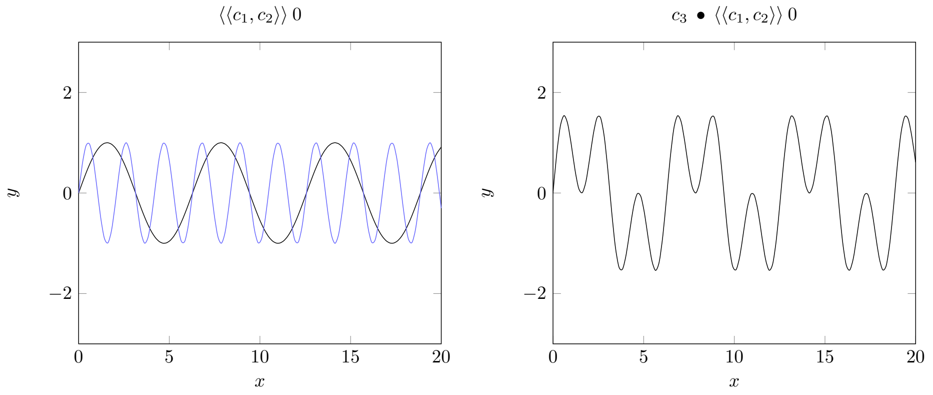
Since parallelism comes from a limit, we have again a number of laws for free; for instance

$$\langle\langle c_1, c_2 \rangle\rangle \bullet d = \langle\langle c_1 \bullet d, c_2 \bullet d \rangle\rangle \tag{5}$$

Example 3. *Consider two signal generators, c_1, c_2 such that*

$$\begin{aligned} c_1\, x &= (\, x + (\sin\ \text{-}\,), 20\,), \\ c_2\, x &= (\, x + \sin\,(3 \times\ \text{-}\,), 20\,) \end{aligned}$$

For input 0, their parallel evolution $\langle\langle c_1, c_2 \rangle\rangle$ is illustrated in the plot below on the left. Moreover, we can combine signals. For example, to add incoming signals, take the active component c_3, formally defined as $c_3(x, y) = (\underline{x + y}, 0)$. For input 0, the system $c_3 \bullet \langle\langle c_1, c_2 \rangle\rangle$ yields the plot shown below, on the right.



$\langle\langle c_1, c_2 \rangle\rangle\ 0$

$c_3 \bullet \langle\langle c_1, c_2 \rangle\rangle\ 0$

We close this section introducing a tensorial strength for monad $\mathcal{H}$ — which turns out to be an essential mechanism for the generation of a calculus for hybrid components.

Definition 7 *Tensorial strength for monad $\mathcal{H}$ is a natural transformation*

$$\tau : \mathcal{H}X \times Y \dot{\to} \mathcal{H}(X \times Y)$$

defined as $\tau = \langle f_1, f_2 \rangle$ where $f_1 : \mathcal{H}X \times Y \to (X \times Y)^{\mathbb{R}_0}$, $f_1((f, d), y) = \langle f, \underline{y} \rangle$, and $f_2 : \mathcal{H}X \times Y \to D$, $f_2((f, d), y) = d$.

Theorem 2 *$\langle \mathcal{H}, \eta, \mu \rangle$ is a strong monad.*

Proof. In document [7].

5 Conclusions and future work

Software systems are becoming prevalently intertwined with (continuous) physical processes. This renders their rigorous design (and analysis) a difficult challenge that calls for a wide, uniform framework where 'Continuous' Mathematics and Computer Science must work together. As a first step towards a calculus of hybrid components in the spirit of [2], this paper showed how continuous behaviour can be encoded in the form of a strong topological monad, and briefly explored the corresponding Kleisli category.

Our current research investigates how hybrid behaviour can be rendered by arrows typed as $\langle c, p \rangle : S \times I \to S \times \mathcal{H}O$, where $c : S \times I \to S$ is a discrete arrow (S comes equipped with the discrete topology) and $p : S \times I \to \mathcal{H}O$ is a continuous system. This paves the way to extending the component calculus in [2] to hybrid systems.

Acknowledgements. This work is funded by ERDF - European Regional Development Fund, through the COMPETE Programme, and by National Funds through FCT within project FCOMP-01-0124-FEDER-028923. The author is also sponsored by FCT grant SFRH/BD/52234/2013.

References

1. L. S. Barbosa, Components as coalgebras, Ph.D. thesis, DI, Minho University (2001).
2. L. S. Barbosa, Towards a calculus of state-based software components, Journal of Universal Computer Science 9 (2003) 891–909.
3. J. Rutten, Universal coalgebra: a theory of systems, Theoretical Computer Science 249 (1) (2000) 3 – 80, modern Algebra.
4. A. Kock, Strong functors and monoidal monads, Archiv der Mathematik 23 (1) (1972) 113–120.
5. E. Moggi, Notions of computation and monads, Information and computation 93 (1) (1991) 55–92.
6. R. Neves, Logics and calculi for hybrid components, Ph.D. thesis, DI, Universidade do Minho (2017).
7. R. Neves, L. S. Barbosa, D. Hofmann, M. A. Martins, Continuity as a computational effect, CoRR abs/1507.03219.
   URL http://arxiv.org/abs/1507.03219
8. R. Neves, L. S. Barbosa, M. A. Martins, A kleisli category for hybrid components, in: Formal Aspects of Component Software (FACS) 2015, submitted.
9. H. Lourenço, A. Sernadas, An institution of hybrid systems, in: D. Bert, C. Choppy, P. Mosses (Eds.), Recent Trends in Algebraic Development Techniques, Vol. 1827 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2000, pp. 219–236. doi:10.1007/978-3-540-44616-3_13.
   URL http://dx.doi.org/10.1007/978-3-540-44616-3_13
10. B. Jacobs, Object-oriented hybrid systems of coalgebras plus monoid actions, Theoretical Computer Science 239 (1) (2000) 41 – 95. doi:http://dx.doi.org/10.1016/S0304-3975(99)00213-3.
    URL http://www.sciencedirect.com/science/article/pii/S0304397599002133
11. E. Haghverdi, P. Tabuada, G. J. Pappas, Bisimulation relations for dynamical, control, and hybrid systems, Theor. Comput. Sci. 342 (2-3) (2005) 229–261. doi:10.1016/j.tcs.2005.03.045.
    URL http://dx.doi.org/10.1016/j.tcs.2005.03.045
12. P. Höfner, Algebraic calculi for hybrid systems, Ph.D. thesis, University of Augsburg (2009).
13. R. Bird, O. de Moor, The Algebra of Programming, Prentice-Hall, 1996.
    URL http://www.cs.ox.ac.uk/publications/books/algebra/
14. M. Escardó, R. Heckmann, Topologies on spaces of continuous functions, in: Topology Proceedings, Vol. 26, 2001, pp. 545–564.