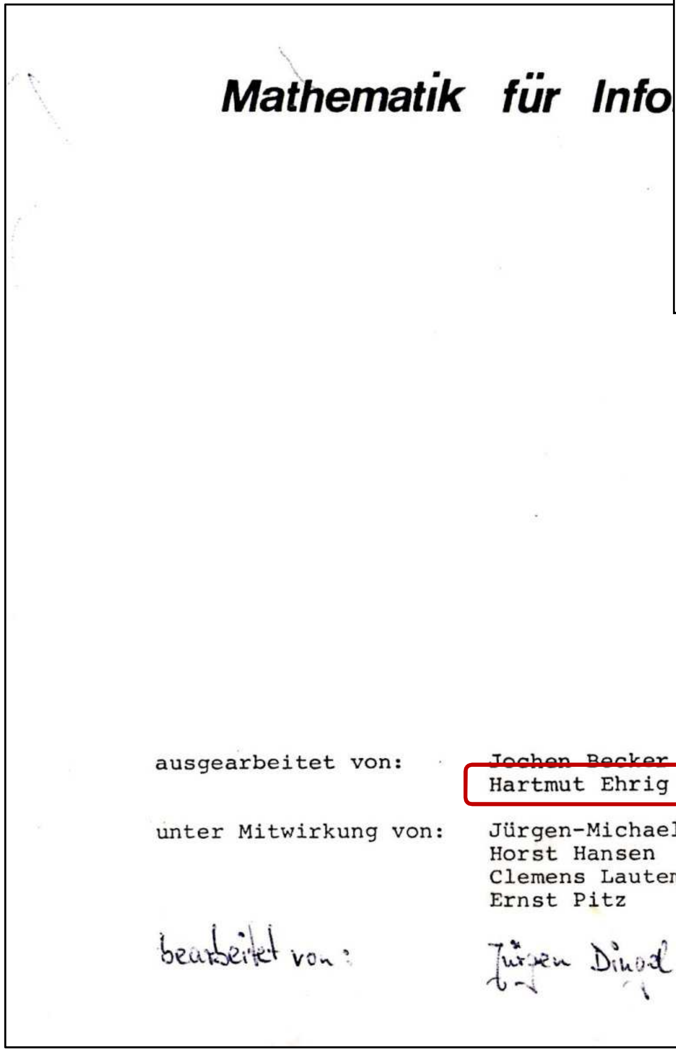
Complexity is the Only Constant: Trends in Computing & Their Relevance to MDE

Juergen Dingel

July 5, 2016

30 Years Ago at the TU Berlin

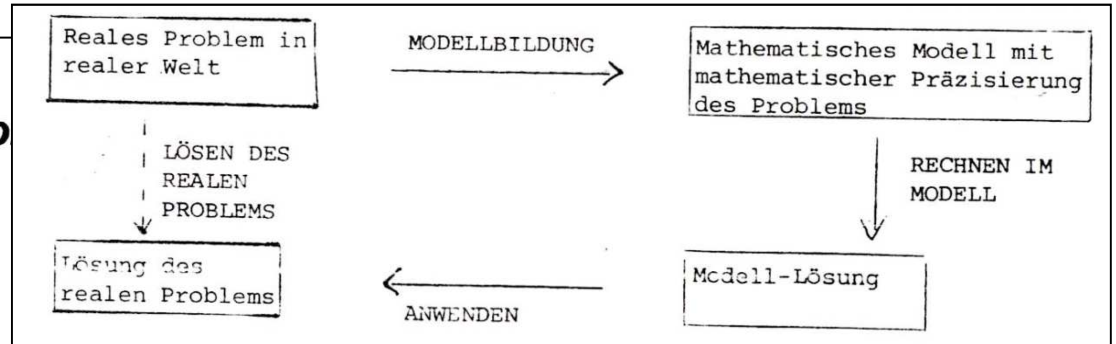


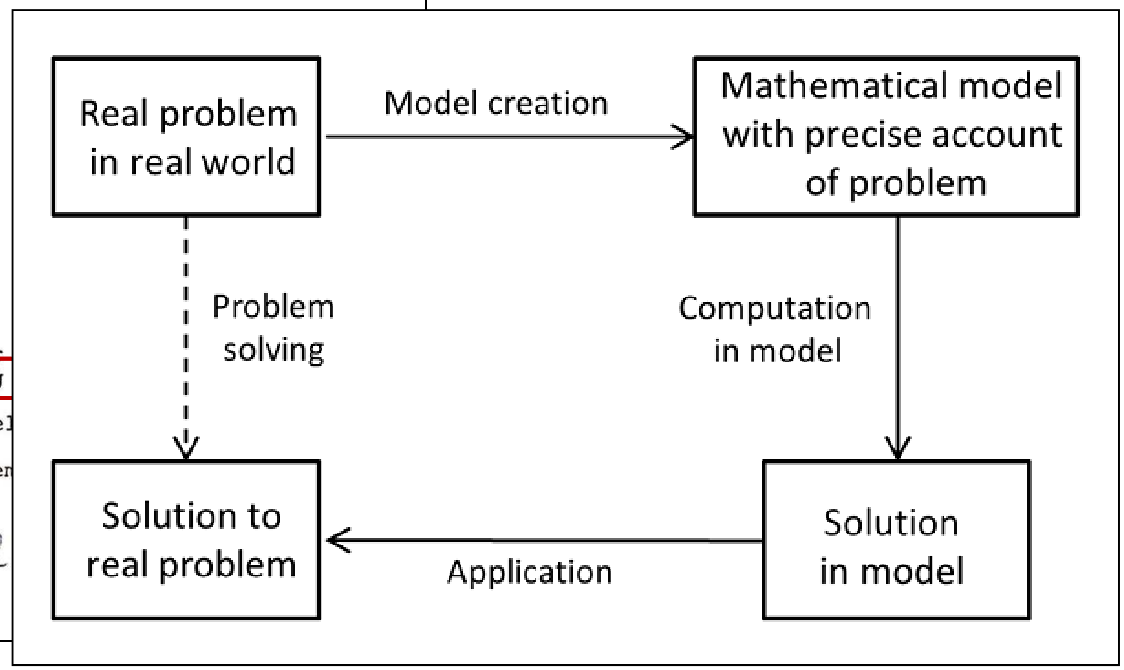
Mathematik für Info

ausgearbeitet von: Jochen Becker
Hartmut Ehrig

unter Mitwirkung von: Jürgen-Michael
Horst Hansen
Clemens Lauten
Ernst Pitz

bearbeitet von: Jürgen Dingel

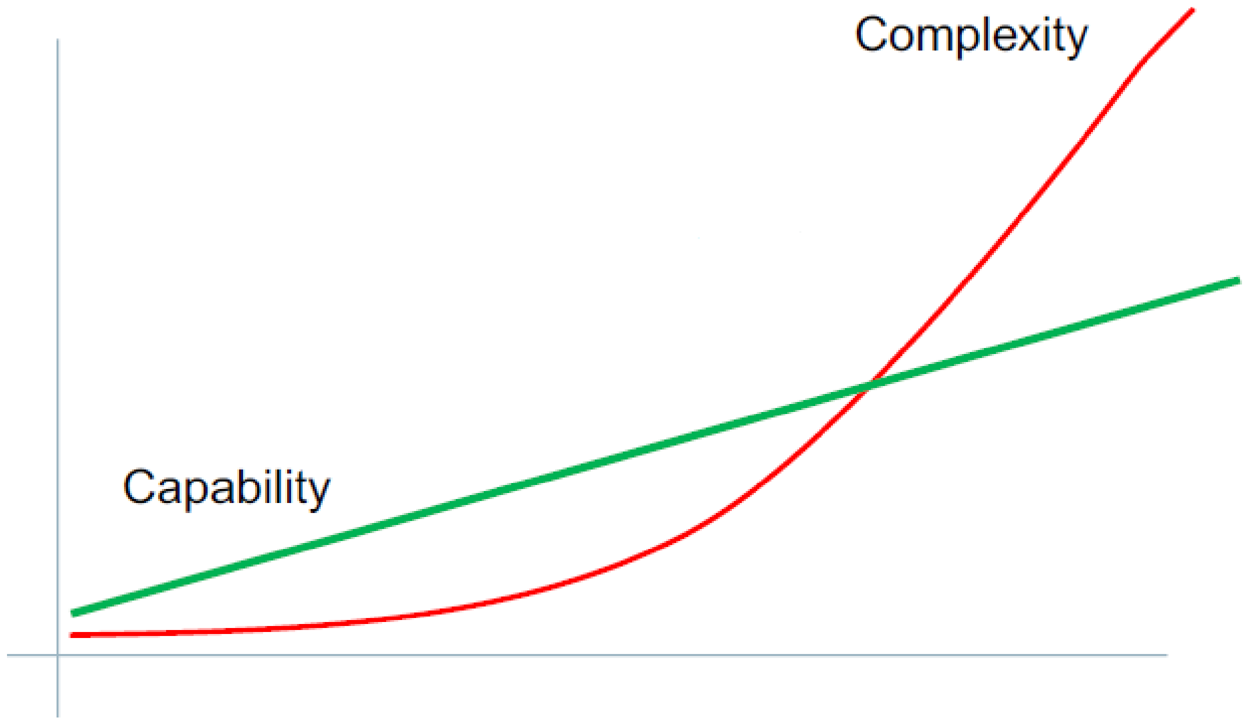




Goals for This Talk

- **The more things change, the more they stay the same**
  - Changes: Progress
  - Constants: Complexity & techniques to deal
- **Highlight some select work**
  - Semantics engineering
  - Synthesis
  - Provenance

48 Years Ago at 1st NATO SW Eng Conference



HW computing power ⇑

⇒ Complexity of tasks SW asked to do ⇑

⇒ Complexity of SW ⇑

⇒ Existing SW development capabilities strained

⇒**"Software crisis"**

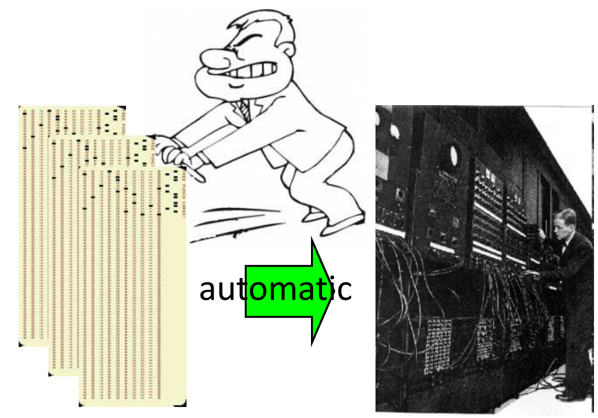
Since Then: LOTS of Progress

- **Hardware**
  - **Computing power** (2016 vs 1969) [Paul Ledak on quora.com]:
    - Number of transistors:
      - iPhone 6 = Apollo 11 GC x 180,000
    - Clock frequency:
      - iPhone 6 = Apollo 11 GC x 32,000
    - Instructions per second:
      - iPhone 6 = Apollo 11 GC x 80 million
    - Overall:
      - iPhone 6 = Apollo 11 GC x 120 million
  - **Cost of 1 MB of memory** in US$ [www.jcmit.com]:
    - Dec 2015 = 1957 / 100 billion

