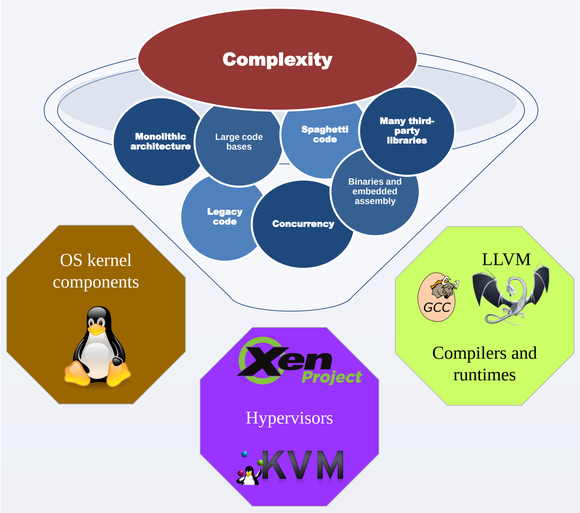
Orca: A Bottom-Up Verification Framework for Systems Software

Joshua A. Bockenek, Yakoub Nemouchi, Binoy Ravindran
{jabocken,nemouchi,binoy}@vt.edu

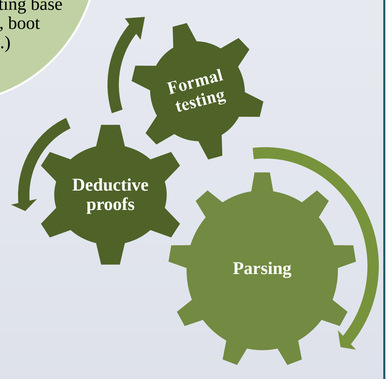
SYSTEMS SOFTWARE

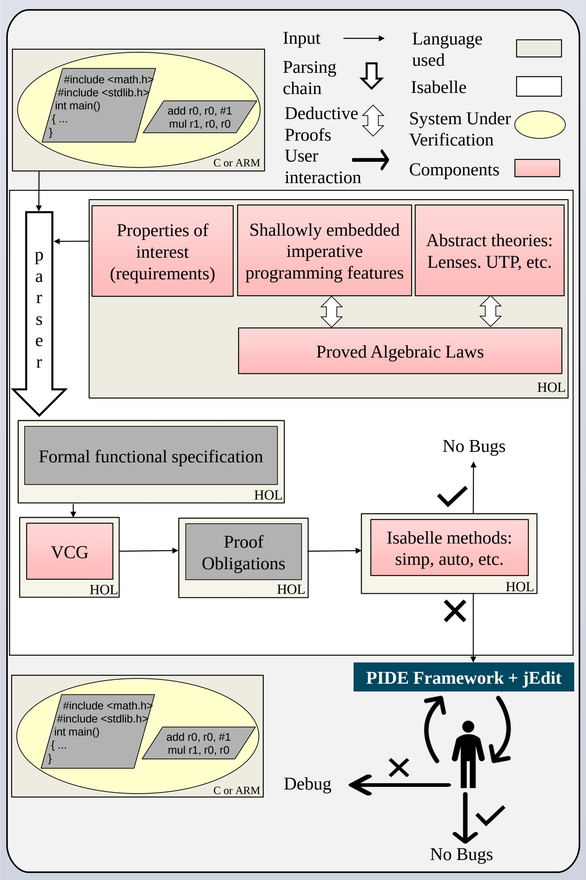


ORCA WORKFLOW

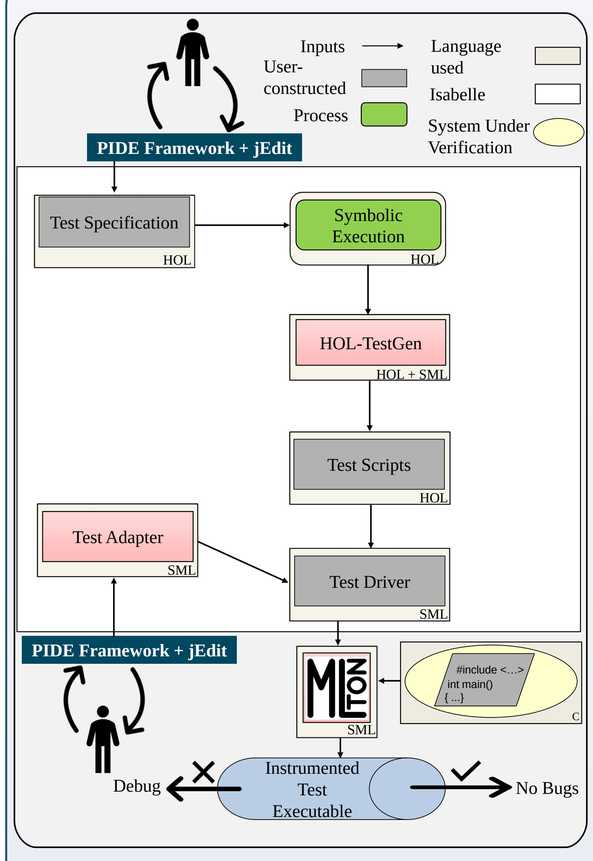
Requirements
- Sophisticated formal behavioral specification and verification techniques covering concurrency, etc.
- Bottom-up approach: scalable verification framework for legacy code.
- Large trusted computing base (compiler, assembler, boot loader, hardware, etc.)

Maximum automation!



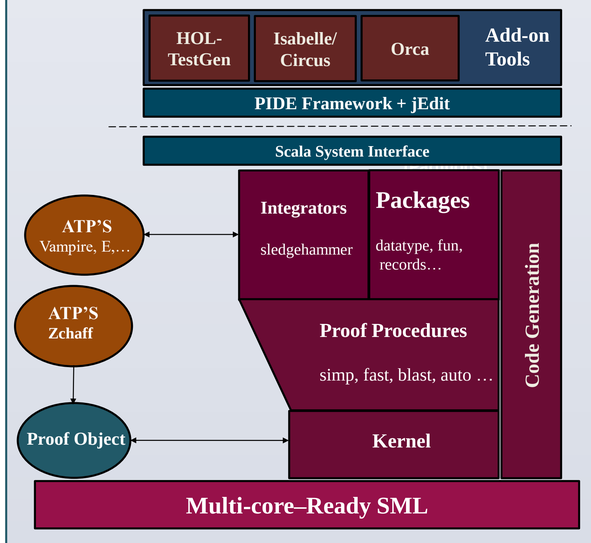


- Isabelle/HOL [10] is extended with formal specifications of a subset of the semantics of C and ARM assembly.
