Synthesizing and Utilizing Partial Behavior Models During Requirements Elicitation

Ivo Krka
Department of Computer Science
University of Southern California
Los Angeles, CA 90089-07081, USA
krka@usc.edu

Categories and Subject Descriptors

D.2 [**Software Engineering**]: Requirements/Specifications; D.2 [**Software Engineering**]: Software Architectures

General Terms

Design

Keywords

behavior model synthesis, requirements elicitation

1. RESEARCH PROBLEM

During a software system's requirements elicitation, the system stakeholders provide a number of informal statements about the desirable features of the system. The requirements analysts then map these statements into more formal yet straightforward and intuitive specifications, such as system *use case scenarios* and *properties*. Such specifications have proven useful for different developmental activities, including communicating intent among stakeholders, capturing different viewpoints on the system, and directly documenting the requirements. However, scenarios and properties provide only a partial description of the system-to-be. For example, a use-case scenario usually describes one desirable sequence of system events. Similarly, system properties express the high-level *goals* of the system and *constraints* on system operations. I posit that *mapping partial specifications to more comprehensive models would be beneficial for gradual refinement of requirements, and exploration of the system solution space*.

State machine-based models of software behavior, such as UML statecharts and Labeled Transition Systems, are leveraged for different tasks during software development [7]. For example, these models facilitate various analyses (e.g., model checking), they form a basis for system simulation and code generation, and are often leveraged as a "blueprint" for the subsequent implementation. Consequently, researchers have proposed numerous techniques for synthesizing behavioral models from requirements (e.g., [2, 12]). Most of these techniques require scenarios as inputs and return final behavioral models, thus overlooking the inherent partiality of the input specifications [9]. Similarly, some recent approaches take properties as inputs and synthesize models which do not violate the properties [1, 3]. Although these property-based approaches build less restrictive models, they primarily assure the satisfaction of the properties without considering whether the specification should be further refined, and which behaviors should be implemented.

Modal Transitions Systems (MTS), a partial behavior modeling formalism, have recently been presented as a promising alternative for capturing both scenarios and properties in a single behavioral model [10]. Scenarios are incorporated into an MTS as *required* behavior, while the behavior that is not proscribed by the properties is captured as *potential* behavior. Additionally, two algorithms for MTS *merging* have been proposed [4, 8], thus making MTSs appropriate for specifying and merging stakeholders' viewpoints. The existing MTS synthesis approach, however, produces system-level, as opposed to component-level, models, hence suffering from potential scalability issues and inability to expose certain specification discrepancies (see [6] for details). Moreover, rigorous methods for elicitation of new requirements based on the obtained partial models have not been proposed.

In this thesis, I will provide *a holistic approach for capturing, analyzing, and refining system requirements by synthesizing, analyzing, and exploring component-level MTS models*. First, I will design a novel synthesis algorithm that creates component-level MTSs from available requirements specifications. Second, I will explore how analysis of the synthesized MTSs, as well as their composition, can help to detect previously undiscovered specification flaws. Third, I will explore how the produced MTSs can be utilized for elicitation of new requirements. The following section provides more detail on the proposed approach, while I present the immediate thesis research in Section 3.

2. PROPOSED APPROACH

In this section, I will illustrate the envisioned contributions of this thesis. The intended approach, depicted in Figure 1, will account for a rich set of requirements specifications, including positive and negative scenarios, goal models and operational pre- and post-condition constraints on system operations. The assumptions here are twofold: (1) the scenarios are given in terms of event sequences, and (2) goals and constraints are given as logical formulas on system state variables. Neither should be overly restrictive as these types of specifications are widely used [11].

The first contribution of this thesis will be an algorithm for synthesizing MTS models of system components (*task 1* in Figure 1). I choose to create component-level models as they will capture the properties arising from the system decomposition more precisely than the system-level models produced by existing techniques. Intuitively, the resulting MTSs comprise required behavior from the positive scenarios and potential behavior which does not conflict with the negative scenarios, goals, and operational constraints. The algorithm is intended to first create the most general MTS which captures the behavior allowed by the goals and operational constraints. Next, the behavior from the negative scenarios is removed and the resulting MTS is refined with the behavior required in the
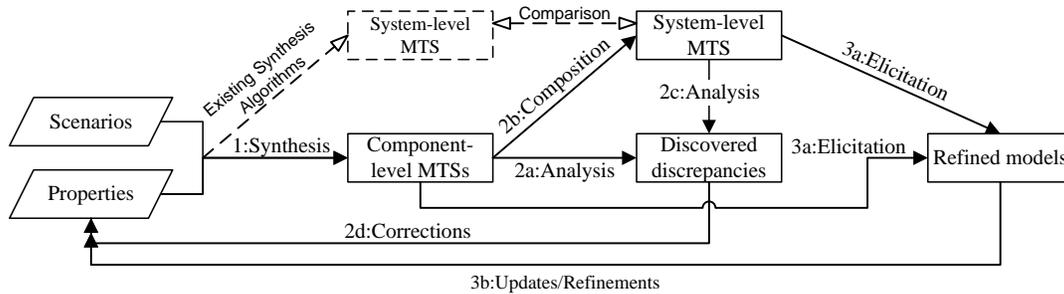
Figure 1: The proposed approach

positive scenarios. To evaluate the algorithm, I will analyze its worst-case as well as practical complexity. Additionally, I will assess its correctness and provide corresponding formal proofs.