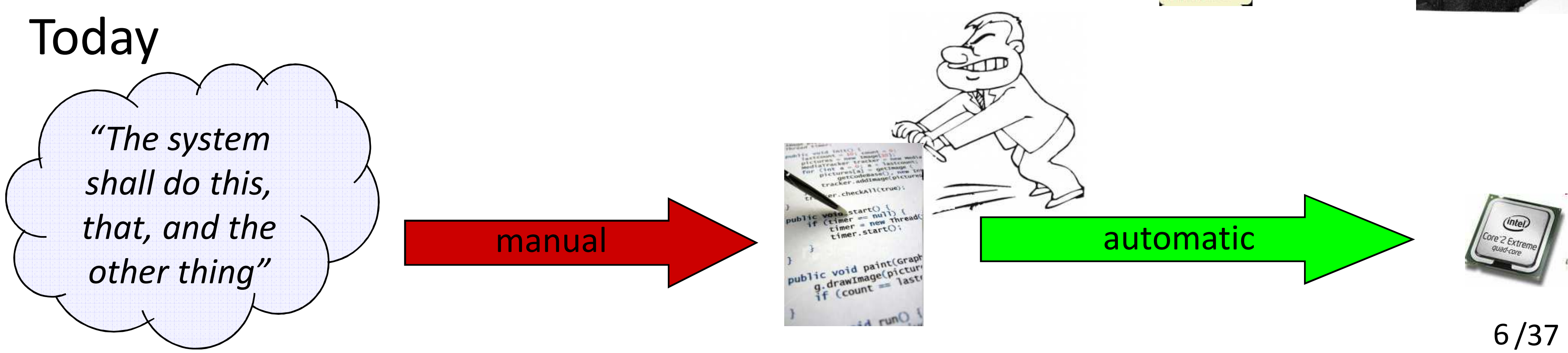
Since Then: LOTS of Progress

- **Software engineering**
  - Information hiding via modularization, encapsulation, interfaces, MDE, …
- **Programming languages**
  - Compilers, user-defined data types, OO, …
- **Data bases**
  - Relational model, …
- **Operating systems**
  - Virtual memory, …

40 years ago



Today



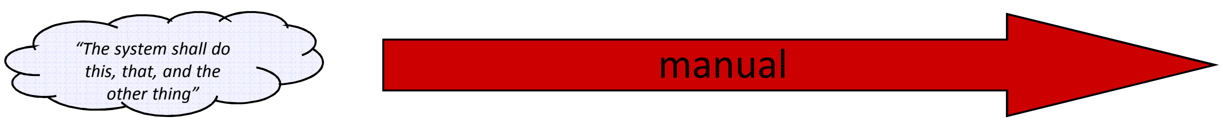
Since Then: LOTS of Progress

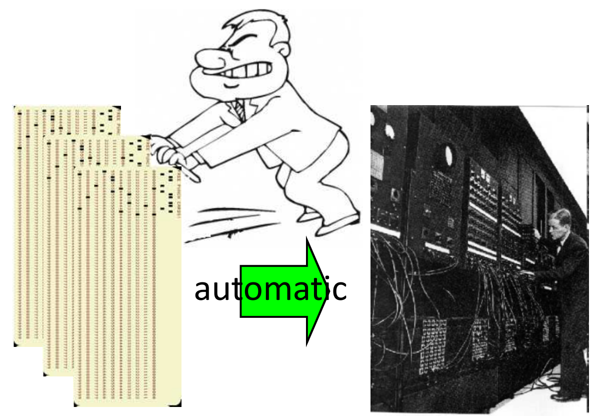
- **Software engineering**
  - Information hiding via modularization, encapsulation, interfaces, MDE, …
- **Pro**
  -
- **Da**
  -
- **Operating systems**
  - Virtual memory, …

Key general techniques:

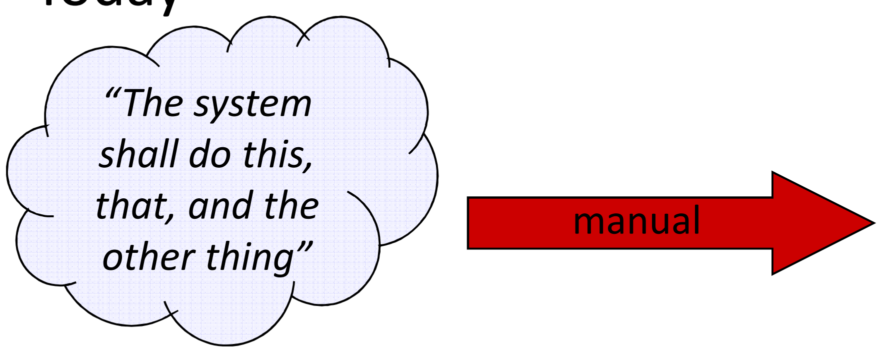
Abstraction, automation, and analysis

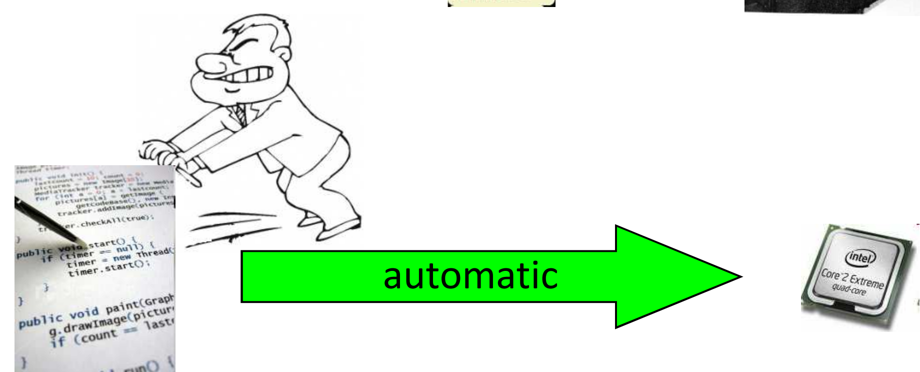
40 years ago





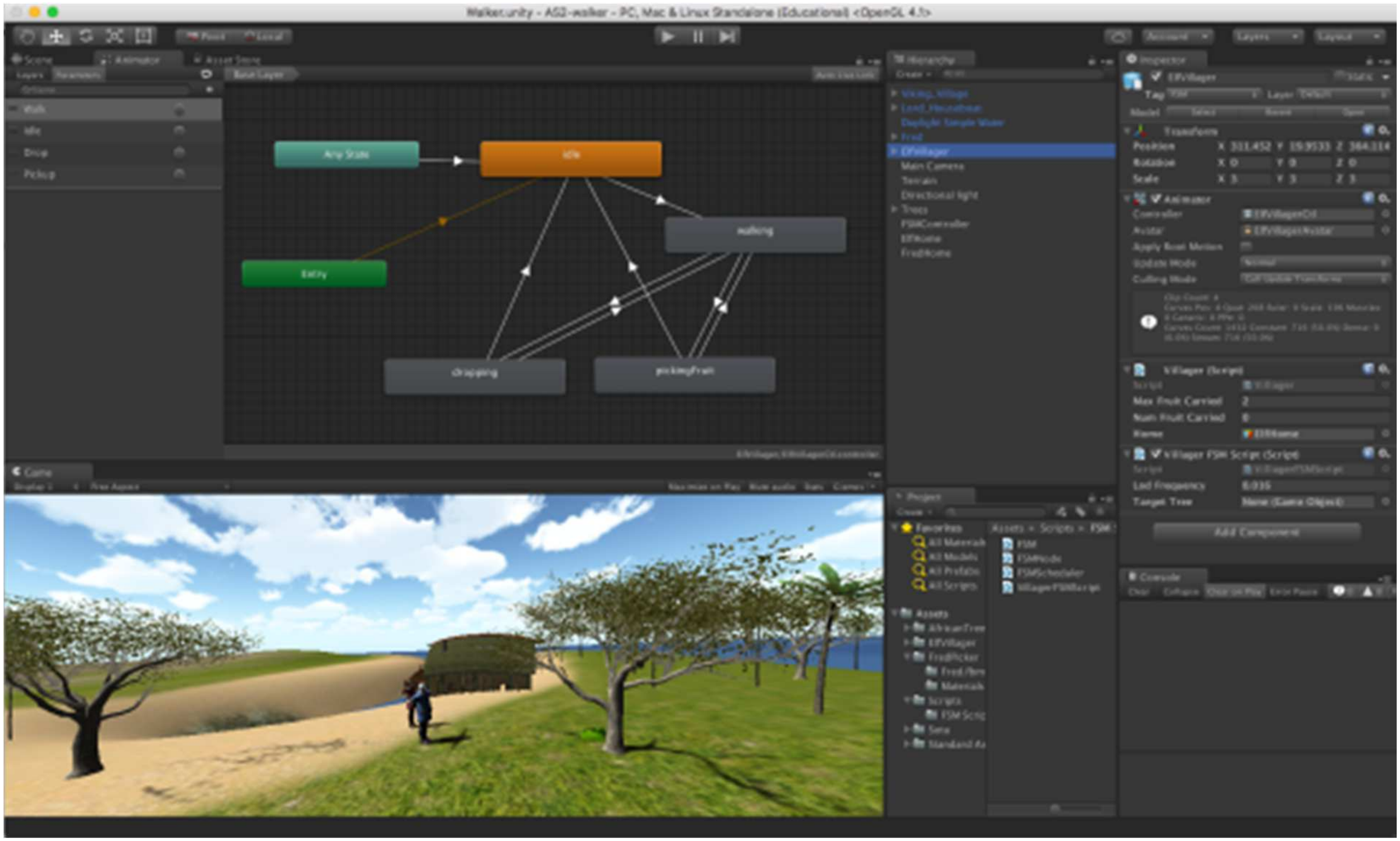
Today





Even the Game Industry is Using MDE Now

http://docs.unity3d.com/Manual/Animator.html

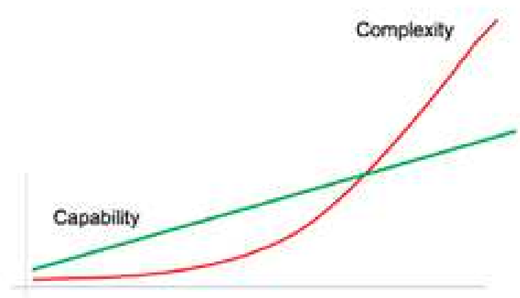


Screenshot courtesy Nick Graham

But, We Still Seem to Be in Crisis

Avionics: limits of affordability near

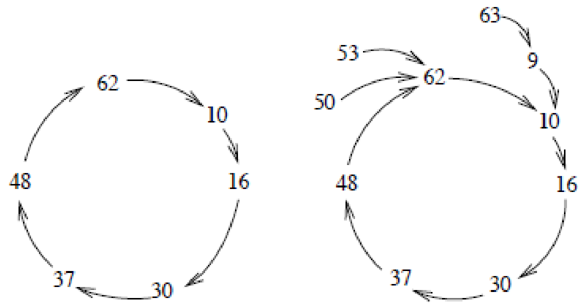
The cautionary tale of Chord



HW computing power ⇑
⇒ Complexity of tasks SW asked to do ⇑
⇒ Complexity of SW ⇑
⇒ Existing SW development capabilities strained
⇒“Software crisis”

Still valid

The Cautionary Tale of Chord



Chord: Distributed hash table [Chord01]

[Chord01] Stoica, Morris, Karger, Kaashoek, Balakrishnan. *"Chord: A scalable peer-to-peer lookup service for Internet applications"*. SIGCOMM. 2001.

