LOGIC in computer science, engineering, industry (and, time permitting, math)

Yuri Gurevich

Microsoft Research

ALL MAJOR LOGIC IDEAS OF PRE-COMPUTER AGE FOUND APPLICATIONS IN CS

Types

- Frege; Russell & Whitehead
- Programming languages
- Java virtual machine, .Net

Computability

- Three approaches
  - Recursive functions
  - Lambda calculus
  - Turing machines

Recursion in CS

- Syntax of various formal languages
- Runtime recursion

Lambda calculus

- Functional programming languages

Machines

- Time and space classes
  - Juris Hartmanis and Boris Trakhtenbrot
- P=?NP
  - Steve Cook
  - Leonid Levin

Model theory

- Relational databases
  - First-order structures. Schemas, with their attributes, are improved vocabularies
  - Codd’s operations vs. Tarski’s cylindrical algebra
- Implementation independence, and polynomial time for arbitrary structures (not only strings)

Proof theory → logic engines

- SAT solvers
- Satisfiability modulo theories
- Proof engines

NEW CS ISSUES UNFORESEEN BY PRE-CS LOGICIANS

Finite automata

- Rabin & Scott
- Lexical analysis

Formal languages

- Kleene’s regular languages
- Context-free languages

Levels of abstraction

- Machine languages, assembly languages, programming languages, specification languages
- Compilers, interpreters, translators

