Real-world formal documentation

Thomas Tuerk

Independent Scholar

13 April, 2018

Formal methods are great!

- complexity of hard- and software is ever increasing
- we rely on these systems more and more
- formal methods play a vital part in dealing with this
- amazing feats have been achieved in recent years
  - very trustworthy formal models of hardware, protocols and programming languages
  - verified and verifying compilers of real-world languages
  - verified operating systems
  - . . .
- I'm especially interested in *interactive theorem proving* (ITP)
  - I want to harden soft- and hardware, i. e. find and fix bugs
  - proving properties of model is useful to find bugs

Formal methods are hardly used

- formal methods tools (including interactive theorem provers) are powerful, mature and very useful
- however, they are hardly used in practise
- this especially holds if non-trivial user-input is required
- even critical, well founded projects hardly use ITP

Formal methods are hardly used

- formal methods tools (including interactive theorem provers) are powerful, mature and very useful
- however, they are hardly used in practise
- this especially holds if non-trivial user-input is required
- even critical, well founded projects hardly use ITP

Why ?

Issues with ITP (and partly formal methods in general)

- intrinsic complexity of logic, proofs, . . .
- formal methods experts are needed
- it takes a lot of time, huge initial investment
- progress and benefits are hard to measure

Issues with ITP (and partly formal methods in general)

- intrinsic complexity of logic, proofs, ...
- formal methods experts are needed
- it takes a lot of time, huge initial investment
- progress and benefits are hard to measure

Well, there is something to all these points, but ...

Biggest problem: Prejudice

Even very skilled, clever developers often consider ITP (and formal methods in general) as a form of black magic:

- way to complicated for mere mortals
- huge gains are luring
- but you need to sell your soul to get them

Biggest problem: Prejudice

Even very skilled, clever developers often consider ITP (and formal methods in general) as a form of black magic:

- way to complicated for mere mortals
- huge gains are luring
- but you need to sell your soul to get them

Don't tell people that they are using formal methods. Then they are happy to do it.

Mitigating Issues

- intrinsic complexity of logic, proofs, . . .
  most bugs are found in practice by testing, writing formal specifications and formal sanity checks, not by deep proofs
- formal methods experts are needed
  experts only needed for deep proofs,
  many tasks can be done by programmers
- it takes a lot of time, huge initial investment
  synergies with documentation and testing tasks safes time
- progress and benefits are hard to measure
  yes, but measures can be invented, good tool support needed

Good tool support vital.

Advanced Documentation and Testing Tool (ADATT)

- I recently started developing a tool called ADATT
- disguised as a markup + functional programming language
- inspired by Lem
- provide "compiler" + common programming language tools
- compilation to
  - high quality human readable documentation
  - executable specification in common programming languages
  - specification for common theorem provers
- ease of usage important for acceptance
  - good IDE integration
  - good error messages
  - ...

Advanced Documentation and Testing Tool (ADATT) II

- workflow
  - start with completely informal, natural language documentation
  - incrementally add formal content
    (e. g. type signatures or test cases)
  - immediate benefits of adding more formal content
  - ultimate goal: complete, executable formal specification
- good support for
  - partial specifications
  - testing
  - statistics and simple progress measures
- helps communication between software engineers, test engineers and formal method engineers
- nothing fancy like natural language processing involved

Advanced Documentation and Testing Tool (ADATT) III

- ADATT is still in very early stages
- no working prototype yet
- however, early feedback is very welcome