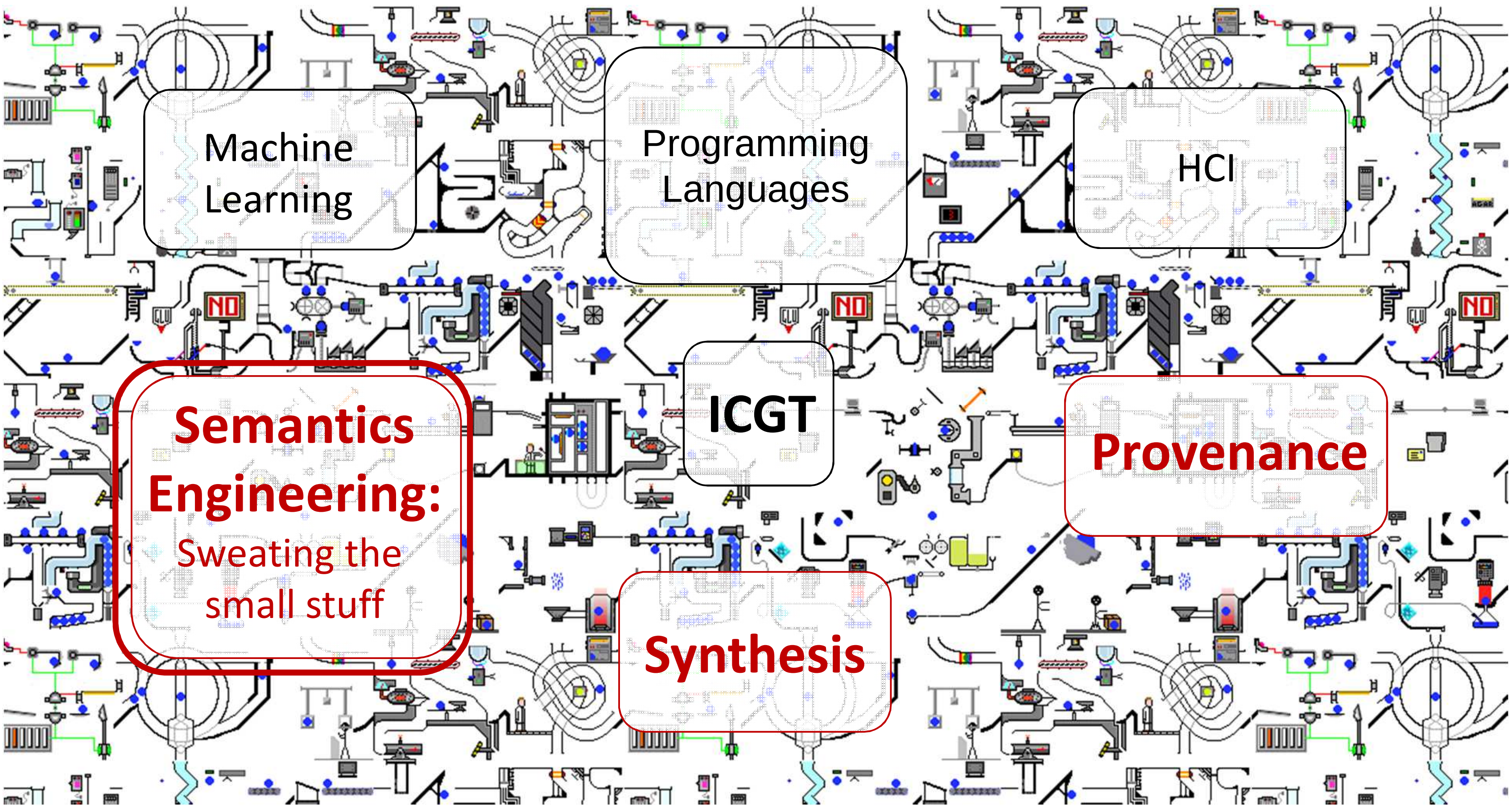
- *"3 features that distinguish Chord from many other peer-to-peer lookup protocols are its simplicity, provable correctness, and provable performance"*
- Papers present properties, invariants and manual proofs
- 4th most-cited paper in CS for years (CiteSeer)
- 2011 SIGCOMM Test-of-Time Award

"Unfortunately, the claim of correctness is not true. The original specification [...] does not have eventual reachability, and not one of the seven properties claimed to be invariants [...] is actually an invariant."
"For complex protocols such as Chord, there is every reason to use lightweight modeling as a design and documentation tool"

P. Zave. 2012.

Various papers on http://www.research.att.com/~pamela/chord.html

Research Landscape is Complex, too



http://blueballfixed.ytmnd.com/

Semantics Engineering: Background

Big advances in use of formal semantics

E.g., formalization (and verification) of

- OS kernels [Klein et al, CACM'10]
- Programming languages (Java, JS) [Rosu et al, '15]
- (Optimizing) compilers [CompCert, CACM'09]
- Concurrent code
  - Fine-grained locking ('hand-over-hand locking')
  - Lock-free data structures ('lock-free queues')
  - Preemptive OS kernels [Feng et al, CAV'16]

First-order logic, Isabelle/HOL

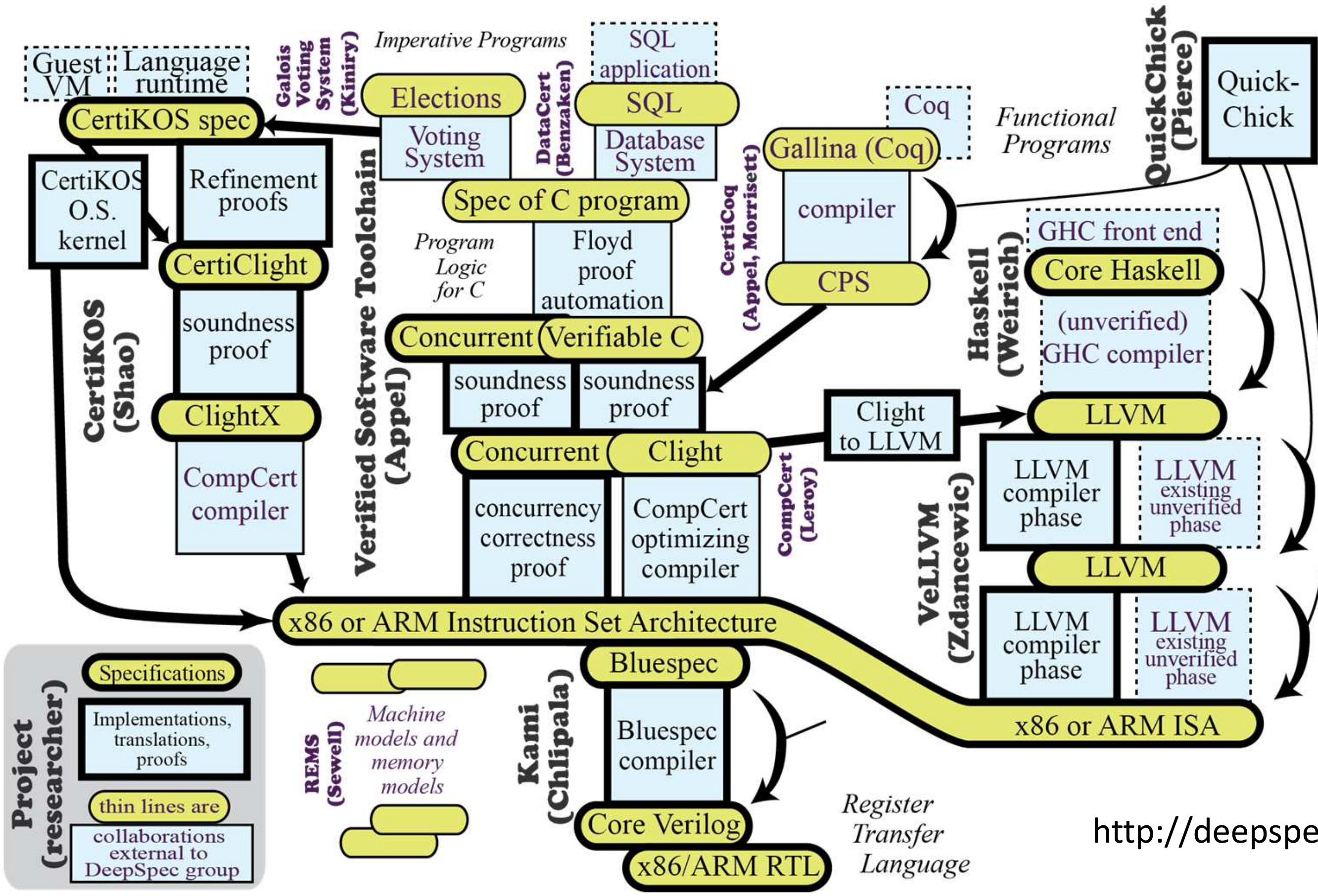
Rewrite logic, Maude

First-order logic, Coq

Separation logic

Next: Verifying Entire Software Stacks

The science of deep specification [DeepSpec.org, Appel et al, US$10million over 5 years from NSF]



