The Science of Deep Specification

Andrew W. Appel, **Benjamin Pierce**, Stephanie Weirich,
Steve Zdancewic, Zhong Shao, Adam Chlipala

Princeton

Penn

Yale

MIT

A high-value "niche"

Zero-vulnerability critical software

- Compilers, interpreters
- Operating systems
- Filesystems, networking stacks
- Distributed middleware
- Databases
- Crypto, security protocols

A pipe dream?

Maybe until recently!

Heroic proofs of concept

- CompCert (C compiler)
- L4.verified (OS)

Proliferation of "point solutions"

- CertiKos (hypervisor)
- Verdi (distributed algorithms toolkit)
- RockSalt (software fault isolation)
- CakeML (ML compiler)
- VeLLVM (LLVM optimizations)
- HMAC + SHA (crypto)
- …

Individually impressive!

But disconnected

The Rise of Integrated Stacks

- CompCert ecosystem
- L4.verified ecosystem
- IronClad Apps
- Bedrock web server
- Everest (verified https)
- …

What makes this challenging?

(lots of things, but in particular…)

Specification Engineering!

What we learned from CompCert

Shao

OS client interface

CertiKOS hypervisor kernel

C language

Program Logic

Verifiable C System

C language

Appel

C language

CompCert Compiler

PowerPC ISA

Xavier Leroy
Inria

Peter Sewell
Univ. of Cambridge

PowerPC ISA

IBM's CPU

Transistors

What we discovered . . .

Shao

OS client interface

CertiKOS hypervisor kernel

C language

Program Logic

Verifiable C System

C language

Appel

C language

CompCert Compiler

PowerPC ISA

Xavier Leroy
Inria

Peter Sewell
Univ. of Cambridge

PowerPC ISA

IBM's CPU

Transistors

Solution: exercise spec. from both sides (2006-2015)

Shao

Appel

Xavier Leroy
INRIA

Peter Sewell
Univ. of Cambridge

(or, at least, "a")

The Future of Formal Methods…

Integration!