- Then, the parser automatically maps the raw code describing the program to a formal code specification.
- The generated formal specification is then verified using deductive proofs (both automatic and interactive) to ensure full functional correctness.



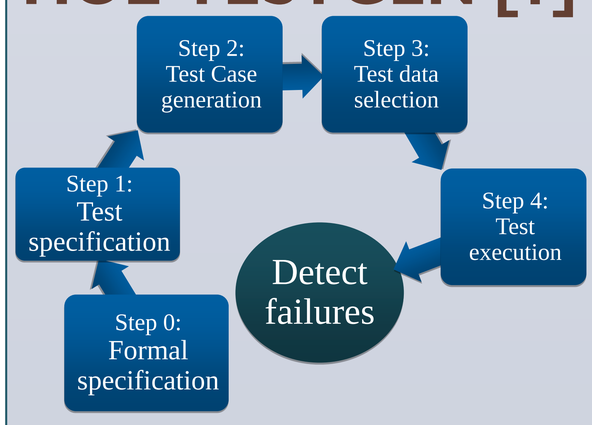
- Deductive proof verification assumes a trusted computing base (TCB) including hardware, compiler, OS, and toolchain.
- The TCB is validated using HOL-TestGen [1].

ISABELLE/HOL [9]



Isabelle
- ✓ Interactive proof assistant.
- ✓ Supports many logics (FOL, HOL, ZF).
- ✓Supports many specification constructs.
- ✓Provides tools for proof automation.

HOL-TESTGEN [1]



HOL-TestGen
- ✓Built on top of Isabelle/HOL.
- ✓Provides modeling environment for test theory.
- ✓Supports transformation of test specifications.
- ✓Supports test generation process.

