Language-Based Verification Will Change the World

Tim Sheard[1] Aaron Stump[2] Stephanie Weirich[3]

[1]Computer Science
Portland State University

[2]Computer Science
The University of Iowa

[3]Computer Science
University of Pennsylvania

U.S. National Science Foundation grant: 0910510

Verification is Powerful!

Verification is Powerful!

- Amazing *tour de force* examples:
  - Formally verified compilers [Leroy '06].
  - Operating systems [Klein et al. '09].
  - Relational database management systems [Malecha et al. '10].
- Full power of higher-order logic, type theory.
  - Very expressive logical languages.
  - Sophisticated theorem-proving environments (COQ, ISABELLE, etc.).

Verification is Hard.

Verification is Hard.

SEL4: Formal Verification of an OS Kernel [Klein et al '09]

- Verified that microkernel refines abstract spec of OS.
- Microkernel: 8700 lines C, 600 assembly.
- Proof: 200,000 lines ISABELLE.
- We estimate: 1 line proof $\approx$ 10 lines of C.
- So equiv. to around **2 million lines** of C.
- Best paper SOSP 2009.
- True TDF verification.

Verification is Irrelevant.

Verification is Irrelevant.

- Amazing things possible.
- Needed for niche applications (e.g., safety-critical).
- But just too costly for mainstream.

But

But

What is Verification?

But

What is Verification?

- TDF verification, certainly.
- But also type checking, static analysis:
  - data structure invariants.
  - path properties (e.g., adherence to library protocols).
  - timing correctness (WCET).
  - information-flow (security).
- JAVA, C# programmers use verification every day!
- This is the way forward.

Language-Based Verification

- Static typing provides light-weight verification.
- Scale up with more expressive types.
- *Dependent types*:

```
[ "Santa" , "Fe" , "NM" ] : list string 3

nil : list 'a 0
cons : 'a -> list 'a 'n -> list 'a ('n+1)

append : list 'a 'n -> list 'a 'm -> list 'a ('n+'m)
```

- Continuum from very light properties to deep ones.
- Incremental verification, pay as you go.
- Familiar language, toolset.
- Catch errors very early in development.

Verification is Everywhere.

- Type checking is verification.
- More advanced typing on the way.
- Coming from HASKELL, SCALA to JAVA, C#.
- Dependent types next step in evolution (COQ, AGDA, **Trellys**).
- A true change in the nature of programming.