Logics and Calculi for Cyber–Physical Components

Renato Neves

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

Abstract

This document is to be considered for an Early Ideas talk.

Keywords and phrases Monads, components, cyber–physical systems, control theory

Aimed at mastering the ever-increasing complexity of software systems, *component–based development* is a prominent paradigm that supports techniques based on composition, reuse, and parametrisation. The underlying idea is to start with small, simple computational units, treated as black–boxes (the components); then, depending on the objective, to combine them in different ways, giving rise to bigger and more complex systems whose emerging behaviour goes often beyond the plain composition of its constituents. This makes possible to shift the focus from component implementation to component composition, which favours a third–party assembly of 'off–the–shelf' implementations, and brings coordination of software components to a central place in software development.

Components can, therefore, be regarded as entities that announce a public interface for interaction with the environment but hide their internal memory which constrains the observable behaviour. Actually, such a notion is 'in line' with that of coalgebra – a formalisation of state–based system whose states are distinguishable only by what is observed"from the outside".

This alignment did not pass unnoticed and, consequently, a number of results advocate a *coalgebraic* formalisation of software component (*e.g.* [2, 4, 1]). In particular, reference [2] introduces a coalgebraic component calculus where generic techniques for composition and refinement can be suitably developed (*e.g.* [6]). Genericity is achieved by parametrizing the construction with a *strong monad* [7] to capture component's structural behaviour, instead of fixing it from the outset. Actually, each monad captures a specific behaviour, which is then reflected in the corresponding component calculus; for example, the *maybe monad* ($\mathcal{M}$) introduces partial components, the *powerset* monad ($\mathcal{P}$) non-deterministic ones, and the *distribution monad* ($\mathcal{D}$) brings (discrete) probabilistic behaviour into the scene.

Can a similar strategy give rise to a calculus of *cyber–physical* components? Such is the question we want to address. Indeed *cyber–physical*, or *hybrid* systems, are becoming ubiquitous. The qualifier encompasses all sorts of devices whose operation is deeply intertwined with physical processes – *e.g.* related to motion, thermodynamics, time – that are under control of and influence digital computations. Hence, any (formal) method for the design and analysis of cyber–physical systems needs to accommodate both *cyber* and *physical* behaviour, as well as their interactions. This challenge is still largely unmet by current formal techniques (but see [8, 5, 9]).

In this context, we propose and characterise a strong monad ($\mathcal{H}$) that encodes continuous behaviour, and can therefore assume a role similar to the one played by $\mathcal{M}$, $\mathcal{P}$, and $\mathcal{D}$ monads in the characterisation of partial, non–deterministic and probabilistic components, respectively.

Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Monad $\mathcal{H}$ is developed over the category of topological spaces – where continuity is a main concern. Its Kleisli category, in essence a prototypical calculus for continuous components, offers a suitable environment to study the effects of continuity over (different forms of) composition. In particular, we will define and discuss sequential composition and mechanisms for synchronisation and parallelisation of continuous components. More generally, we will discuss the following questions:

- *how can components exhibiting continuous behaviour be composed (among themselves and with discrete components as well), adapted and refined?*
- *Is there an algebra (a calculus) for this? Can such a calculus be developed on top of a coalgebraic semantics over a suitable category?*

The research questions presented above focus, essentially, on semantic aspects, with little attention being paid to the logical counterpart. Property–oriented specification is, however, a complementary way for defining component interfaces, which, as discussed in [3], also has a close relation to coalgebra theory. Actually, each endofunctor F (on the category of sets and set-theoretic functions) generates a modal logic whose transition dynamics is defined by the category of F–coalgebras. Thus, one may ask: *"which kind of modal logics monad $\mathcal{H}$ (and its underlying functor) give rise to?"*

This question and the ones presented above concerning semantic issues are at the foundations of logics and calculi for cyber–physical components. To find their answers is the main objective in the author's (starting) PhD project.

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-020537. The author is also sponsored by FCT grant SFRH/BD/52234/2013.

References

1 Marc Aiguier, Frédéric Boulanger, and Bilal Kanso. A formal abstract framework for modelling and testing complex software systems. *Theor. Comput. Sci.*, 455:66–97, 2012.

2 Luís S. Barbosa. *Components as coalgebras.* PhD thesis, DI, Minho University, 2001.

3 Corina Cîrstea, Alexander Kurz, Dirk Pattinson, Lutz Schröder, and Yde Venema. Modal logics are coalgebraic. In Samson Abramsky, Erol Gelenbe, and Vladimiro Sassone, editors, *Visions of Computer Science, BCS International Academic Research Conference (BCS 2008)*, pages 129–140. British Computer Society, 2008.

4 Ichiro Hasuo, Bart Jacobs, and Ana Sokolova. The microcosm principle and concurrency in coalgebra. In *Foundations of Software Science and Computational Structures*, pages 246–260. Springer, 2008.

5 Peter Höfner. *Algebraic calculi for hybrid systems.* PhD thesis, University of Augsburg, 2009.

6 Sun Meng and Luís S. Barbosa. Components as coalgebras: The refinement dimension. *Theoretical Computer Science*, 351(2):276 – 294, 2006.

7 Eugenio Moggi. Notions of computation and monads. *Information and computation*, 93(1):55–92, 1991.

8 André Platzer. *Logical Analysis of Hybrid Systems: Proving Theorems for Complex Dynamics.* Springer, Heidelberg, 2010.

9 Paulo Tabuada. *Verification and Control of Hybrid Systems - A Symbolic Approach.* Springer, 2009.