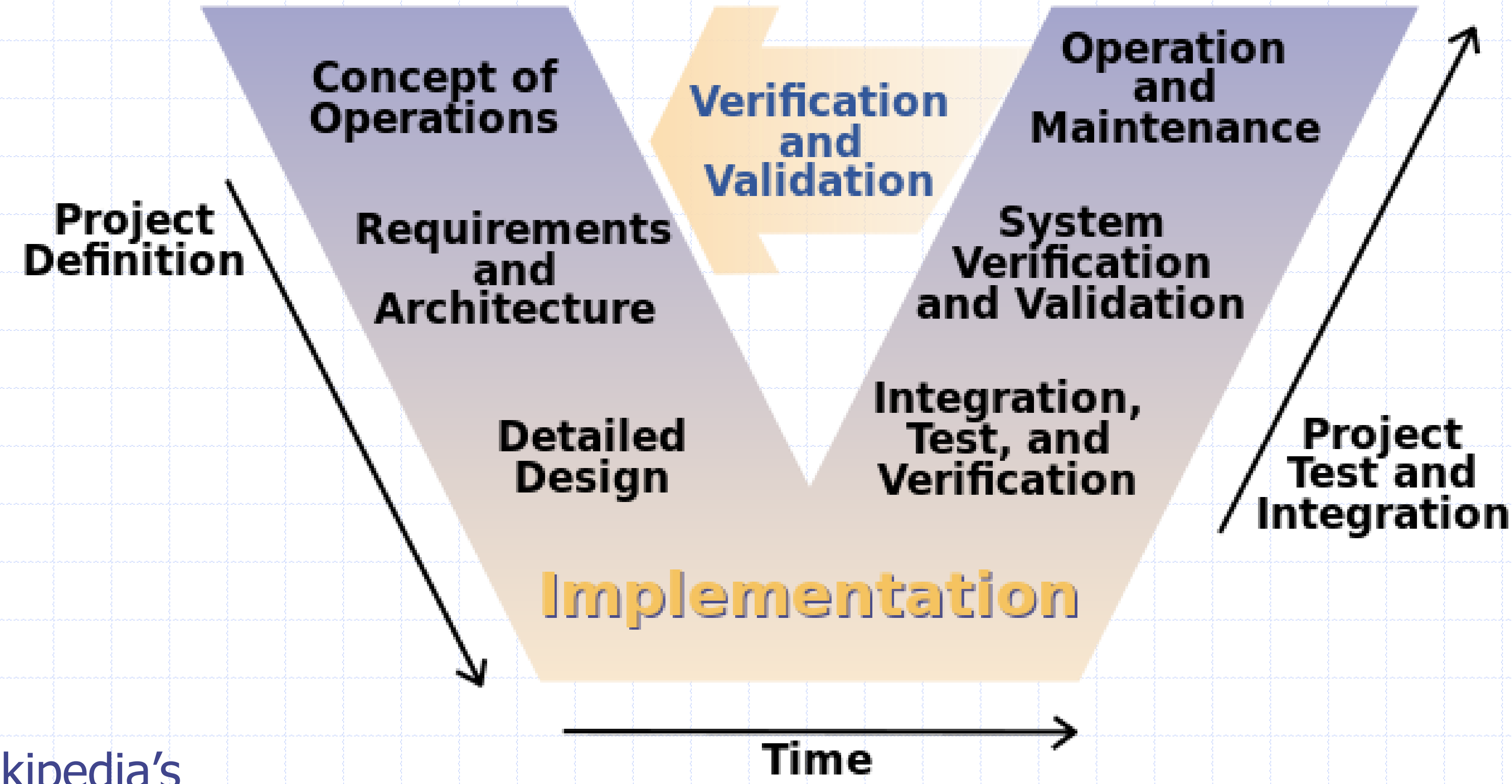
One under-appreciated achievement

Solve the "proportion"

$$\frac{FSA}{Reg} = \frac{NPDA}{CF} = \frac{PDA}{X}$$

Knuth's solution revolutionized compiler construction.

Software engineering V-diagram



Wikipedia's "V-model"

Software specifications

- Declarative vs. operational
- Abstract state machines
- Spec Explorer

EXPLICIT & IMPLICIT USE OF LOGIC BY COMPUTER ENGINEERS

Electrical engineers have been using logic explicitly

- Boolean circuits
- Many-valued logics

But the implicit use is more important.

Proliferation of formal languages

- Programming languages
- Database languages
- Specification languages
- Authentication/authorization languages
- Hardware specification languages, especially Verilog

Logic day in and day out

- Writing specifications and programming at ever increasing levels of abstraction
- Writing compilers. A compiler to L is often written in L.
- Model checking, model based testing, conformance testing
- Creating specialized languages, e.g. XACML
- Formalizing stuff e.g. certificates, claims
- Increasing use of provers

Engineers do not know logic

- Few studied logic. Instead they studied calculus which they rarely, if ever, use.
- Even the brightest of them – brilliant folks – as a rule do not know logic.

They do not have the necessary vocabulary

From a conversation with a talented software architect.

- “I guess their language is a subset of yours."

WHAT DIVIDES LOGICIANS & ENGINEERS?

Syntax divide: precise vs. "precisable"

- Logicians are cavalier about syntax. We speak about formulas but rarely write them in full.
- Engineers take syntax seriously.

Semantics divide

- Engineers often are cavalier about meaning; often syntax is presumed to convey the meaning. The price for this may be large.
- Logicians take semantics seriously. But of course we live in a cleaner world.
  - It is not enough that software has the right functionality. It should have good performance, be maintainable, be legacy compatible, etc.
- It is soundness that is most needed to engineers. Completeness is typically too good to be true.

Feasibility divide

- A problem with complexity: it is asymptotical.
  - We used to neglect complexity altogether. "Wlog φ is in cnf."
- Engineer's feasibility is realistic, down-to-earth.

Declarative/operational divide

- Logicians like declarative approaches these tend to be limited in engineering, e.g. because software evolves.
- Even in largely declarative domains typically there are operational aspects.
  - Authorization: **`if a then send permit p to x.`**
  - Obligations have imperative aspects.
- An important goal of the ASM theory was to bridge that divide: high-level and operational are consistent.

LOGIC & MATH

A history flash

- USA. Logicians have been fighting hard to get acceptance, with patchy success.
- Europe on the example of ETH Zurich.
  - Past: Zermelo, H. Weyl, Gonseth, Bernays, Specker, H. Lauchli, Engeler.
  - Present: nobody.
- Russia on a one-man example.
  - Kolmogorov did logic in 1920s and 1930s until it became too risky.
  - After the 1960s thaw, he headed Logic Dept of Moscow State University till his death.

Future

- There is inertia in academy, yet in the long run much depends on whether logicians contribute to math.
- To contribute, you better know relevant math intimately. Hence the key question:
  Is it easier for you to learn relevant math, or for a mathematician to learn (or get around) relevant logic?
- There have been impressive successes, and surely there will be more. Yet I remain skeptical about the future of logic in math.

Lefschetz’s principle

- Tarski, but also Chevaller (Constructive sets are closed under projections.)
- “Although I am aware of the precise formulations using first order logic and beyond ..., I tend not to use them. Rather I view the Lefschetz principle as more of a philosophical principle of what ought to be possible in general, and do the necessary verifications as and when I need them ... I suspect this attitude is pretty common among many algebraic geometers,” Donu Arapura.

THANKS