As part of my thesis, I will also identify and classify the types of flaws in input specifications that can be discovered through analysis of component MTSs (*task 2a*). Furthermore, I will define guidelines and promising actions for solving these flaws (*task 2d*). For example, analysis of the system goals' effects can discover both (1) overly constraining goals (e.g., an event can never execute because of a goal) and (2) unnecessary goals (e.g., a goal does not influence the system behavior). I will also explore the types of flaws that can be discovered when the component MTSs are composed together (*tasks 2b* and *2c*). Examples of these flaws are undesirable implied scenarios due to imprecise properties, and positive scenarios that cannot execute. I expect that these analyses will provide crucial benefits during requirements elicitation, since propagation of such flaws to later development stages can result in significant expenses.

The next contribution of my thesis will be an iterative technique for eliciting new and refining old requirements (*tasks 3a* and *3b*). The synthesized MTSs explicitly capture the behavior that is neither required by the positive scenarios nor prohibited by the negative scenarios, goals, or operational constraints in terms of potential transitions. Manual inspection and per case decisions whether a potential transition should be removed or become required is a tedious and error-prone process. Therefore, I will investigate ways of guiding the refinement process with smartly focused model exploration and targeted queries which can significantly reduce the number of potential transitions. The main idea is to derive prospective goal models and operational constraints based only on either the currently required behavior. The analyst would then be queried about the proposed specification refinement. In case of a positive answer, potential behavior which violates the refined specification can be eliminated. When the answer is negative or undecided, the potential behavior which violates the refined specification would be displayed as a scenario which should be classified either as positive or negative. Similarly, if a system goal is not reached by the required behavior, the analyst would be offered ways of accomplishing the goal through currently potential transitions.

To initially evaluate the main concepts in my thesis, I intend to utilize commonly used requirements exemplars to assess the correctness and to compare the proposed approach with the existing state-of-the-art (represented with dashed lines in Figure 1). The main part of the evaluation will be concerned with applying the different facets of the approach to realistic, real-world systems. More specifically, I plan to assess the proposed corresponding implementation's scalability, the numbers of uncovered flaws and false positives, and the burden imposed on the requirements analyst.

3. CURRENT STATUS

In an early paper about my ongoing work [5], I described the high-level steps of the component-level MTS synthesis algorithm (*task 1* in Figure 1). In that paper, I also outlined promising ways of utilizing component MTSs, one of which will be a major part of my thesis: requirements elicitation.

I have more precisely specified and implemented the synthesis algorithm (*task 1*), and analyzed its correctness in a subsequent publication [6]. The inputs in the current version of the algorithm are positive scenarios and operational pre- and post- constraints. The algorithm first produces component-level constraints from the system-level constraints that are leveraged to construct the most general component MTSs. The MTSs are then refined with the behavior from positive scenarios. To guide the process, each scenario is annotated with inferred conditions that must hold at particular points of the scenario execution. Finally, I have identified several requirements discrepancies and potential design flaws that can be discovered (*task 2a*), and proposed potential solutions (*task 2d*).

4. REFERENCES

[1] D. Alarjeh et al. Learning operational requirements from goal models. In *Proc. of ICSE*, 2009.

[2] C. Damas et al. Scenarios, goals, and state machines: a win-win partnership for model synthesis. In *Proc. of FSE*, pages 197–207, 2006.

[3] G. de Caso et al. Validation of contracts using enabledness preserving finite state abstractions. In *Proc. of ICSE*, 2009.

[4] D. Fischbein and S. Uchitel. On consistency and merge of modal transition systems. In *Proc. of FSE*, 2008.

[5] I. Krka et al. From system specification to component behavioral models. In *Proc. ICSE NIER*, 2009.

[6] I. Krka et al. Synthesizing partial component-level behavior models from system specifications. In *Proc. of ESEC/FSE*, 2009.

[7] J. Magee and J. Kramer. *Concurrency: State Models and Java Programming*. John Wiley & Sons, 2006.

[8] S. Uchitel and M. Chechik. Merging partial behavioural models. 2004.

[9] S. Uchitel et al. Behaviour model elaboration using partial labelled transition systems. In *Proc. of ESEC/FSE*, 2003.

[10] S. Uchitel et al. Behaviour model synthesis from properties and scenarios. In *Proc. of ICSE*, 2007.

[11] A. van Lamsweerde. *Requirements Engineering: From System Goals to UML Models to Software Specifications*. John Wiley & Sons, 2009.

[12] J. Whittle and J. Schumann. Generating statechart designs from scenarios. In *Proc. of ICSE*, 2000.