http://deepspec.org/research/

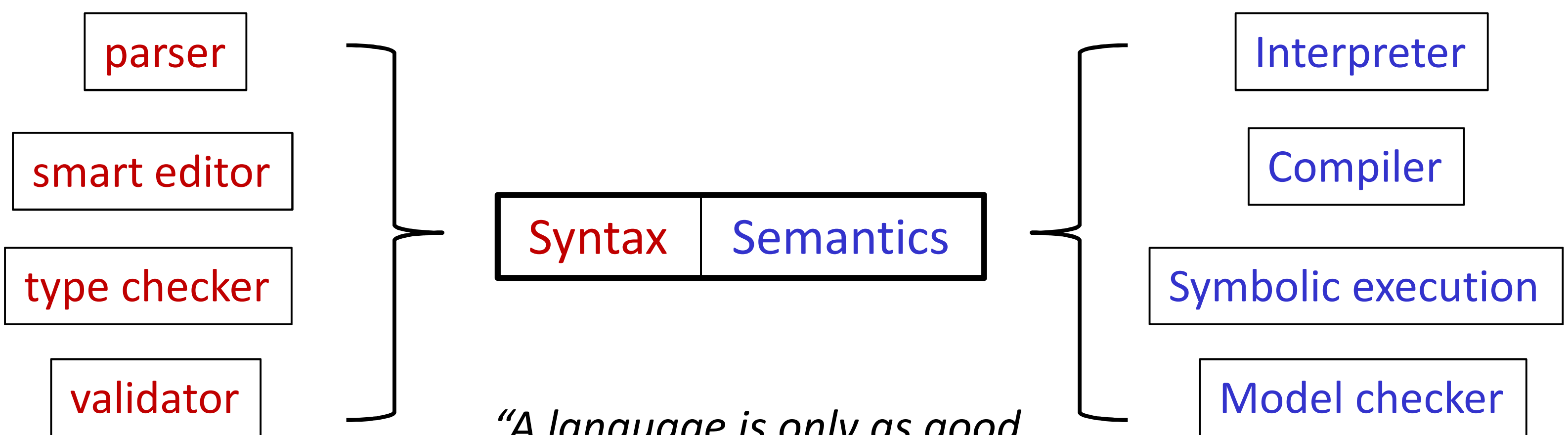
Semantics Engineering

What

- Notations, techniques, tools for creating, manipulating, analyzing formal descriptions of (execution) semantics (of a language)
- To facilitate analysis, development of supporting tooling, …

Inspiration

Make descriptions of execution semantics as useful and common as descriptions of syntax



"A language is only as good as its supporting tooling"
[Bran Selic]

Semantics Engineering: Some Related Work

Notations to specify semantics

- Rewrite logic (Maude [Marti-Oliet & Meseguer et al, ‘98])
- Graph transformation (e.g., Dynamic MM [Engels et al, ‘00], Mograms [Kleppe, ‘08])
- DSL (e.g., PLT Redex [Felleisen, Findler & Flatt, ‘09])

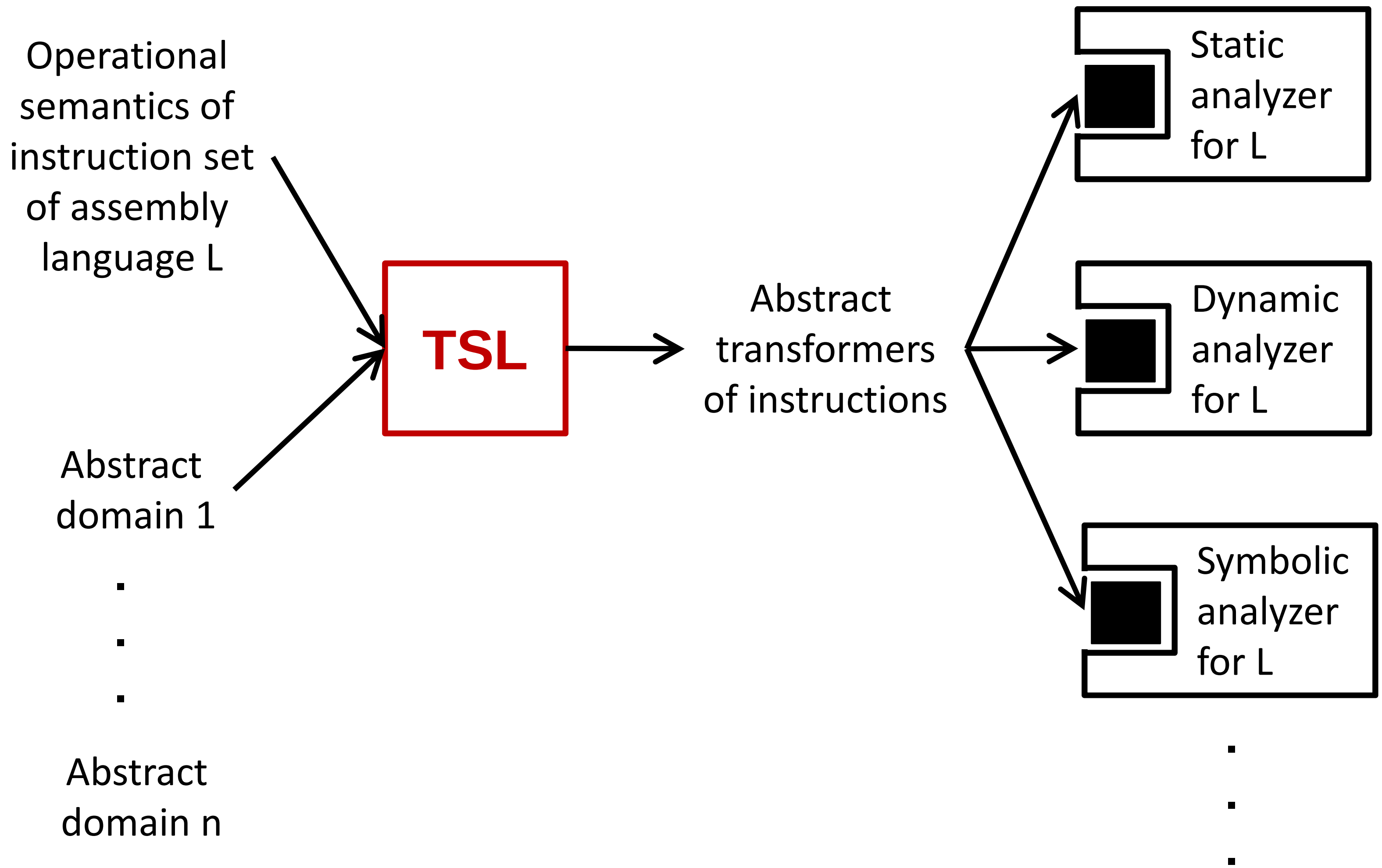
Use semantics to customize supporting tools

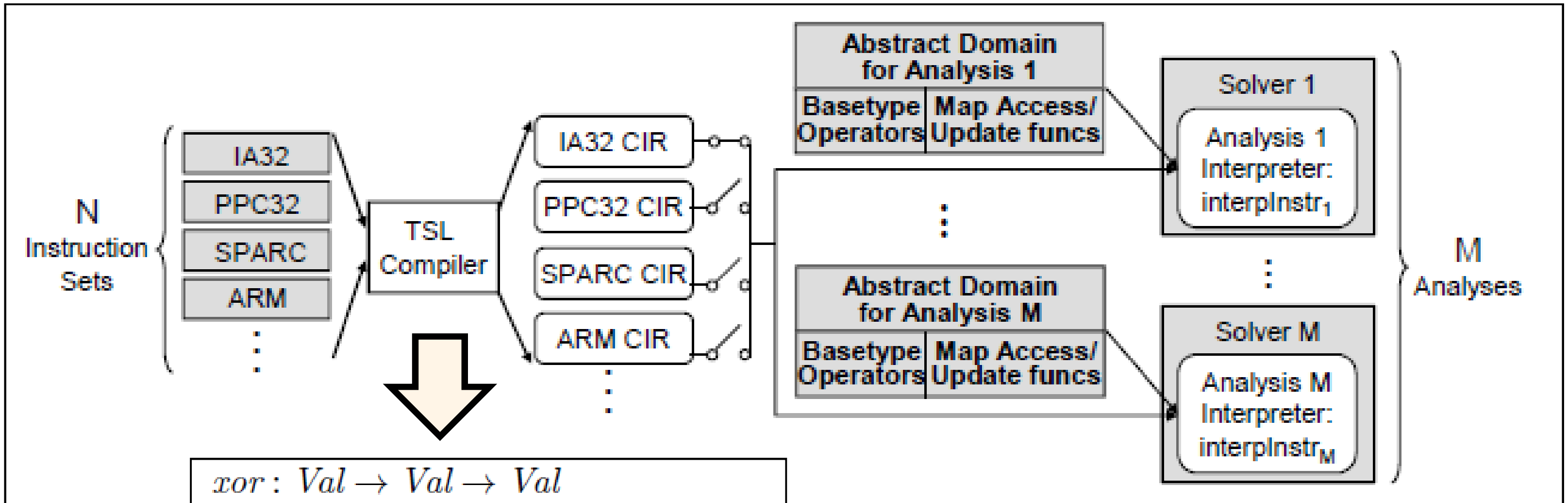
- Code generation [Day & Atlee et al, ‘12]
- Interpretation [Dingel & Zurowska, ‘14]

Use semantics to generate supporting tools

- ASF+SDF [van den Brand & Klint et al, ‘05]
- TSL [Lim & Reps, ’13]

TSL: Generating Analyzers from Semantics





Semantic core

$xor : Val \rightarrow Val \rightarrow Val$
$lookup : State \rightarrow Id \rightarrow Val$
$store : State \rightarrow Id \rightarrow Val \rightarrow State$

$\mathcal{E} : Expr \rightarrow State \rightarrow Val$
$\mathcal{E}[\![I]\!]\sigma = lookup\ \sigma\ I$
$\mathcal{E}[\![E_1 \oplus E_2]\!]\sigma = \mathcal{E}[\![E_1]\!]\sigma\ xor\ \mathcal{E}[\![E_2]\!]\sigma$

$\mathcal{I} : Stmt \rightarrow State \rightarrow State$
$\mathcal{I}[\![I = E;]\!]\sigma = store\ \sigma\ I\ \mathcal{E}[\![E]\!]\sigma$

Standard interpretation

$v \in Val_{std} = Int32$
$State_{std} = Id \rightarrow Val$

$lookup_{std} = \lambda\sigma.\lambda I.\sigma I$
$store_{std} = \lambda\sigma.\lambda I.\lambda v.\sigma[I \mapsto v]$
$xor_{std} = \lambda v_1.\lambda v_2.v_1 \oplus v_2$

Abstract interpretation: 'signs' analysis in two's-complement

$v \in Val_{abs} = \{neg, zero, pos\}^{\top}$
$State_{abs} = Id \rightarrow Val_{abs}$
$lookup_{abs} = \lambda\sigma.\lambda I.\sigma I$
$store_{abs} = \lambda\sigma.\lambda I.\lambda v.\sigma[I \mapsto v]$

$xor_{abs} = \lambda v_1.\lambda v_2.$

		v_2: neg	$zero$	pos	$\top$
v_1	neg	$\top$	neg	neg	$\top$
	$zero$	neg	$zero$	pos	$\top$
	pos	neg	pos	$\top$	$\top$
	$\top$	$\top$	$\top$	$\top$	$\top$