The Science of Deep Specification

A new NSF Expedition…

$10m
5 years

Andrew Appel

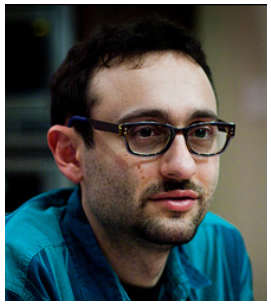
Princeton

Stephanie Weirich

Benjamin Pierce

Penn

Steve Zdancewic



Adam Chlipala

MIT

Zhong Shao

Yale

Deep Specifications

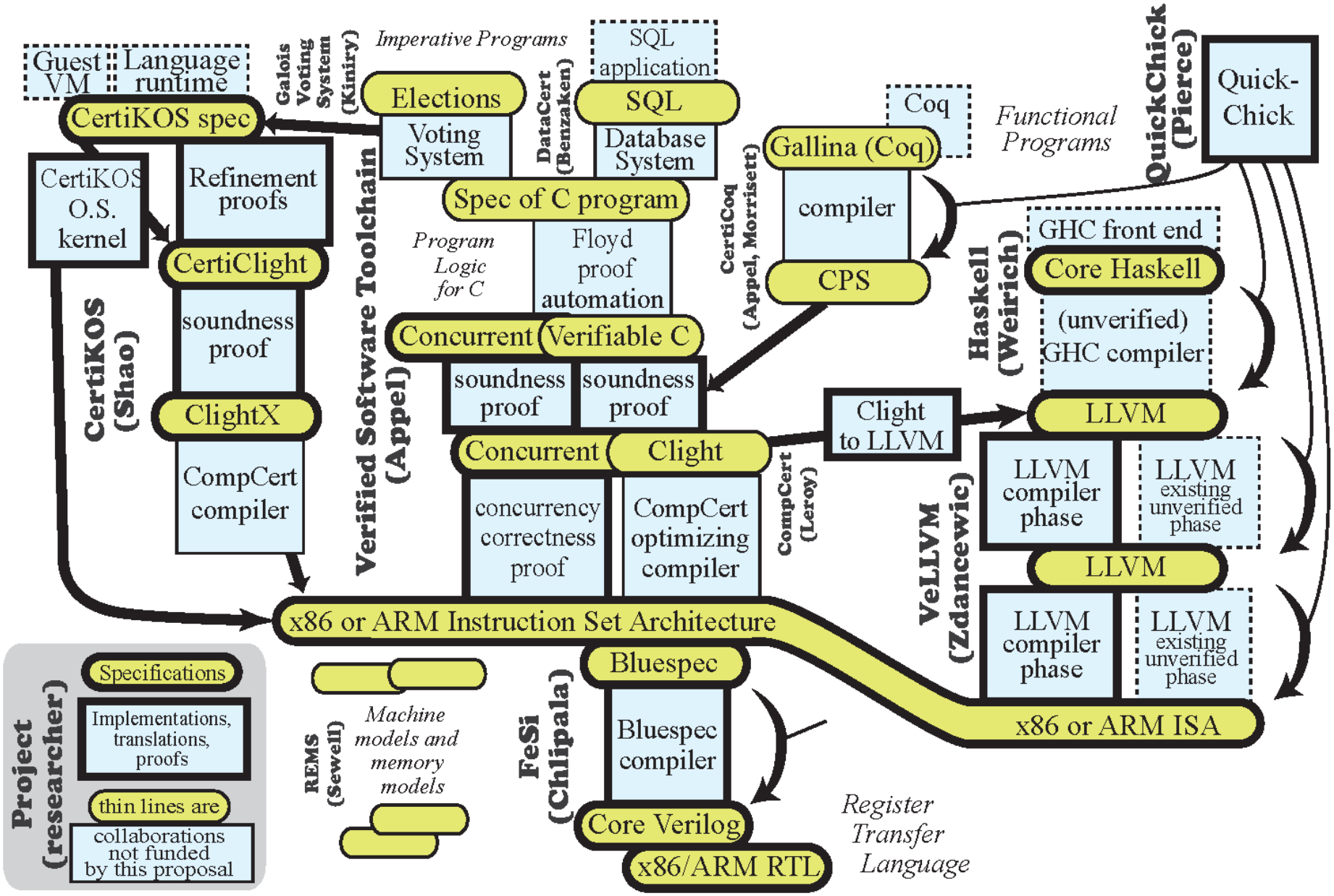
are FORMAL, RICH, LIVE, and 2-SIDED

RICH	describe complex behaviors in detail
FORMAL	in notation with a clear semantics
LIVE	machine-checked connection to implementations
2-SIDED	connected to both implementations & clients

