In its quest for software perfection, the critical software industry has been progressively embracing the use of formal verification tools [static analyzers, program provers, model checkers] as a complement, or even as an alternative, to traditional software validation techniques based on testing and reviews. The usefulness of verification tools in the certification of critical software is, however, limited by the amount of trust one can have in their results. Two major risks exist: unsoundness of verification tools (failing to detect a misbehaving program) and miscompilation (post-verification introduction of bugs during the production of executable code). The Verasco project investigates a radical, mathematically-grounded solution to these issues: the formal verification of compilers and verification tools themselves.

TECHNOLOGICAL OR SCIENTIFIC INNOVATIONS

We set out to develop a generic static analyzer based on abstract interpretation for the C language, along with a number of advanced abstract domains and domain combination operators, and prove the soundness of this analyzer using the Coq proof assistant. Likewise, we continue our work on the CompCert C formally-verified compiler, the first realistic C compiler that has been mechanically proved to be free of miscompilation, and will carry it to the point where it could be used in the critical software industry. We also investigate the tool qualification issues that must be addressed before formally-verified tools can be used in the aircraft industry. Critical software deserves the highest-assurance development and verification tools that computer science can provide. By going all the way to a full formal verification of such tools, our work will generate unprecedented confidence in the results of source-level static analysis, therefore fully justifying its role in the development and certification of critical software.

STATUS - MAIN PROJECT OUTCOMES

The project started on Jan 1st, 2012.