STATE OF THE ART [7]

- Limited to 10kLOC.
- TCB is hard to justify.
- Has difficulty formally describing behaviors.

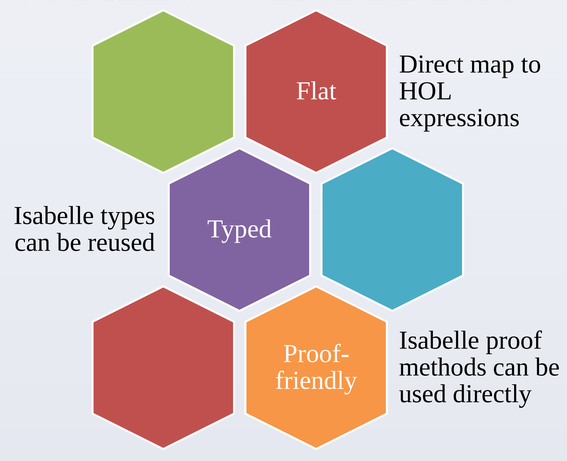
DESIGN CHOICES

Shallow embedding for semantics

State space specified via lenses [3]

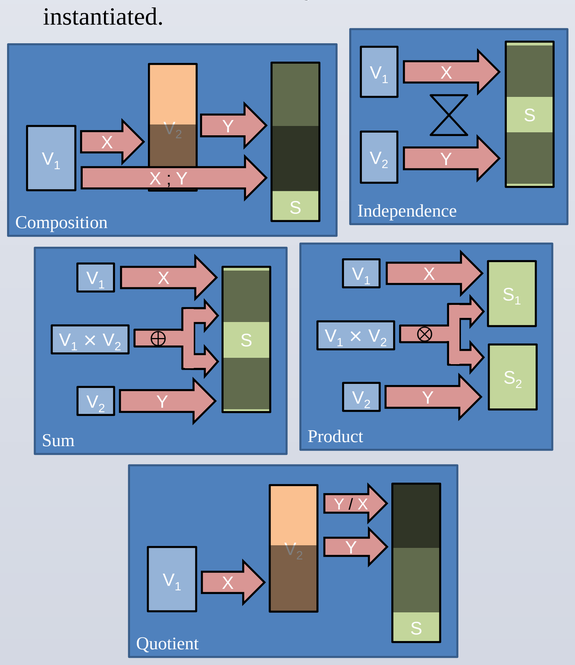
Statements specified via Unifying Theories of Programming (UTP) [2, 6]

SHALLOW EMBEDDING

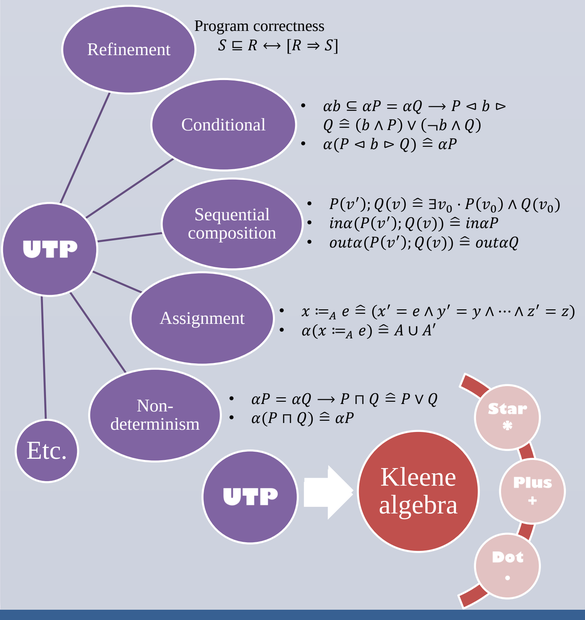


LENSES [2]

- Solve complex data structure problem (structs, unions).
- Allows an abstract memory model that can be later instantiated.