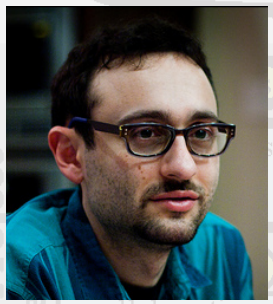
DeepSpec goals

1. Core research
2. Education
3. Community building

Core Research Topics

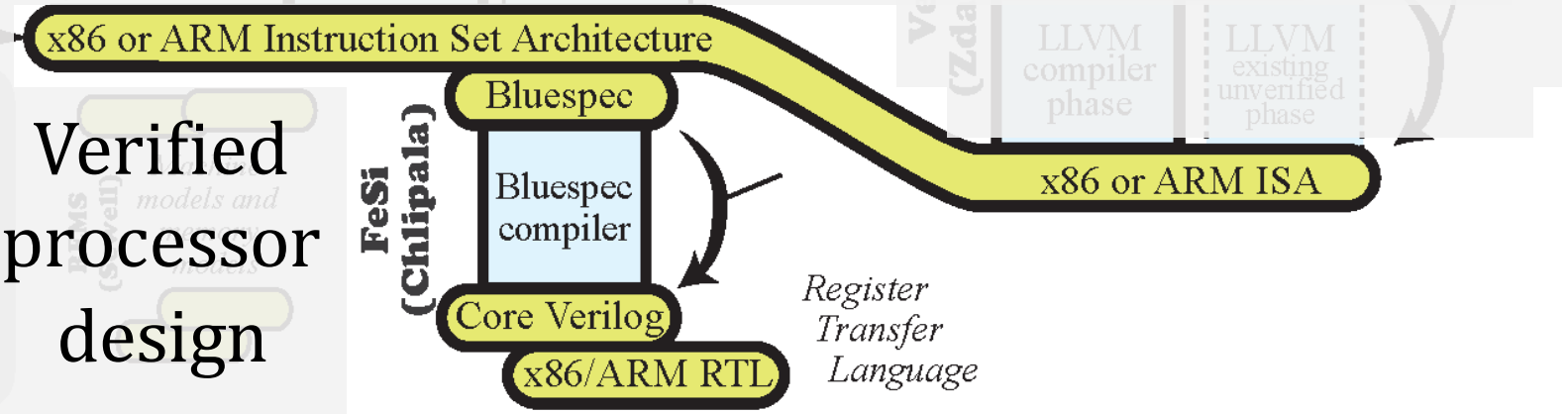


Individual projects, connected at deep specs



Adam Chlipala

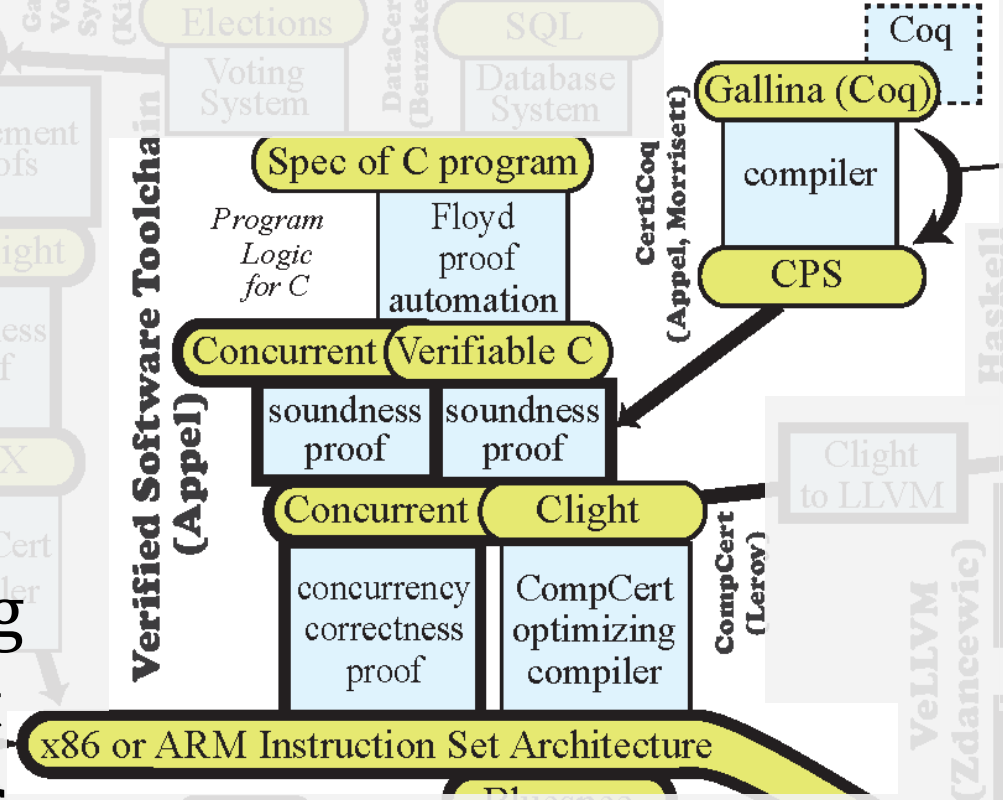
Verified processor design



Individual projects, connected at deep specs

Andrew Appel

Verified toolchain for verifying concurrent C programs



Individual projects, connected at deep specs

Steve Zdancewic

Verified LLVM compiler

Clight to LLVM

LLVM

LLVM compiler phase

LLVM existing unverified phase

VeLLVM (Zdancewic)

LLVM

LLVM compiler phase

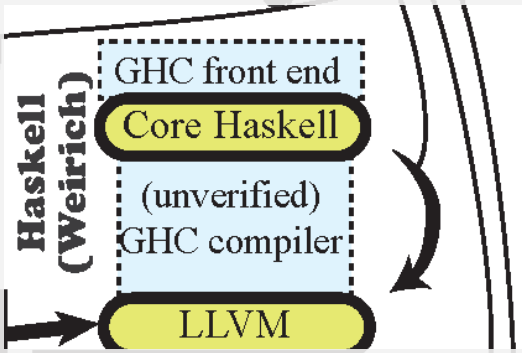
LLVM existing unverified phase

x86 or ARM Instruction Set Architecture

x86 or ARM ISA

Individual projects, connected at deep specs

Stephanie Weirich



Specification of functional language

Individual projects, connected at deep specs

Zhong Shao

Verified hypervisor OS kernel

Guest VM | Language runtime

CertiKOS spec

CertiKOS O.S. kernel

Refinement proofs

CertiClight

soundness proof

ClightX

CompCert compiler

CertiKOS (Shao)

x86 or ARM Instruction Set Architecture

x86 or ARM ISA

Individual projects, connected at deep specs

Benjamin Pierce

Specification-based random testing

Coq

Functional Programs

QuickChick (Pierce)

Quick-Chick

x86 or ARM ISA

Register Transfer Language

Specification and testing

Promising development: The rise of specification-based automated testing techniques

- Property-based random testing (QuickCheck)
- Model-based testing
- Oracle-based testing
- …

End-to-End Demo(s)

