

DISSERTATION PROPOSAL

BENJAMIN BERMAN

1. INTRODUCTION

A long time ago my cello teacher pointed out that the way to play a difficult passage of music is not simply to grit one's teeth and keep practicing but to figure out how to make playing those notes easy. The major goal of the proposed dissertation is to reapply the same idea in the context of theorem proving with the Coq proof assistant: I intend to show ways to make the difficult task of using Coq easier by improving the user interface. As well as solving usability problems for an important and powerful tool, many of the techniques I am developing and testing are widely applicable to other forms of coding.

The research involved in this proposed dissertation, described more fully below, breaks down into two related main parts. Part one is the development of “CoqEdit”, a new theorem proving environment for Coq, based on the jEdit text editor. CoqEdit will mimic the main features of the existing environments for Coq, but will have the important property of being easily extended using Java. Part two is the development and testing of several such extensions.

There are several points that I hope will become clear as I describe the research for the proposed dissertation below. First is that the research will make a significant positive contribution to society. While Coq is a powerful tool, and is already being used for important work, its power has come at the cost of complexity, which makes the tool difficult to learn and use. Finding and implementing better ways to deal with this complexity in the user interface of the tool can allow more users to perform a greater number, and a greater variety, of tasks. At a more general level, the research contributes to a small but growing literature on user interfaces for proof assistants. The work this literature represents can be viewed as an extension of work on proof assistants, which in turn can be viewed as an extension of work on symbolic logic: symbolic logic aims to make working with statements easier, proof assistants aim to make working with symbolic logic easier, and user interfaces for proof assistants aim to make working with proof assistants easier. These all are part of the (positive, I hope we can assume) academic effort to improve argumentative clarity and factual certainty.

In addition, at a somewhat different general level, the research will contribute to our notions of how user interfaces can help people write code with a computer. The complexities of the tool in fact help make it suitable for such research, since a) they are partly the result of the variety of features of the tool and tasks for which the tool may be used (each of which provides an opportunity for design) and b) the difficulties caused by the complexity may make the effects of good user interface design more apparent. Furthermore, although Coq

has properties that make it very appealing for developing programs (in particular, programs that are free of bugs), it also pushes at the boundaries of languages that programmers may consider practical for the time-constrained software development of the “real world”. However, if, as in the proposed research, we design user interfaces that address the specific problems associated with using a language, perhaps making the user interface as integral to using the language as its syntax, these boundaries may shift outward. This means that not only are we improving the usability of languages in which people already are coding, we are also expanding the range of languages in which coding is actually possible.

The second point I hope will become clear is that this research will be an intellectual contribution, i.e. that the project requires some hard original thinking. User interface development is sometimes “just” a matter of selecting some buttons and other widgets, laying them out in a window, and connecting them to code from the back end. While this sort of work can actually be somewhat challenging to do right (just one hurdle is that testing is impossible to fully automate), the project goes well beyond this by identifying specific problems, inventing novel solutions, and testing these solutions in studies with human subjects.

–more than just widgets –new widgets –challenge of understanding the system –challenge of finding things that could be improved upon –novel ideas –clarification of existing ideas

In writing this proposal, I seek to make clear several points.

Why is this work important to society?

Why is this work an intellectual contribution?

Why is this work doable?

2. IDEAS

2.1. jEdit as an Interactive Theorem Proving Environment. Harness the power of Java/Piccolo/jEdit. jEdit also used with Isabelle (CITE).

2.2. Proof Previews.

2.3. Proof Tree Visualization.

2.4. Proof Transitions.

2.5. Syntax Tree Visualization.

2.6. Keyboard-Card Menus.

3. IMPLEMENTATION

3.1. Current.

3.2. Planned.

4. DESCRIPTION OF USER STUDIES

4.1. Keyboard-Card Menu Study Results.

5. LITERATURE REVIEW

6. TIMELINE

7. CONCLUSION

I hope to have made several points in this proposal. First, that this is important work, both because the Coq interactive theorem prover is an important tool that could benefit significantly from improved user interfaces and because many of the ideas generalize to other forms of coding. Second, that as an intellectual challenge this work is non-trivial, not only because of the normal programming problems that must be overcome but because designing good user interfaces for complicated systems, which includes the identification of tractable problems and the testing of potential solutions, is non-trivial. Finally, that, despite this non-trivial nature, the work can be accomplished.