the *Projections*, a part of the transformation metamodel, that accounts for cases where an xDSML concept is transformed into several MoC concepts. The former can be specified using any classical model transformation languages, while we have devised a dedicated metalanguage for the latter. Our contribution has been implemented in the GEMOC Studio, an Eclipse-based language workbench, and illustrated on fUML defined using an xDSML capturing the notions of threads with instructions. We have made available a video showing the execution of an example fUML Activity, and the sources for the two xDSMLs. By defining MoCs as concurrency-aware xDSMLs, we give them a systematic structure, enabling their use at the language-level for the modeling of other concurrency-aware xDSMLs. In particular, it enables the use of the MoC that is the best fit for the concurrency paradigm of the language being developed. It also eases the development of an xDSML, since the model-level application of the MoC is simply a model conforming to an xDSML, that can be executed, debugged and animated like a regular model. Moreover, different formal behavioral properties can be assessed on executable models depending on the MoC used by the language.

Although any concurrency-aware xDSML can be used as a MoC, the concurrency theory community has studied in details a large number of MoCs such as the Actor Model [5] or Petri nets [4]. We plan to provide reference implementations for these ones in a MoC standard library, including Event Structures to bootstrap our approach. Afterwards, existing tools around these formalisms, such as model-checking tools, can be integrated seamlessly in our approach. Still, the approach remains rooted in the seminal MoC (*i.e.*, Event Structures), so higher-order transformations could be used to verify domain properties using the underlying MoC, while translating their results back to the domain [30]. Finally, even though we have considered concurrency-aware xDSMLs as language models for the implementation of more efficient tools, we plan to study how code generation or scheduler synthesis could be used to generate more efficient implementations of concurrency-aware xDSMLs.

Acknowledgments

This work is partially supported by the ANR INS Project GEMOC (ANR-12-INSE-0011).

References

[1] B. Combemale, J. Deantoni, M. Vara Larsen, F. Mallet, O. Barais, B. Baudry, and R. France, "Reifying Concurrency for Executable Metamodeling," in *SLE'13*.

[2] F. Latombe, X. Crégut, J. Deantoni, M. Pantel, and B. Combemale, "Coping with Semantic Variation Points in Domain-Specific Modeling Languages," in *EXE 2015*. Ottawa, Canada: CEUR, 2015.

[3] F. Latombe, X. Crégut, B. Combemale, J. Deantoni, and M. Pantel, "Weaving Concurrency in eXecutable Domain-Specific Modeling Languages," in *8th ACM SIGPLAN International Conference on Software Language Engineering (SLE 2015)*, 2015. [Online]. Available: https://hal.inria.fr/hal-01185911

[4] T. Murata, "Petri nets: Properties, analysis and applications," *Proceedings of the IEEE*, vol. 77, no. 4, 1989.

[5] G. A. Agha, "Actors: A model of concurrent computation in distributed systems." DTIC Document, Tech. Rep., 1985.

[6] G. Winskel, "Event structures," in *Petri Nets: Applications and Relationships to Other Models of Concurrency*, ser. LNCS, 1987.

[7] V. Pratt, "Modeling concurrency with partial orders," *International Journal of Parallel Programming*, vol. 15, no. 1, pp. 33–71, 1986.

[8] J. Deantoni and F. Mallet, "ECL: the event constraint language, an extension of OCL with events," Inria, Tech. Rep., 2012.

[9] OMG, "fUML specification v1.1," 2013. [Online]. Available: http://www.omg.org/spec/FUML/

[10] G. D. Plotkin, "The origins of Structural Operational Semantics," *The Journal of Logic and Algebraic Programming*, 2004.

[11] S. Tasharofi, P. Dinges, and R. E. Johnson, "Why do scala developers mix the actor model with other concurrency models?" in *ECOOP 2013*. Springer, 2013.

[12] F. Mallet and R. De Simone, "Correctness Issues on MARTE/CCSL constraints," *Science of Computer Programming*, vol. 106, pp. 78–92, Aug. 2015. [Online]. Available: https://hal.inria.fr/hal-01257978

[13] J. Armstrong, R. Virding, C. Wikström, and M. Williams, *Concurrent programming in ERLANG*. Citeseer, 1993.

[14] M. Gupta, *Akka essentials*. Packt Publishing Ltd, 2012.

[15] DIVERSE-team, "Github for k3al," 2016. [Online]. Available: http://github.com/diverse-project/k3/

[16] J. Deantoni, P. Issa Diallo, C. Teodorov, J. Champeau, and B. Combemale, "Towards a Meta-Language for the Concurrency Concern in DSLs," in *DATE*, 2015.

[17] OMG, "QVT specification v1.2," 2015. [Online]. Available: http://www.omg.org/spec/QVT/

[18] Language Workbenches Challenge, "Comparing tools of the trade," 2014. [Online]. Available: http://www.languageworkbenches.net/

[19] M. Voelter and V. Pech, "Language modularity with the MPS language workbench," in *ICSE*. IEEE, 2012.

[20] S. Kelly, K. Lyytinen, M. Rossi, and J. P. Tolvanen, "MetaEdit+ at the age of 20," in *CAiSE*. Springer, 2013.

[21] T. Van Der Storm, "The Rascal Language Workbench," 2011.

[22] L. C. Kats and E. Visser, "The spoofax language workbench: rules for declarative specification of languages and ides," in *ACM Sigplan Notices*, 2010.

[23] T. Mayerhofer, P. Langer, M. Wimmer, and G. Kappel, "xMOF: Executable DSMLs based on fUML," in *SLE*, 2013.

[24] G. Rosu and T. F. Serbanuta, "K overview and simple case study," in *Proceedings of International K Workshop (K'11)*, 2014.

[25] M. Nielsen, "Models for concurrency," in *Mathematical Foundations of Computer Science*, 1991.

[26] E. Lee, A. Sangiovanni-Vincentelli *et al.*, "A framework for comparing models of computation," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 1998.

[27] M. Nielsen, V. Sassone, and G. Winskel, *Relationships between models of concurrency*. Springer, 1994.

[28] C. Ptolemaeus, *System Design, Modeling, and Simulation: Using Ptolemy II*. Ptolemy. org Berkeley, CA, USA, 2014.

[29] B. Selic, "An architectural pattern for real-time control software," in *Workshop on Frameworks and Architectures, PLoP Conference*, 1996.

[30] F. Zalila, X. Crégut, and M. Pantel, "A transformation-driven approach to automate feedback verification results," in *Model and Data Engineering*. Springer, 2013, pp. 266–277.