Formal Specification and Verification of System and Software Models

Ferhat Erata
UNIT Information Technologies R&D Ltd.
(ferhat@computer.org)

The complexity of software systems in safety critical domains (e.g., avionics and automotive) has significantly increased over the years. To tackle with this complexity, development of such systems requires various phases which result in several artifacts that are increasingly represented as models (e.g., requirement models, architecture models, and design models). To achieve model correctness and thereby to increase software quality, formal verification of software models has become a promising approach and received a considerable amount of attention in the last decade. Different methods and tools exist in the literature that allow to specify models based on mathematical theories or formalisms and to reason about the correctness properties.

In this tutorial, I am going to introduce two mainstream formal specification and formal analysis techniques to ensure model correctness on an illustrative reactive system. The tutorial will be split into three parts. The first part will immediately introduce to the audience the basic mathematical notions required to model an abstraction of the example reactive system. The second part demonstrates how to formally specify the system using two different formalisms: first-order relational logic (FORL), an expressive logic with transitive closure supported by Alloy Analyzer and the logic of equality with uninterpreted functions (EUF), a first-order logic with many well-supported tools (e.g., SMT solvers). The final part presents finding correct instances and checking assertions written in these formalisms using Alloy Analyzer (https://github.com/AlloyTools/) and Z3 SMT Solver (https://github.com/Z3Prover/z3). All three parts are accompanied by live demonstrations.