

Monad-based logics for computational effects

Till Mossakowski

DFKI Laboratory, Bremen, and
Department of Computer Science, University of Bremen

The presence of computational effects, such as state, store, exceptions, input, output, non-determinism, backtracking etc., complicates the reasoning about programs. In particular, usually for each effect (or each combination of these), an own logic needs to be designed.

Monads are a well-known tool from category theory that originally has been invented for studying algebraic structures. Monads have been used very successfully by Moggi [2] to model computational effects (in particular, all of those mentioned above) in an elegant way. This has been applied both to the semantics of programming languages (e.g. [4, 10, 1, 9]) and to the encapsulation of effects in pure functional languages such as Haskell [11].

Several logics for reasoning about monadic programs have been introduced, such as evaluation logic [6, 3], Hoare logic [7] and dynamic logic [8, 5]. Some of these logics have a semantics and proof calculus given in a completely monad independent (and hence, effect independent) way. We give an overview of these logics, discuss completeness of their calculi, as well as some application of these logics to the reasoning about Haskell and Java programs, and a coding in the theorem prover Isabelle [12].

References

- [1] B. Jacobs and E. Poll. Coalgebras and Monads in the Semantics of Java. *Theoret. Comput. Sci.*, 291:329–349, 2003.
- [2] E. Moggi. Notions of computation and monads. *Inform. and Comput.*, 93:55–92, 1991.
- [3] E. Moggi. A semantics for evaluation logic. *Fund. Inform.*, 22:117–152, 1995.
- [4] E Moggi. An abstract view of programming languages. Technical Report ECS-LFCS-90-113, Dept. of Computer Science, Edinburgh Univ., 90.
- [5] Till Mossakowski, Lutz Schröder, and Sergey Goncharov. Completeness of monad-based dynamic logic. Technical report, University of Bremen, 2006.
- [6] A. Pitts. Evaluation logic. In *Higher Order Workshop*, Workshops in Computing, pages 162–189. Springer, 1991.
- [7] L. Schröder and T. Mossakowski. Monad-independent Hoare logic in HASCASL. In *Fundamental Aspects of Software Engineering*, volume 2621 of *LNCS*, pages 261–277, 2003.
- [8] L. Schröder and T. Mossakowski. Monad-independent dynamic logic in HASCASL. *J. Logic Comput.*, 14:571–619, 2004.
- [9] Mark R. Shinwell and Andrew M. Pitts. On a monadic semantics for freshness. *Theoret. Comput. Sci.*, 342:28–55, 2005.

- [10] Philip Wadler. Comprehending monads. In *LFP '90: Proceedings of the 1990 ACM conference on LISP and functional programming*, pages 61–78, New York, NY, USA, 1990. ACM Press.
- [11] Philip Wadler. How to declare an imperative. *ACM Computing Surveys*, 29:240–263, 1997.
- [12] Dennis Walter. Monadic dynamic logic: Application and implementation. Master's thesis, University of Bremen, 2005.