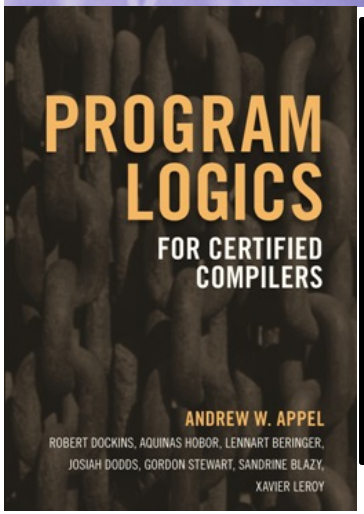
Leading candidates:

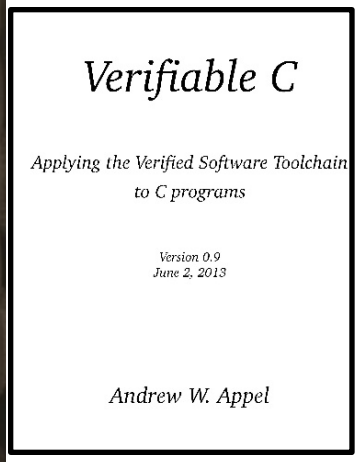
- Voting systems
- Automotive software
- Data center infrastructure

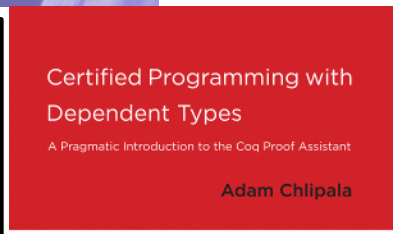
Other suggestions??

Education and training

Textbooks and on-line materials







Software Foundations text is used at dozens of universities. Now we know:

With good instructional materials and interactive proof checkers, specification & verification can be taught…

. . . just like programming and software engineering can be taught!

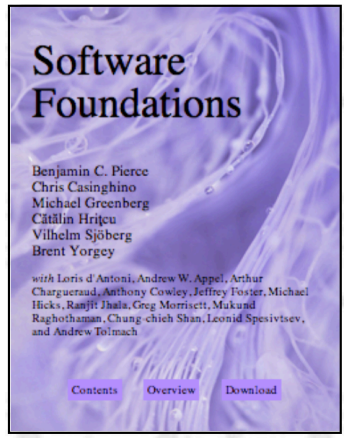
Book Development

- Goal: Use *Software Foundations* to seed a new series of "verified textbooks"
- First step:

Verified
Functional
Algorithms

Andrew Appel

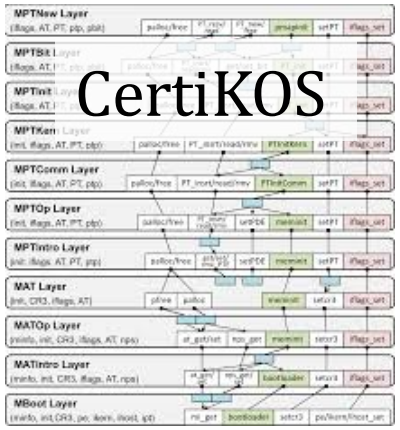
(fall 2016)



- Later:
  - A verified compiler textbook?

Curriculum Development

New Compiler & OS Courses based on



- Modularity ⇒ clean pedagogical implementations
- Precise (and correct!) description of relevant abstractions
- Specifications ⇒ automated test harnesses / test cases / property-based testing (for grading)
- Connects to formal methods course that teaches verification techniques for these artifacts

Education Specialists

Bruce Lenthall
Executive Director
Penn CTL
(Center for Teaching & Learning)

Emily Elliott
Associate Director, Penn CTL

Responsibilities:

- determine appropriate metrics for learning outcomes
- design assessment plan
- develop data collection plans
- help design measurement instruments
- analyze data
- work with IRBs

Ananda Gunawardena
Lecturer, Princeton CS

Responsibilities:

- manage implementation of data collection plan
- send out, collect, and compile assessments
- etc.

Assessment tools

1. ABET course outcomes
   - Compare "pre-DS" to "DS-ified" versions of course at the same university (e.g., Princeton), where DS-ified versions will be test driven in later years of the project
2. Student surveys
3. Instructor surveys
4. Tracking changes between successive offerings of DS-ified courses

Community building

Goal is to act as a point around which things crystallize…

- Workshops (every summer)
- Summer schools (beginning next summer)
- Visitor program (accepting applications!)
- Industrial Advisory Board
- Support for Coq development
- Jobs for postdocs, engineers, PhD students

Join us!

- DeepSpec is not about building a single system or stack
  - It's about finding out how to make *connections* between systems
- Who would *you* like to connect to?