Risk-Driven Engineering of Requirements for Dependable Systems

Axel van Lamsweerde
Université Catholique de Louvain, Belgium

Getting the right software requirements under the right environment assumptions is a critical precondition for developing the right software. This task is intrinsically difficult. We need to produce a complete, adequate, consistent, and well-structured set of measurable requirements and assumptions from incomplete, imprecise, and sparse material originating from multiple, often conflicting sources. The system to be considered comprises software and environment components such as people and devices.

Requirements completeness in particular is among the most critical and difficult challenges. Missing requirements and assumptions are known to be the major cause of software failure. They often result from a lack of anticipation of unexpected conditions under which the software should behave adequately. A natural inclination to conceive over-ideal systems prevents adverse conditions from being properly identified and, when likely and critical, resolved through appropriate countermeasures. Risk analysis should thus be at the heart of the RE process.

The course will present a systematic approach for integrating risk analysis in model-based RE. The models we will consider inter-relate the goals to be achieved by the system, the concepts involved in their specification, the agents responsible for the goals, the operations needed to ensure them, and the agent behaviors required to meet them. To enable formal reasoning about goals when and where needed, a realtime linear temporal logic will be used to specify them.

In such models, obstacles are introduced as a natural abstraction for risk analysis. An *obstacle* to a goal is a precondition for the non-satisfaction of this goal. Obstacle analysis consists of (a) *identifying* as many obstacles as possible to every leaf goal in the goal refinement graph from relevant domain properties; (b) *assessing* the likelihood and severity of each obstacle; and (c) *resolving* likely and critical obstacles by integrating appropriate countermeasures in the goal model.

The course will detail and illustrate a variety of techniques supporting the *identify-assess-resolve* cycles of obstacle analysis.

- For obstacle *identification*, we may use a formal regression calculus to derive obstacles, or instantiate goal obstruction patterns to shortcut such derivations, or combine model checking and inductive learning to generate a domain-complete set of obstacles.
- For obstacle *assessment*, we need to enrich our models with a probabilistic layer allowing us to estimate the likelihood of fine-grained obstacles and propagate these through the obstacle and goal models in order to determine the severity and likelihood of obstacle consequences.
- For obstacle *resolution*, we may explore alternative countermeasures by use of systematic model transformations that encode risk-reduction tactics, and select most appropriate ones based on the likelihood of obstacles and the severity of their consequences.

References

[1] A. van Lamsweerde. *Requirements Engineering: From System Goals to UML Models to Software Specifications*. Wiley; 2009.

[2] A. van Lamsweerde, E. Letier. *Handling Obstacles in Goal-Oriented Requirements Engineering*. IEEE Transactions on Software Engineering, Vol. 26(10), pp. 978-1005; 2000.

[3] A. van Lamsweerde. *Elaborating Security Requirements by Construction of Intentional Anti-Models*. Proc. of the ICSE04, 26th Intl. Conf. on Software Engineering, ACM-IEEE, pp. 148-157; 2004.

[4] D. Alrajeh, J. Kramer, A. van Lamsweerde, A. Russo, S. Uchitel. *Generating Obstacle Conditions for Requirements Completeness*. Proc. of the ICSE'2012: 34th Intl. Conf. on Software Engineering; 2012.