TSL

Results

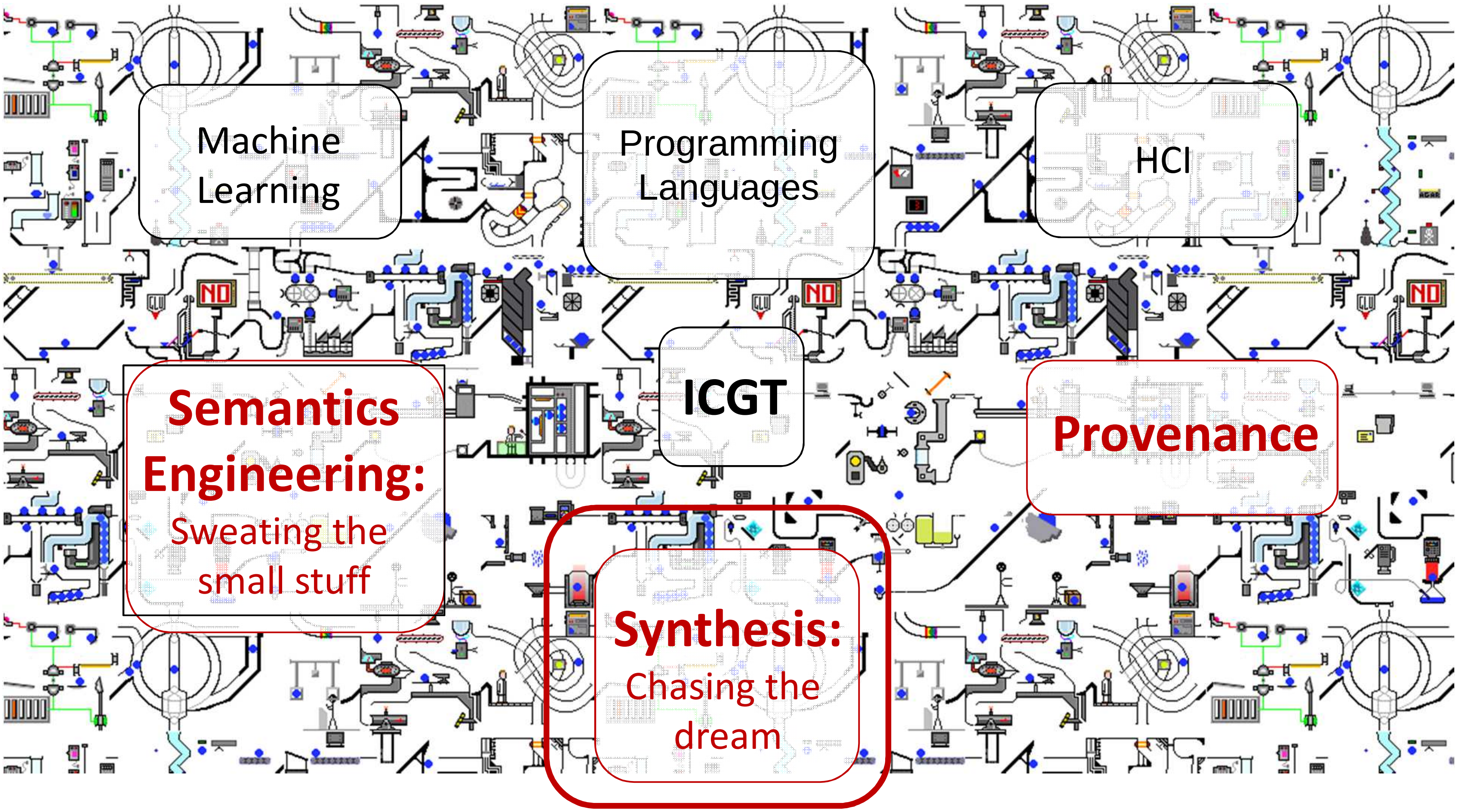
- CodeSurfer/x86 (1 vs 20 man months)
- Generated transformers very precise (optimal for 97.5% instructions)
- Different static analyses for IA32 and PowerPC
- Model checker
- Botnet analyzer

- Represents an astounding unification of research topics

Semantics Engineering: Concluding Remarks

- Significant progress
- Open questions
  - Most suitable ways to specify semantics?
    - DSL (e.g., TSL, PLTRedex)
    - Translation to GPL (e.g., Xsemantics)
    - First-order logic (e.g., CompCert)
    - Rewrite logic (e.g., Maude)
    - Graph transformation
  - How to improve support?
    - Testing, analysis (e.g., Groove)
    - Visualization
    - Automation
  - “Killer applications”?
    - DSL integration?

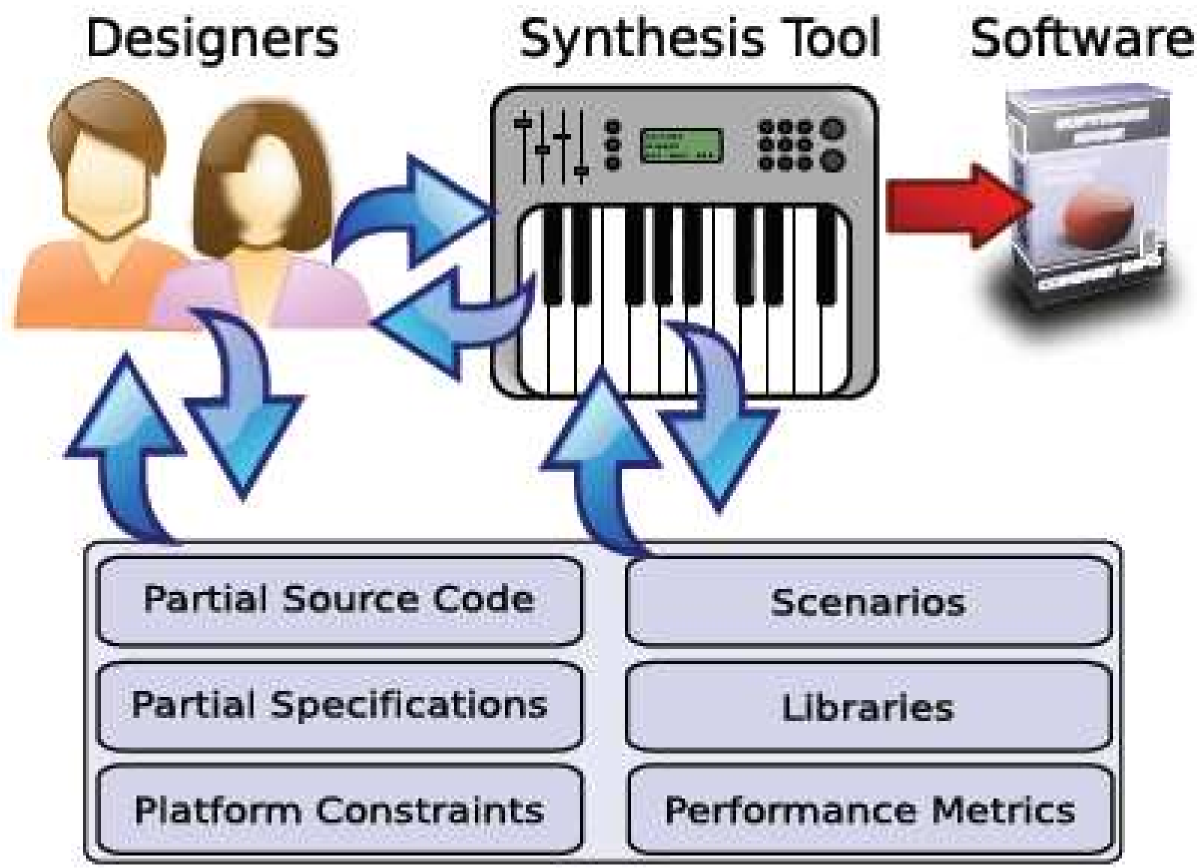
Research Landscape is Complex, Too



http://blueballfixed.ytmnd.com/

Synthesis

ExCAPE project in US (UPenn, Berkeley, MIT, Cornell, …), https://excape.cis.upenn.edu/



ExCAPE: Some Results [Alur et al, 2015]

(Semi-automatic) synthesis of, e.g.,

- Program from specs (e.g., pre-, post, program template)
- Protocols from partial EFSMs, invariants, and scenarios
- Spreadsheet expressions from examples
- Biological models
- Optimal programs (e.g., bitvector manipulation, array search)

Key techniques

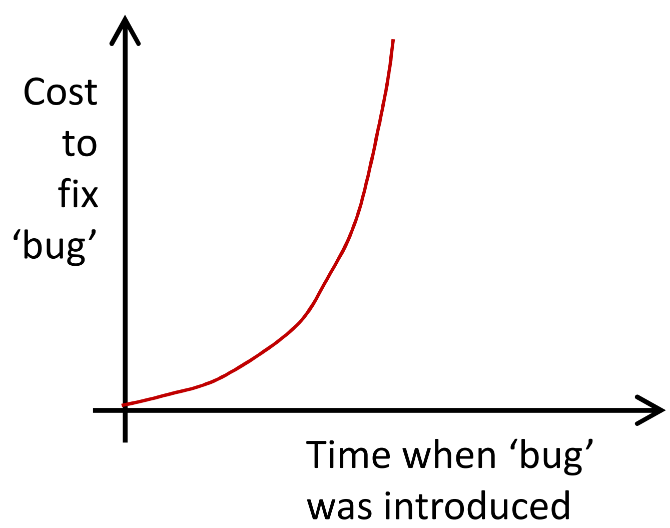
- SMT and SAT
- Machine learning
- DSLs

Synthesis: So What?

