A Substructural Modal Type Theory to handle Mobility Failures in Distributed Computing

Giuseppe Primiero

Department of Computer Science
Middlesex University
G.Primiero@mdx.ac.uk

Error detection and fault tolerance in software systems are well-studied notions since the 70's in the context of engineered systems for resource sharing, resource typing and accessibility [9, 5, 22, 6, 7]. In these contexts, error-resolution consists in affected service identification and restart.

From the programming viewpoint, functional languages without side-effects limit the damage caused by failing computations and restarting them can be performed without multiple updates. A typical example of the standard strategy of re-evaluation of referentially transparent expressions to implement fault-tolerance is Erlang [3, 2].

Formal systems to study validity and satisfiability under failure conditions are limited in number and approaches. Semantically, Propositional Dynamic Error Logic, an extension of Propositional Dynamic Logic PDL [11], offers a labelled transition system including an error state, but it does not provide recovery means. Syntactically, typing systems are more common for implementations in distributed programming where failures can occur: type-checking and inference mechanisms for data sharing, [15]; systems for sound computations with explicitly distributed data structures, [14]; session types to mimic and check broadcasting with and without end-to-end reliable communication, [13, 10]; modal types to model mobile computing, [18]. In general, all failure mechanisms in type systems are expressed as forms of abortion procedures, reflecting the kill-and-restart strategy. In a typed natural deduction system, for example, the standard rule for falsehood (formally a negation introduction, corresponding to contradiction elimination) can be formulated as an instruction to abort a process, see e.g. [20]:

$$\frac{\Delta; \Gamma \vdash m : \bot}{\Delta; \Gamma \vdash ABORT(m) : \alpha}$$

This rule preserves correctness in a normalisation process, but it forbids any attempt at identifying and resolving the source of error.

Failures of mobility are an interesting and important sub-class of the several typologies of errors occurring in mobile and distributed computing. They do not seem to require necessarily abortion processes, as errors of typing do. Previous works on constructive modalities [4, 1] with applications to type theories [21, 19] and distributed systems [16, 8, 17, 18, 12] offer a possile basis for a syntactic investigation of failures in mobile distributed computing. We present a substructural modal type theory to reason about mobility failures and their resolutions. Our strategy is given in three steps:

1. we formulate a language with indexed processes for valid code and values;
2. we enrich the language with the means for *mobility* of code and values via modal-style functions;
3. we consider functions for mobility failures and their resolution, and for unresolvable errors.

The substructural nature of the language highlights some important aspects of failure-prone mobile computing:

- failure of commutativity indicates that resource allocation is relevant in view of mobility rules;
- failure of weakening indicates that (sufficiently complete) resource redundancy at locations guarantees availability of a resolution strategy;
- finally, failure of contraction shows that location redundancy is not trivial.

We present first a simple explanatory example, introduce rules for failures of mobility with appropriate handling procedures, and a standard abort rule. We formulate meta-theoretical properties of error expressions, show local soundness and completeness and prove termination of resolution or failure.

References

1. N. Alechina, M. Mendler, V. de Paiva, and E. Ritter. Categorical and Kripke Semantics for Constructive S4 Modal Logic. In *Proceedings of the 15th International Workshop on Computer Science Logic*, volume 2142 of *Lecture Notes In Computer Science*, pages 292 – 307, 2001.
2. Joe Armstrong. *Making reliable distributed systems in the presence of software errors*. PhD thesis, Royal Institute of Technology, Stockholm, Sweden, 2003.
3. Joe Armstrong. Erlang. *Commun. ACM*, 53(9):68–75, September 2010.
4. G.M. Bierman and V. de Paiva. On an Intuitionistic Modal Logic. *Studia Logica*, (65):383–416, 2000.
5. F. Cristian. Exception Handling and Software Fault-Tolerance. *IEEE Transactions on Computers*, C-31:531–540, 1982.
6. F. Cristian. A rigorous Approach to Fault-Tolerant Programming. *IEEE Transactions on Software Engineering*, SE-11:23–31, 1985.
7. F. Cristian. Understanding Fault-Tolerant Systems. *Commun. ACM*, 34:56–78, 1991.
8. R. Davies and F. Pfenning. A modal analysis of staged computation. *Journal of the ACM*, 48(3):555–604, 2001.
9. J.B. Goodenough. Exception handling: issues and a proposed notation. *Commun. ACM*, 8:683–696, 1975.
10. Ramnas Gutkovas, Dimitrios Kouzapas, and Simon J. Gay. A session type system for unreliable broadcast communication.
11. David Harel, Dexter Kozen, and Jerzy Tiuryn. *Dynamic Logic*. MIT Press, Cambridge, MA, 2000.
12. L. Jia and D. Walker. Modal Proofs as Distributed Programs. In *Programming Languages and Systems, ESOP2004*, volume 2986 of *Lectures Notes in Computer Science*. Springer Verlag, 2004.

13. Dimitrios Kouzapas, Ramunas Gutkovas, and Simon J. Gay. Session types for broadcasting. In Alastair F. Donaldson and Vasco T. Vasconcelos, editors, *Proceedings 7th Workshop on Programming Language Approaches to Concurrency and Communication-cEntric Software, PLACES 2014, Grenoble, France, 12 April 2014.*, volume 155 of *EPTCS*, pages 25–31, 2014.
14. Ben Liblit and Alexander Aiken. Type systems for distributed data structures. In Mark N. Wegman and Thomas W. Reps, editors, *POPL 2000, Proceedings of the 27th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, Boston, Massachusetts, USA, January 19-21, 2000*, pages 199–213. ACM, 2000.
15. Ben Liblit, Alexander Aiken, and Katherine A. Yelick. Type systems for distributed data sharing. In Radhia Cousot, editor, *Static Analysis, 10th International Symposium, SAS 2003, San Diego, CA, USA, June 11-13, 2003, Proceedings*, volume 2694 of *Lecture Notes in Computer Science*, pages 273–294. Springer, 2003.
16. J. Moody. Modal logic as a basis for distributed computation. Technical Report CMU-CS-03-194, School of Computer Science, Carnegie-Mellon University, Pittsburgh, PA, USA, 2003.
17. T. Murphy. *Modal Types for Mobile Code*. PhD thesis, School of Computer Science, Carnegie Mellon University, 2008. CMU-CS-08-126.
18. T. Murphy, K. Crary, and R. Harper. *Type-Safe Distributed Programming with ML5*, volume 4912 of *Lectures Notes in Computer Science*, pages 108–123. Springer Verlag, 2008.
19. A. Nanevski, F. Pfenning, and B. Pientka. Contextual Modal Type Theory. *ACM Transactions on Computational Logic*, 9(3):1–48, 2008.
20. F. Pfenning. *Constructive Logic*. Carnegie Mellon University, draft of december 28, 2000 edition, December 2000.
21. F. Pfenning and R. Davies. A judgemental reconstruction of modal logic. *Mathematical Structures in Computer Science*, 11:511–540, 2001.
22. D.A. Rennels. Fault-Tolerant Computing - Concepts and Examples. *IEEE Transactions on Computers*, C-33:1116–1129, 1984.