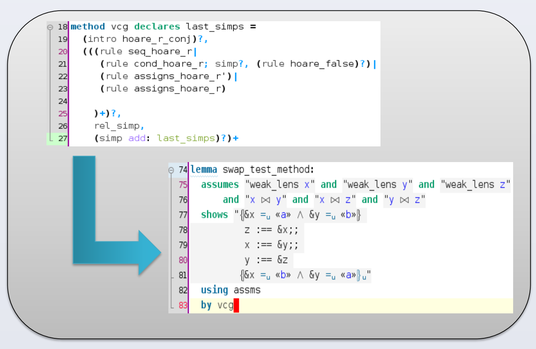
UTP [2]



VERIFICATION CONDITION GENERATOR

- Based on rules of Hoare logic [4] and algebraic laws of programming [5].
- Hoare notation: {pre-condition}program{post-condition}
- Implementation via Eisbach methods [8] and ML-level tactics.

SKIP	• $\{P\}SKIP\{P\}$
ASN	• $\{P[a/x]\}x := a\{P\}$
SEQ	• $\{P\}C_1\{Q\} \Rightarrow \{Q\}C_2\{R\} \Rightarrow \{P\}C_1; C_2\{R\}$
COND	• $\{P \wedge b\}C_1\{Q\} \Rightarrow \{P \wedge \neg b\}C_2\{Q\} \Rightarrow \{P\}C_1 \lhd b \rhd C_2\{Q\}$
WHILE	• $\{P \wedge b\}C\{P\} \Rightarrow \{P\}WHILE\ b\ DO\ C\ OD\{P \wedge \neg b\}$
PRE/POST	• $P \longrightarrow P' \Rightarrow \{P'\}C\{Q\} \Rightarrow \{P\}C\{Q\}$ • $Q' \longrightarrow Q \Rightarrow \{P\}C\{Q'\} \Rightarrow \{P\}C\{Q\}$
MISC	• $\{P\}C\{true\}$ • $\{false\}C\{Q\}$ • $\{P\}C\{Q\} \Rightarrow \{P\}C\{R\} \Rightarrow \{P\}C\{Q \wedge R\}$



CURRENT STATUS

- ✓ UTP specification extended
- ✓ VCG development nearing completion
- ✓ Test specifications in planning stage

REFERENCES

1. D. Brucker and B. Wolff. On theorem prover-based testing. *Formal Asp. Comput. (FAOC)*, 25(5):683–721, 2013.
2. S. Foster and J. Woodcock. Unifying theories of programming in Isabelle. In Z. Liu, J. Woodcock, and H. Zhu, editors, *Unifying Theories of Programming and Formal Engineering Methods*, chapter 3, pages 109–155. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.
3. S. Foster, F. Zeyda, and J. Woodcock. Unifying heterogeneous state-spaces with lenses. In *International Colloquium on Theoretical Aspects of Computing*, pages 295–314. Springer International Publishing, Oct. 2016.
4. C. A. R. Hoare. An axiomatic basis for computer programming. *Communications of the ACM*, 12(10):576–580, Oct. 1969.
5. C. A. R. Hoare, I. J. Hayes, H. Jifeng, et al. Laws of programming. *Communications of the ACM*, 30(8):672–686, Aug. 1987.
6. C. A. R. Hoare and H. Jifeng. *Unifying Theories of Programming*. Prentice Hall Englewood Cliffs, 1998.
7. G. Klein, K. Elphinstone, G. Heiser, et al. seL4: Formal verification of an OS kernel. SOSP '09, pages 207–220, New York, NY, USA, 2009. ACM.
8. D. Matichuk, M. Wenzel, and T. Murray. An Isabelle proof method language. ITP 2014, pages 390–405, Vienna, Austria, July 2014. Springer International Publishing.
9. Yakoub Nemouchi. Model-based Testing of Operating System-Level Security Mechanisms. *PhD thesis*. 2016.
10. T. Nipkow, L. C. Paulson, and M. Wenzel. *Isabelle/HOL—A Proof Assistant for Higher-Order Logic*, volume 2283 of *Lecture Notes in Computer Science*. Springer-Verlag, Heidelberg, 2002.

ACKNOWLEDGEMENTS

- Office of Naval Research
- Burkhart Wolff, University of Paris-Sud
- lewing@isc.tamu.edu (for the Tux icon)

Naval Engineering Education Consortium