Enable more abstraction and automation for, e.g.,

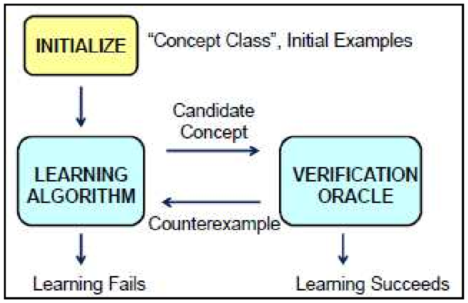
- More user-friendly, yet executable specifications
- Treatment of partial, incomplete models
  - Automatic completion, early analysis

  ⇒ Finding problems earlier



Key Techniques

- **Constraints and constraint solving**
  - Better integration into PLs
    - Constraints , solving, symbolic variables, …
    - GPLs: e.g., Kaplan = Scala+Contraints [Kuncak et al, POPL'12]
    - DSLs: e.g., Rosette = framework for solver-aided DSLs [Torlak et al, Onward!'13]
- **Counter example-guided inductive synthesis (CEGIS)**
    - Learning from examples and counter-examples
    - Solves "$\exists x. \forall y. \varphi(x,y)$" type formulas
    - Often 'syntax-guided': "$\exists x \in G. \forall y. \varphi(x,y)$"



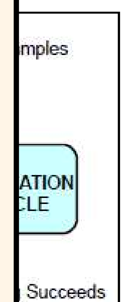
- **DSLs**
  - Define manageable candidate space G
- **Machine learning (inductive inference)**

Key Techniques

- **Constraints and constraint solving**
  - Better integration into PLs
    - Constraints, solving, symbolic variables

!'13]

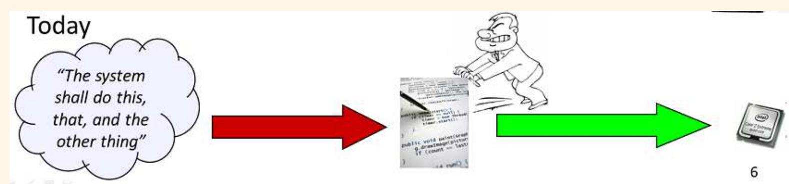
- C
- I
- M



Questions:

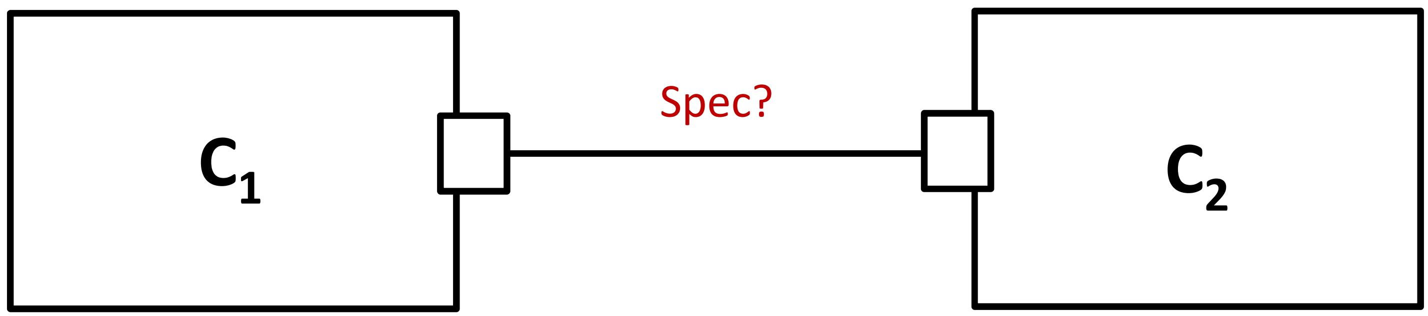
How could these techniques be used to facilitate, e.g.,

- use of more user-friendly ('declarative') models
- completion, execution, analysis of partial, incomplete models
- model transformation
- design space exploration
- …



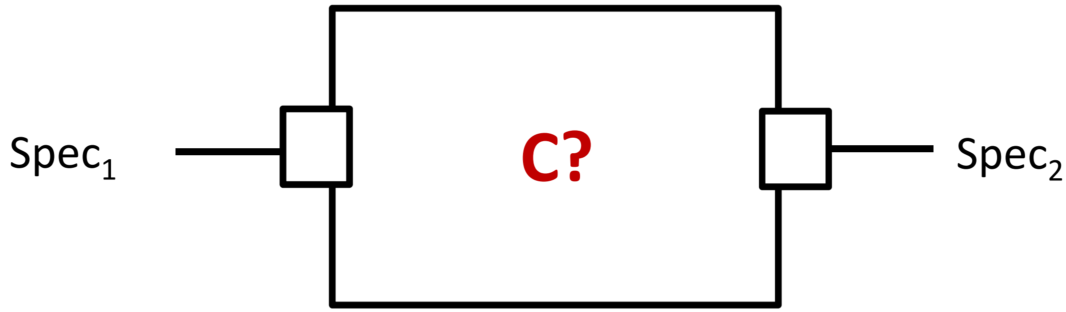
CEGIS for Interfaces & Implementations?

Interface extraction



$\exists$Spec. $\forall e \in \text{Exec}(C_1 \,||\, C_2)$. Conform(e, Spec)

Implementation generation



$\exists$C. $\forall e \in \text{Exec}(C \,||\, \text{Spec}_1 \,||\, \text{Spec}_2)$. Complete(e, C)

CEGIS for Model Transformations?

Transformation generation

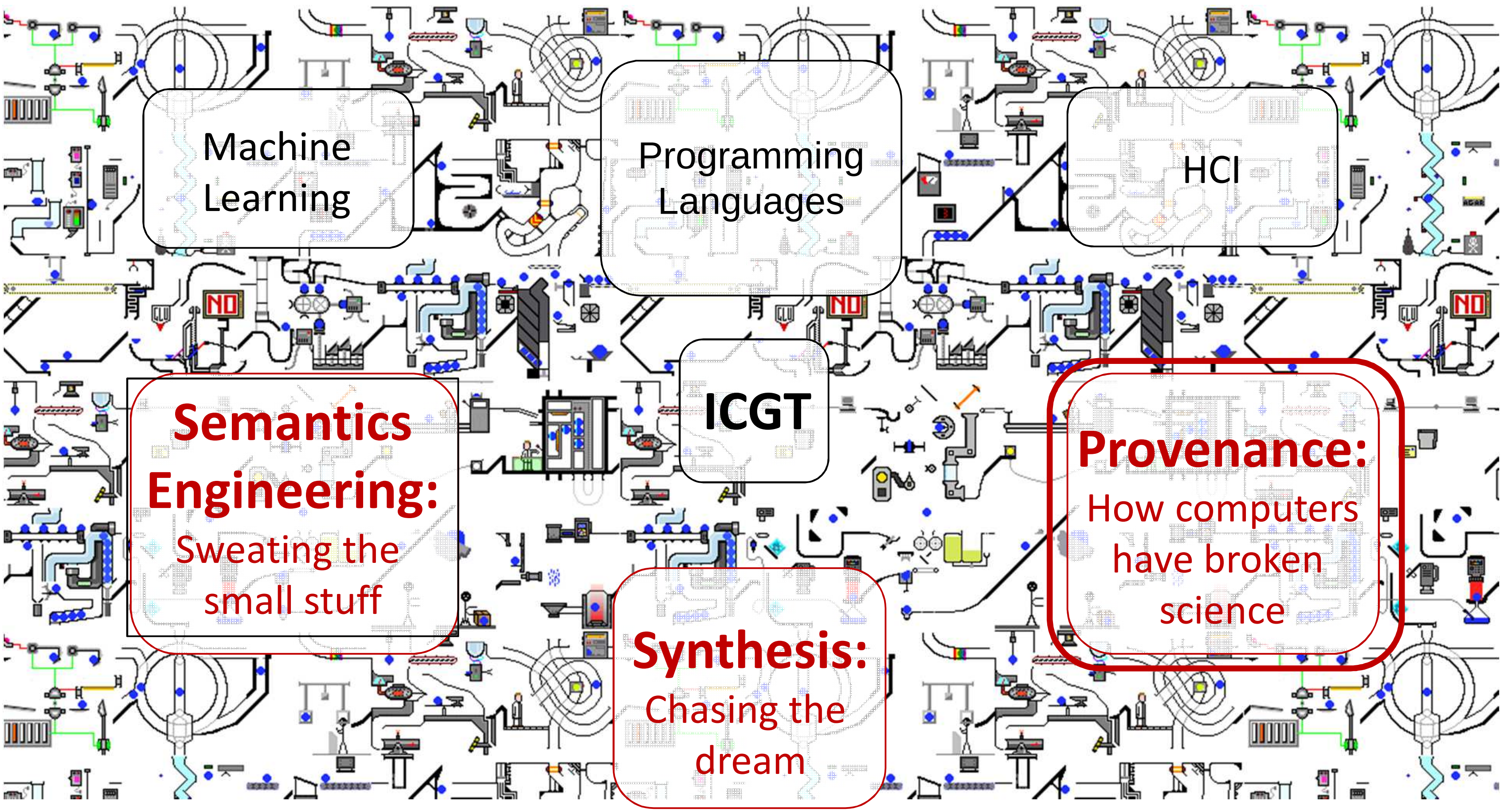
$$M \xrightarrow{T?} T(M)$$

$$\exists T.\ \forall M.\ \varphi(M, T(M))$$

Transformation implementation

$$M \xrightarrow{\exists M'.\ \varphi(M, M')} M'$$

Research Landscape is Complex, Too



http://blueballfixed.ytmnd.com/

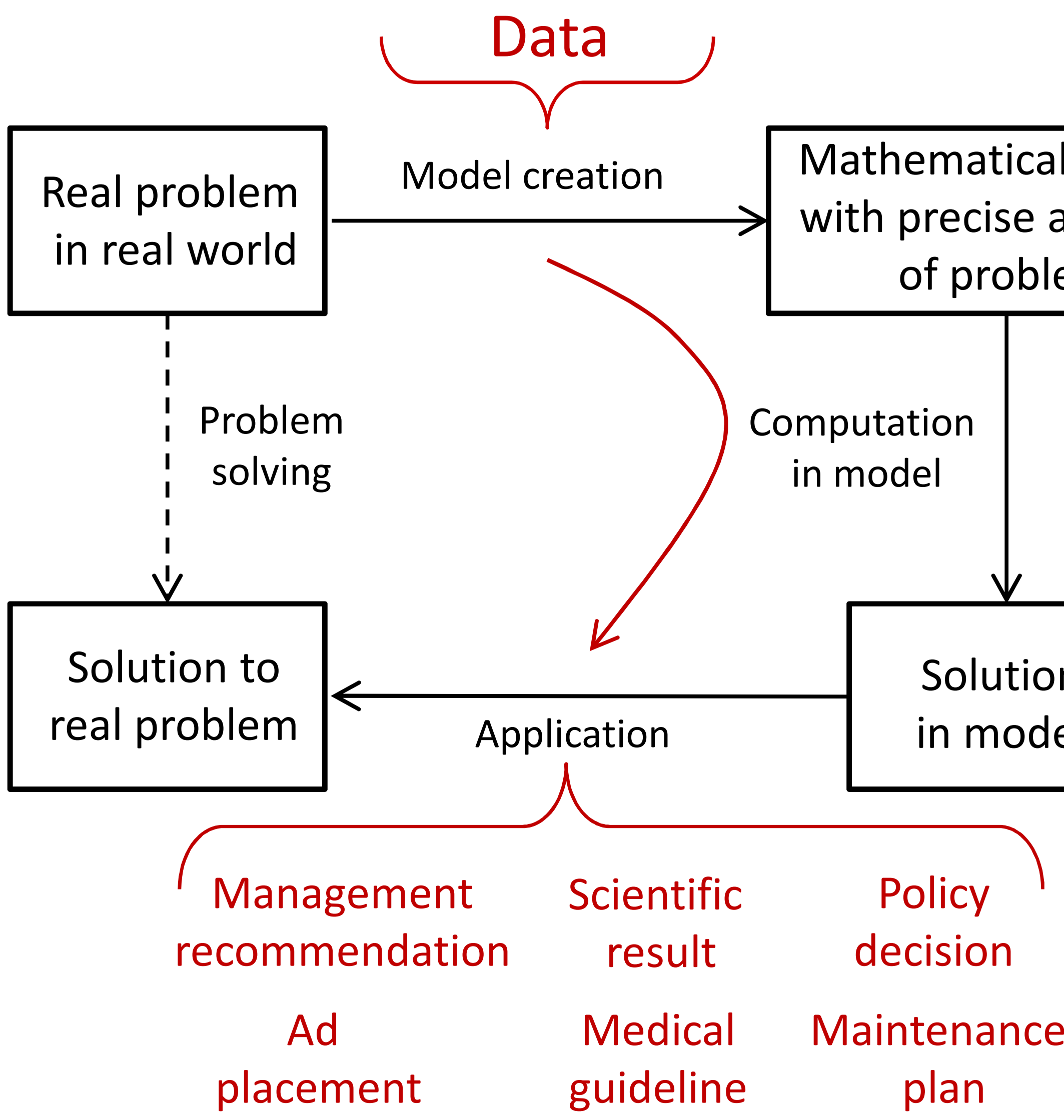
A Lot of Modern Society Relies on Hartmut's Diagram

