Explicating and Reasoning with Model Uncertainty

Marsha Chechik

University of Toronto
Toronto, Canada
chechik@cs.toronto.edu

Abstract

> *"The reality of today's software systems requires us to consider uncertainty as a first-class concern in the design, implementation, and deployment of those systems"*
> David Garlan [4].

Uncertainty has been studied in many software engineering contexts, such as self-adaptive systems [3], probabilistic systems [5], requirements engineering [10], risk management [6] and others. In this talk, I focus on the problem of *uncertainty that the modeler has about the different aspects of software*. Such uncertainty is (a) *reducible*, i.e., it concerns things that are not inherently unknowable, and (b) *epistemic*, i.e., it is caused by a particular stakeholder's lack of knowledge, as opposed to being a property of the world.

Model uncertainty can be introduced into the modeling process in many ways: alternative ways to fix model inconsistencies [9,2,12], different design alternatives [13,8], modeler's knowledge about the problem domain [14], multiple stakeholder opinions [11], etc. Instead of waiting until uncertainty is resolved or forcing premature design decisions, we propose to defer the resolution of uncertainty for as long as necessary, while supporting a variety of transformation and reasoning operations that allow modelers to "live" with this uncertainty. In this talk, I survey some of our recent work on creating, transforming, and reasoning with models containing uncertainty. I also discuss the relationship between our treatment of model uncertainty and the popular alternatives: underspecification and non-determinism (and their close relatives, "I don't know" and "I don't care").

Our specification of models with uncertainty implicitly encodes a set of alternative possible models, where we are not sure which is the correct one. This notion has been introduced in behavioural modeling [7], but we expanded it to arbitrary modeling languages. Thus, such models with uncertainty can be thought of as "plural". Interestingly, plural models can capture a variety of other SE concepts: products in a product line, models adhering to a metamodel, member models in a megamodel [1]. I further describe how this analogy enables us to lift our uncertainty results to these domains.

Acknowledgements

This is joint work with members of the Modeling Group at the University of Toronto and specifically, with Michalis Famelis, Rick Salay, Alessio DiSandro.

References

1. J. Bezivin, F. Jouault, and P. Valduriez. On the Need for Megamodels. In *Proc. of OOPSLA'04 Workshop on Best Practices for Model-Driven Soft. Development*, 2004.
2. A. Egyed, E. Letier, and A. Finkelstein. Generating and Evaluating Choices for Fixing Inconsistencies in UML Design Models. In *Proc. of ASE'08*, pages 99–108, 2008.
3. N. Esfahani, E. Kouroshfar, and S. Malek. Taming Uncertainty in Self-Adaptive Software. In *Proc. of ESEC/FSE'11*, pages 234–244, 2011.
4. D. Garlan. Software Engineering in an Uncertain World. In *Proc. of FoSER'10*, pages 125–128, 2010.
5. A. Hinton, M. Kwiatkowska, G. Norman, and D. Parker. PRISM: A Tool for Automatic Verification of Probabilistic Systems. In *Proc. of TACAS'06*, volume 3920 of *LNCS*, pages 441–444. 2006.
6. S. Islam and S. Houmb. Integrating Risk Management Activities into Requirements Engineering. In *Proc. of RCIS'10*, pages 299–310, 2010.
7. K. G. Larsen and B. Thomsen. A Modal Process Logic. In *Proc. of LICS'88*, pages 203–210, 1988.
8. A. Mashiyat, M. Famelis, R. Salay, and M. Chechik. Using Developer Converstions to Resolve Uncertainty in Software Development: A Position Paper. In *Proc. of ICSE'14 Workshop on Recommendation Systems for Soft. Eng.*, 2014.
9. C. Nentwich, W. Emmerich, and A. Finkelstein. Consistency Management with Repair Actions. In *Proc. of ICSE'03*, pages 455–464, 2003.
10. J. Noppen, P. van den Broek, and M. Aksit. Software Development with Imperfect Information. *J. Soft Computing*, 12(1):3–28, Aug 2007.
11. M. Sabetzadeh, S. Nejati, M. Chechik, and S. Easterbrook. Reasoning about Consistency in Model Merging. In *Proc. of LWI'10*, 2010.
12. R. Van Der Straeten, J. Pinna Puissant, and T. Mens. Assessing the Kodkod Model Finder for Resolving Model Inconsistencies. In *Proc. of ECMFA'11*, pages 69–84, 2011.
13. A. van Lamsweerde. *Requirements Engineering - From System Goals to UML Models to Software Specifications.* Wiley, 2009.
14. H. Ziv, D.J. Richardson, and R. Klösch. The Uncertainty Principle in Software Engineering, 1996.