Scalable Component-based Model Revision with Separation of Concerns for Cyber-Physical systems.

1 Introduction

Cyber-physical systems integrate the computational aspects with physical aspects. Such systems are deployed in the context of life critical systems such as a smart cruise controller, a fuel injection controller and an airbag controller in an automobile, an altitude switch controller and a navigation system controller in an airplane, electronic toys, household devices such as dishwashers and microwaves, communication devices such as routers, and medical devices. Assurance of such systems is very important. For example, cyber-physical systems in an airplane and automobile are mission critical. Failure of controllers in medical devices can be life threatening. Techniques based on model-checking have been used extensively in providing assurance about these systems. Approaches based on model checking generally deal with the design of the system, i.e., they attempt to ensure that the system developed satisfies the current specification.

In many scenarios, however, it is often necessary to revise an existing system due to changes in requirements and/or environment. Moreover, in this case, it is necessary that no new bugs be introduced, i.e., it is required that the existing specification continues to be satisfied. And, in such revision, the original specification may not be available especially if the designers of the original model are unavailable during the revision process.

The objective of our project is to utilize formal methods for gaining justifiable confidence in such maintenance, where an existing system is revised to satisfy new properties such as safety properties, liveness properties, fault-tolerance properties and timing constraints. Specifically, we focus on developing algorithms and tools to facilitate **Scalable** Component-based **Model Revision** with Separation of Concerns for cyber physical systems.

Model Revision. The problem of model revision begins with a model M and properties Σ and Π. It is given that M satisfies property Σ. But it does not satisfy property Π. The goal of model revision is to revise model M so that it satisfies Π without violating Σ. Moreover, it is expected that the revised model, say M', satisfies Σ in the same fashion as M did. In other words, the revision is performed only to add a new property Π and not to introduce new ways to satisfy Σ.

Thus, model revision is a compromise between two main approaches for providing assurance, *model checking* and *model synthesis*, with formal methods. Specifically, the model checking problem begins with a model M and a property Ψ. The goal of this problem is to determine whether M satisfies Ψ. Hence, given this input, the model checker either declares that M satisfies Ψ. Or, (typically), it declares failure and provides a counterexample demonstrating why Ψ is not satisfied. One cannot apply model checking to solve the problem of model revision since a model checker will simply declare that M does not satisfy Π without a significant help in how it could be satisfied.

Model synthesis begins with a property Ψ and synthesizes a model M such that M satisfies Ψ. One can envision applying model synthesis to solve model revision by providing the property $\Sigma \wedge \Pi$. However, with such an approach there is no guarantee of reusing the original model. In other words, the new model may satisfy Σ in a completely new way. This is undesirable if the original model was satisfying Σ in a fashion that made it easier to obtain efficiency. It is also undesirable if Σ is unknown (e.g., when the original model is the de-facto specification).

As one can expect, model synthesis provides a better assurance since it generates a model that is guaranteed to be *correct-by-construction*. However, it suffers from the shortcoming of (potentially) increased complexity and/or loss of reuse. Model revision overcomes these drawbacks. In particular, because it constructs a revised model that is *correct-by-construction*, it provides additional assurance compared with model checking. Also, in several instances, the complexity of model revision is closer to that of model checking. Additionally, model revision is especially useful when existing programs need to be revised due to bug fixes, revised requirements, etc. Moreover,

in this case, it is necessary that no new bugs be introduced, i.e., it is required that the existing specification continues to be satisfied. And, in such revision, the original specification may not be available especially if the designers of the original model are unavailable during the revision process.

Component-based Design. Component-based design is an approach that subdivides the given model into components that can be designed and developed independently. In the context of cyber-physical systems, we anticipate that some of the components would be realized using software (computational components) while some would be realized as physical components. In the original design of the system, one generally performs a tradeoff between achieving functionality in these two types of components. Our approach for model revision, however, emphasizes mainly on the revision of computational components leaving the physical components unchanged. This is due to the fact that we expect that it would be significantly harder to change physical units. Only when it is impossible to perform the required revision with computational components alone, we will identify the changes that need to be satisfied by revised physical components. Some of these changes could be performed using automation although for some changes manual revision may be required.

Separation of Concerns. System requirements can often be thought in terms of functional and non-functional requirements. Intuitively, the former define what the system is expected to accomplish. Examples include calculations, data manipulation, etc. And, the latter include constraints on the designer such as fault-tolerance and performance requirements. While component-based design assists in separating concerns of functionality, it introduces another problem, namely the need dealing with non-functional properties that are often cross-cutting. Therefore, to apply model revision in this context, it is important to develop new algorithms that will still allow us to modify individual components in a systematic way while providing the assurance.

Scalability. Scalability in the context of model revision refers to feasibility of revising complex/large models. In this project, we intend to pursue four strategies to improve scalability of algorithms to perform model revision. The first strategy will utilize the component based structure so that revision could be achieved without performing state space exploration on the entire composed system. The second strategy will utilize the theory of detectors and correctors that are necessary and sufficient for designing a large class of fault-tolerant programs. This will allow revision algorithm to compute the *specification* of the desired component rather than its detailed implementation. Since specification of desired components can be computed by techniques from symbolic model checking, it would reduce the time in adding fault-tolerance. The third strategy will utilize complexity analysis to identify bottlenecks during the revision algorithm. We have shown that these bottlenecks can be effectively addressed with Binary Decision Diagrams (BDDs) in the context of revising distributed programs. We have also evaluated the role of SMT solvers in the context of verification of stabilizing algorithms. We intend to utilize these strategies in improving scalability of the revision algorithms for cyber-physical systems. Finally, the last strategy will focus on reducing time complexity by using untimed models instead of models that use time explicitly. However, unlike the timed models, such models often require fairness to ensure that all components can execute eventually. We intend to develop techniques to manage the complexity raised by fairness constraints so that revision can be performed in time that is closer to revising untimed models without fairness.

2 Goals

The goal of this project is to (1) develop constraints that have to be satisfied during revision of models for cyber-physical systems so that the revised model can be realized in the given cyber-physical system, (2) develop algorithms that utilize component-based nature and symbolic techniques to efficiently revise models for cyber-physical system, (3) specialize those algorithms in the context of addition of fault-tolerance by using the theory of separation of concerns, and (4) develop techniques for effective management of fairness during revision so that revision could be performed on untimed models rather than timed ones.