"Executable model of gene expression"
[Fisher et al, CAV'16]

"Russian Track and Field Team Barred From Rio Olympics"
[NY Times, 06/17/16]

"CDC Concludes Zika Causes Microcephaly"
[04/13/16]

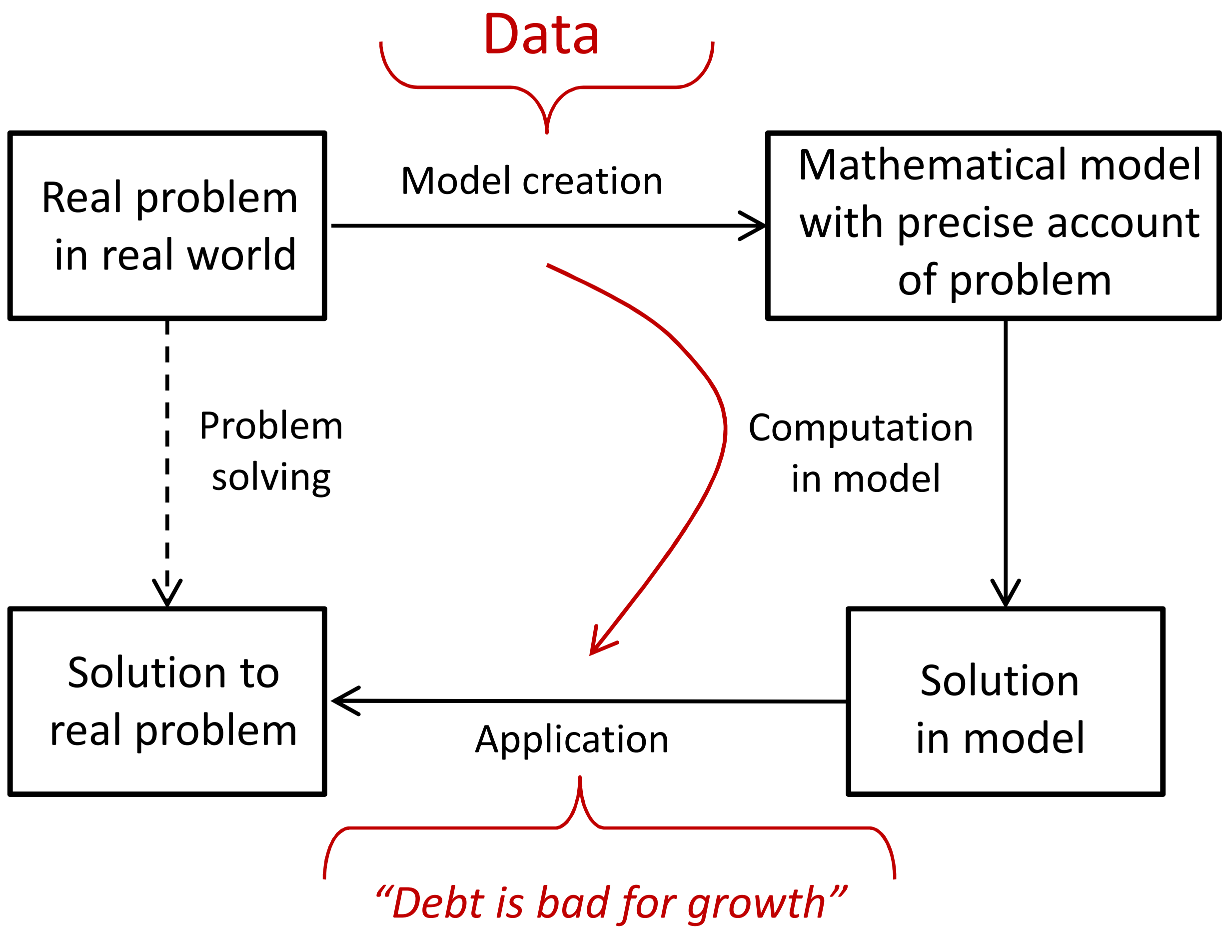
"Programs can be analyzed productively with statistical models"
[Devanbu et al, CACM'16]



But There are Problems

How good is our implementation of this process?

- Reproducibility?



Methodology flawed! [2013]

Growth in a Time of Debt

By Carmen M. Reinhart and Kenneth S. Rogoff

American Economic Review: Papers & Proceedings 100 (May 2010): 573–578

Reproducibility

Economics:

78% of 162 replication studies disconfirm major finding of original study

[Duvendack et al, 2015]

Computer systems:

Of 402 papers:

- *No or negative response: 176 (43%)*
- *Code built in less than 30mins: 130 (32%)*

[Colberg et al, CACM'16]

How computers broke science :

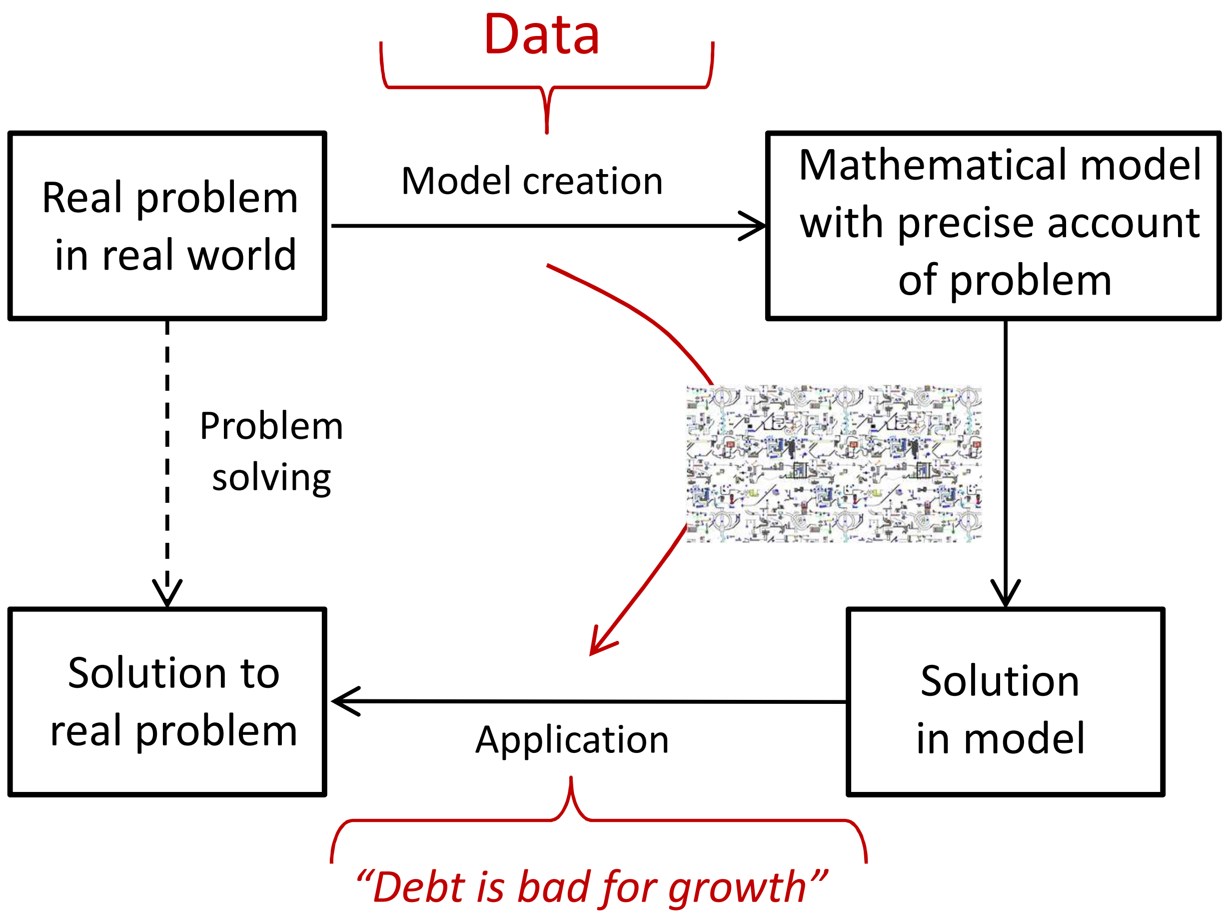
"But, since the introduction of the personal computer […] reproducibility of much research has become questionable, if not impossible.
Too much of the research process is now shrouded by the opaque use of computers that many researchers have come to depend on."

[B. Marwick, Nov 2016]

But There are Problems

How good is our implementation of this process?

- Reproducibility?
- Quality, accuracy, availability, trustworthiness, …
  of
  data, software, hardware, people, … ?



Methodology flawed! [2013]

Growth in a Time of Debt

By Carmen M. Reinhart and Kenneth S. Rogoff

American Economic Review: Papers & Proceedings 100 (May 2010): 573–578

What to Do

- **Produce better code**
  - "code for people", "add assertions", "use off-the-shelf unit testing library", "write code in the highest-level language possible", "use version control", "document design and purpose, not mechanics", "use issue tracking tool", "use pair programming"

[Wilson et al, PLoS Biology 2014]

Oh, really?

