The Gap between Theory and Practice:
A Challenge

Joan Krone, Denison University

Abstract—The RESOLVE project has made encouraging progress during the past three decades both in the area of developing mathematical support for specifying and verifying software and in the area of expanding a collection of formally specified data structures and implementations for them. However, we do not have examples of real world applications to use in our Software Engineering courses. The purpose of this session is to discuss the state of RESOLVE in the context of narrowing the gap between theory and practice.

Index Terms—formal specifications, software engineering, mathematical theories.

I. INTRODUCTION

The RESOLVE project has progressed from a pure research endeavor in the 1980’s to its current status which includes both serious research, particularly from the verification point of view, and significant contributions to a variety of courses in the computer science curriculum. There are faculty and graduate students working on mathematical type checking, verification condition (VC) generation, devising and applying proof rules to those VC’s, building a website which supports programming in RESOLVE, and writing specifications for RESOLVE components.

For the CS curriculum, tools have been developed to encourage formal reasoning at all stages. For example, beginning students can use tools that give them immediate feedback when they reason about what particular program portions should produce. As students continue through the curriculum, they learn to write loop invariants as well as pre and post conditions for their programs.

There is a substantial collection of data structure examples for students and lots of exercises that give students experience in putting together structures in a modular, cohesive fashion, building new ones from existing ones noting the importance of needing to see only the interfaces of each layer in order to build a new one using it.

THE CHALLENGE

For this workshop it will be interesting to discuss two education challenges, one that we appear almost ready for and another that will no doubt take considerable time to achieve.

Let’s first look at the challenge of providing students in an intermediate level data structures course the opportunity to carry out an assignment that captures the essential elements of RESOLVE—building a piece of software that is formally specified, modular, and verifiable. One such assignment that some faculty have used, though at this point, the RESOLVE compiler/verifier is not yet ready to support totally, is that of writing a Kruskal implementation for the Spanning Forest Template. This is an ideal way for students to build a RESOLVE program by putting together a Coalescable Equivalence Relation, a Prioritizer (that might even be built on top of a Spiral), to realize the Spanning Forest with a Kruskal realization.

At present, students can do most of the necessary steps for this assignment, but the verifier lacks some important math units and some rule applications that are not yet available in the compiler/verifier.

This assignment seems close to being ready, and the remaining steps for making the entire project possible, seem within our grasp.

However, the second challenge I want to propose is that of figuring out a “real world” application for a software engineering (SE) course. At Denison we cover many RESOLVE principles in the SE course, which is a junior/senior level course. Additionally, we carry out some real world project provided by an “industrial partner,” a company or entity who provides our students with requirements and who actually use the software produced by the students.

At Denison, the SE students do undertake a real world project, and they use RESOLVE principles to the extent possible. Our students have had many successful projects over the years in the sense that they met whatever requirements they needed to meet and the client is actually using their software. However, the requirements documents have all been informal, as well as the designs. The students have devised multiple test cases, but verification has remained a dream for the future.

Even a “contrived” project would be a great achievement, but at this time, we do not have such an assignment. There remains a gap between theory and practice. Questions we need to address include: How wide is this gap? What might some possibilities be? How close can we come to a real project?

In 2003 we saw the verifying compiler challenge. At the RESOLVE 2016 workshop, we face the challenge of devising a real world project.