BOMBARDIER
the evolution of mobility

Introduction of a formally proven safety violation check function

The demand in society for increased system safety requires that companies continually work on improving their products and processes to meet standards and regulations of national and international standardization bodies. When an increased effort in test and verification renders diminishing returns, a shift to alternative or complementary methods can prove a more efficient allocation of resources.

For an existing product, we have long wanted to introduce formal verification of the business logic, but earlier discussions on formal verification have resulted in the conclusion that since we have no formal specifications there is nothing to verify the implementation against. To develop full formal specifications for the current functionality would be prohibitively expensive, and, even if we did, the current implementation, being written in C, does not lend itself well to formal proof.

Therefore, we want to explore the possibility to formally prove only specific, safety relevant, properties. This could be accomplished through a separately specified and implemented module (Safety Violation Check Package). The SCVP module shall check only safety relevant invariants; invariants that are believed to hold as a result of the combined requirements on the product, but cannot be formally proven. This relatively small set of rules could be expressed more succinctly than the functional requirements, and they shall be implemented such that formal proof of correctness of the rules is possible.

The SCVP shall shut down the system in case of a violation of the rules but not otherwise affect the output. This allows the SCVP to be incrementally developed all the while keeping a fully functional system. Rules can be refined and added, or even removed, without affecting the functionality.

The following is a rough outline of the steps involved in the study:

- Perform a literature study on formal verification and safety supervision functions.
- Gain an understanding of the architecture and workings of the product.
- Propose an architecture that lets the SCVP monitor the state of the system at key moments, allowing it to evaluate the state and shut down if any violations are found.
- Evaluate and select a language for the implementation and tools for the formal proof.
- Develop a specification for the new module.
- Propose a set of rules for the proof of concept.
- Implement a proof of concept with a limited rule set.
- Evaluate the proof of concept.

The following questions should be answered in the study:

- What kind of properties can be proven to hold and to what extent?
- How natural is the integration with the current C-code?
    - What effort is needed to translate the state and present it to the SCVP?
    - How much will this interface code and translation weaken the safety arguments?
    - What is the impact of reusing functional code to access the state from the SCVP?
- What performance penalties can be expected?
    - What is the complexity of the algorithms?
    - Does the solution scale well for large states?
    - Perform empirical time measurements on the proof of concept.
- What are the probable effect on our current development process?
    - Do we need independence between the functional implementation and the SCVP implementation?
    - Do we need independent test and verification of the SCVP?

For more information, contact:

Bombardier Transportation - Rail Control Solutions
Magnus Adamsson
email: magnus.adamsson@rail.bombardier.com
tel: 073-448 08 00

Lund University, Department of Computer Science
Martin Höst
email: martin.host@cs.lth.se
tel: 046-222 90 16