Synthesis of executable implementations from scenario-based specifications, Dirk Fahland

Scenario-based modeling has emerged as an accepted paradigm for developing complex systems of various kinds. Its main purpose is to ensure that a system provides desired behavior to its users. A scenario is generally understood as a behavioral requirement, denoting a course of actions that shall occur in the system. A typical notation of scenarios is Message Sequence Charts or, more general, finite partial orders of actions. A specification is a set of scenarios. Intuitively, a system implements a specification if all scenarios of the specification occur in the system.

The main challenge in this approach is to systematically synthesize a state-based implementation from a given scenario-based specification; preferably to be achieved automatically. A further challenge is to analyze scenarios to avoid erroneous specifications. Existing scenario-based techniques exhibit a conceptual and formal gap between the paradigms of scenario-based specification and state-based implementation. This gap often renders synthesis surprisingly complex, and obscures the relationship between a specification and its implementation. Additionally, existing techniques for analyzing models from one paradigm cannot immediately be re-used in another paradigm.

In this thesis, we introduce a semantic model for scenarios that seamlessly integrates scenario-based specifications and state-based implementations. We focus on modeling and analyzing the control-flow of systems. Technically, we use Petri nets together with the well established notion of distributed runs for describing the semantics of scenarios, for systematically constructing and analyzing a specification, and for synthesizing an implementation from a given specification.

The contributions of this thesis follow from a few observations on how a system implements a specification. We derive two fundamental semantic concepts for scenario-based specifications. Firstly, each scenario has a history describing when and where the scenario occurs. Secondly, scenarios necessarily require progress for specifying behavior of distributed systems; existing semantic models for scenarios cannot express progress in a natural way.

Our first contribution is to combine history and progress as the only semantic concepts in a (novel) semantic model for scenarios, called ocllets. We derive an ocllet-based, but nevertheless classical declarative semantics for scenario-based specifications which defines when a given set of runs satisfies a given specification. We then consider minimal satisfying sets of runs (possibly containing infinitely many infinite runs), and characterize specifications which can be implemented as a distributed system. We thereby show that our semantics is compositional: a set of runs satisfies a composition of two specifications iff it satisfies each of the specifications.

The second contribution of this thesis closes the conceptual and methodological gap between scenario-based specifications and state-based system models. In our approach, the semantics of scenarios and the semantics of systems use the same concepts from Petri nets. We obtain an operational semantics for scenario-based specifications. This semantics constructs the minimal set of runs that satisfies a specification. By our operational semantics, a set of scenarios turns into an executable system. On the basis of this operational semantics, we consider the problems of analyzing behavioral properties of specifications and of synthesizing an implementation. We show that these problems are undecidable in general and we present a sufficient property for the decidable case. We then present algorithms for analysis and synthesis. We derive theses results by generalizing existing techniques from Petri nets to the domain of scenario-based specifications. Among others, we present an algorithm for deciding on the syntactic (static) level whether a Petri net implements a scenario-based specification.

The third contribution of this thesis addresses the problem of introducing new means of expression into a scenario-based technique. Each such expression contributes to the technique’s semantic model. To preserve analysis and synthesis results likewise as far as possible, new means of expression must be introduced systematically. We present a systematic approach for extending our semantic model for scenarios. By the example of anti-scenarios and imperative scenarios we address all aspects regarding behavior, analysis, and synthesis. We thereby derive a few simple operations and relations on scenarios. With these, we identify cases where analysis and synthesis results can fully be preserved, adapt existing algorithms where this is possible, or show why distinguished problems turn out undecidable.

Finally, we demonstrate the feasibility of our approach. Firstly, we report on modeling a workflow from disaster management with our approach. Secondly, we present a case study on analyzing industrial process models with our algorithms. We implemented our algorithms for simulating and analyzing scenario-based specifications in our tool Greta.