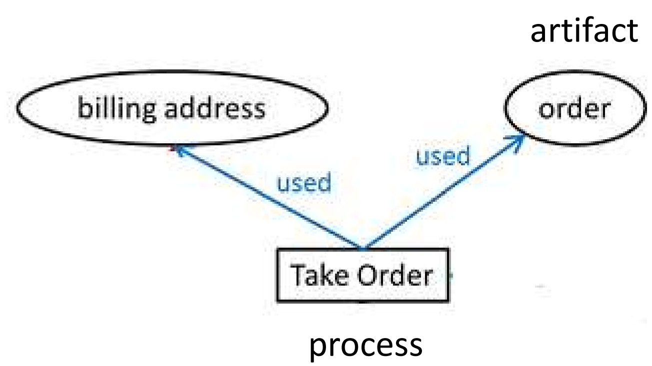
- **Open data, open formats, standards, open source sw**
  - Big topic at SC'15
  - Artifact submission & evaluation (19 CS conferences since '11)
    - STAF?, MODELS?
- **Record everything needed to**
  - recreate output (e.g., sources, workflows, versions of data, software, and hardware)
  - assess quality of relevant artifacts, processes

"Provenance"

Work in progress

Open Provenance Model (OPM)

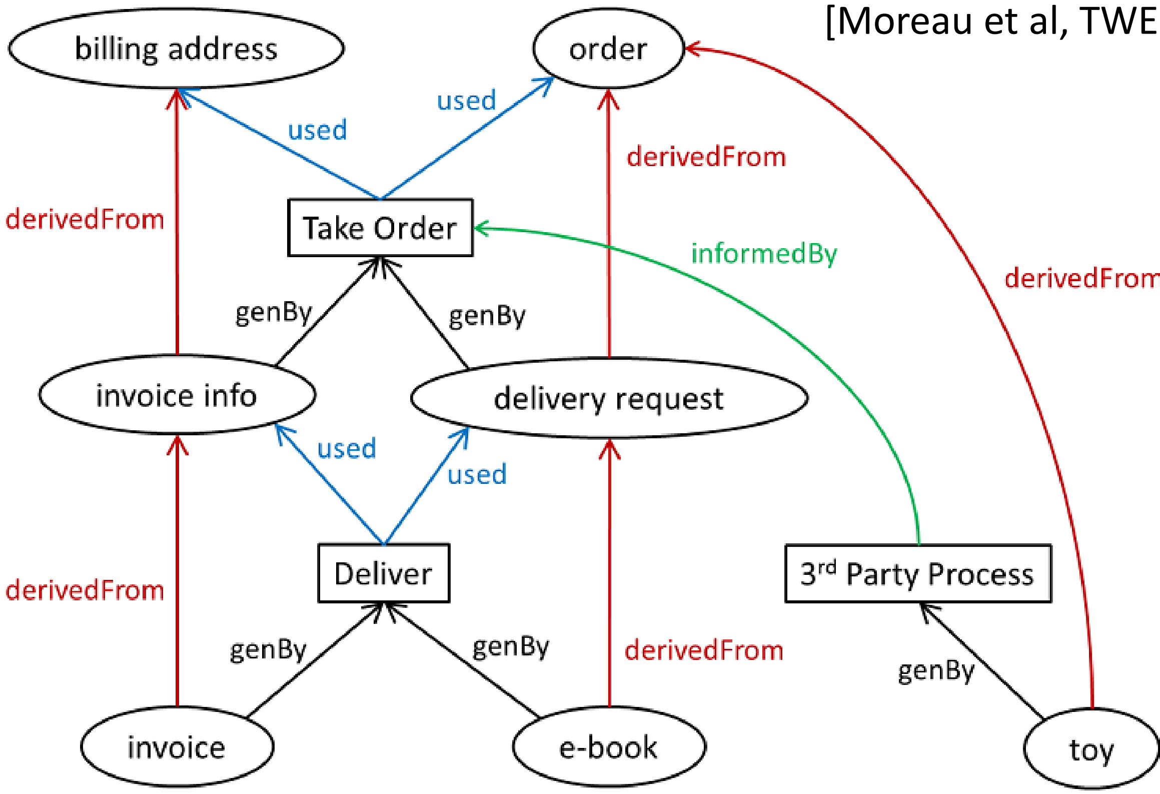
- **Metamodel**
  - Nodes
    - Process, Artifact, Roles
  - Edges
    - 4 different kinds
      - generatedBy $\subseteq$ Art $\times$ Roles $\times$ Proc
      - derivedFrom $\subseteq$ Art $\times$ Art
      - used $\subseteq$ Proc $\times$ Roles $\times$ Art
      - informedBy $\subseteq$ Proc $\times$ Proc
  - Time

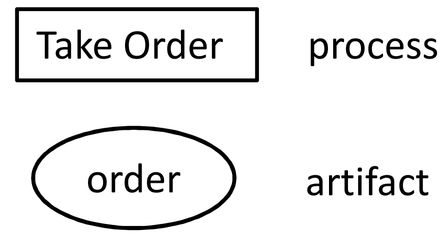


- **Semantics**

OPM graphs as temporal theories over events [Moreau et al, TWEB'15]

[Moreau et al, TWEB'15]





generatedBy $\subseteq$ Art$\times$Roles$\times$Proc
used $\subseteq$ Proc$\times$Roles$\times$Art
derivedFrom $\subseteq$ Art$\times$Art
informedBy $\subseteq$ Proc$\times$Proc

A **model** of an OPM graph G is a triple (T, $\leq$, τ), where

- T set of time points,
- $\leq$ is a partial order on T
- τ is a mapping from Events(G) to T

where

Events(G) = {begin(P),end(P) | P$\subseteq$Proc} $\cup$ {create(A) | A$\subseteq$Art} $\cup$ …

such that

$\forall$P$\in$Proc. begin(P) $\leq$ end(P),
$\forall$(A,r,P)$\in$generatedBy. begin(P) $\leq$ create(A) $\leq$ end(P),
etc

} axioms

Operations:
union,
intersection,
merge,
renaming,
refinement,
completion,
summarization,
…

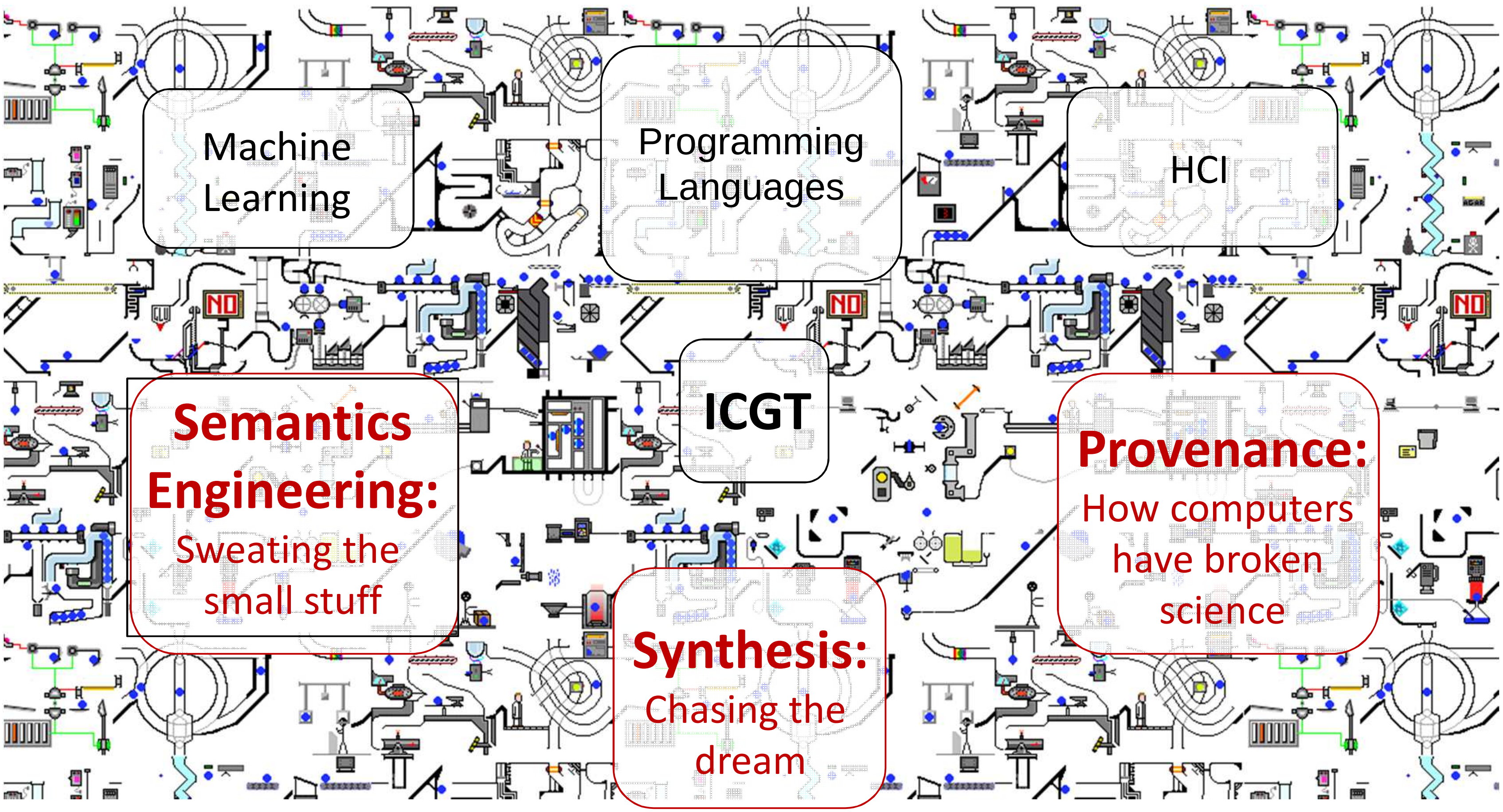
Provenance: Concluding Observations

- Will receive growing attention
- Provenance models seem 'right down our alley'
  - Implementation in GT tools?
  - Extending semantics (agents, refinement, parallelism)?
- Provenance for model transformations
  - Leverage existing work on traceability?

  $\Rightarrow$ "*Model transformations that explain their work*"?

  [Acar et al. Functional Programs that Explain their Work. ICFP'12]
- Relevance of work on
  - model management?
  - model-driven compliance?

Research Landscape is Complex, Too



http://blueballfixed.ytmnd.com/

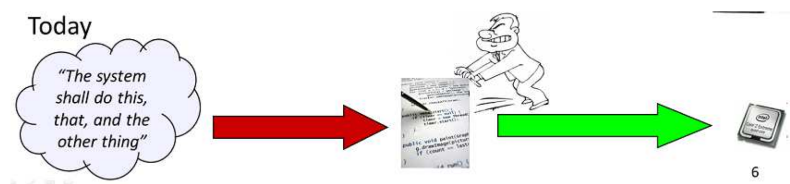
Conclusions

- **The more things change, the more they stay the same**
  - Increasing HW power $\Rightarrow$ progress, but also more complexity
  - Complexity

    vs

    abstraction, automation, analysis

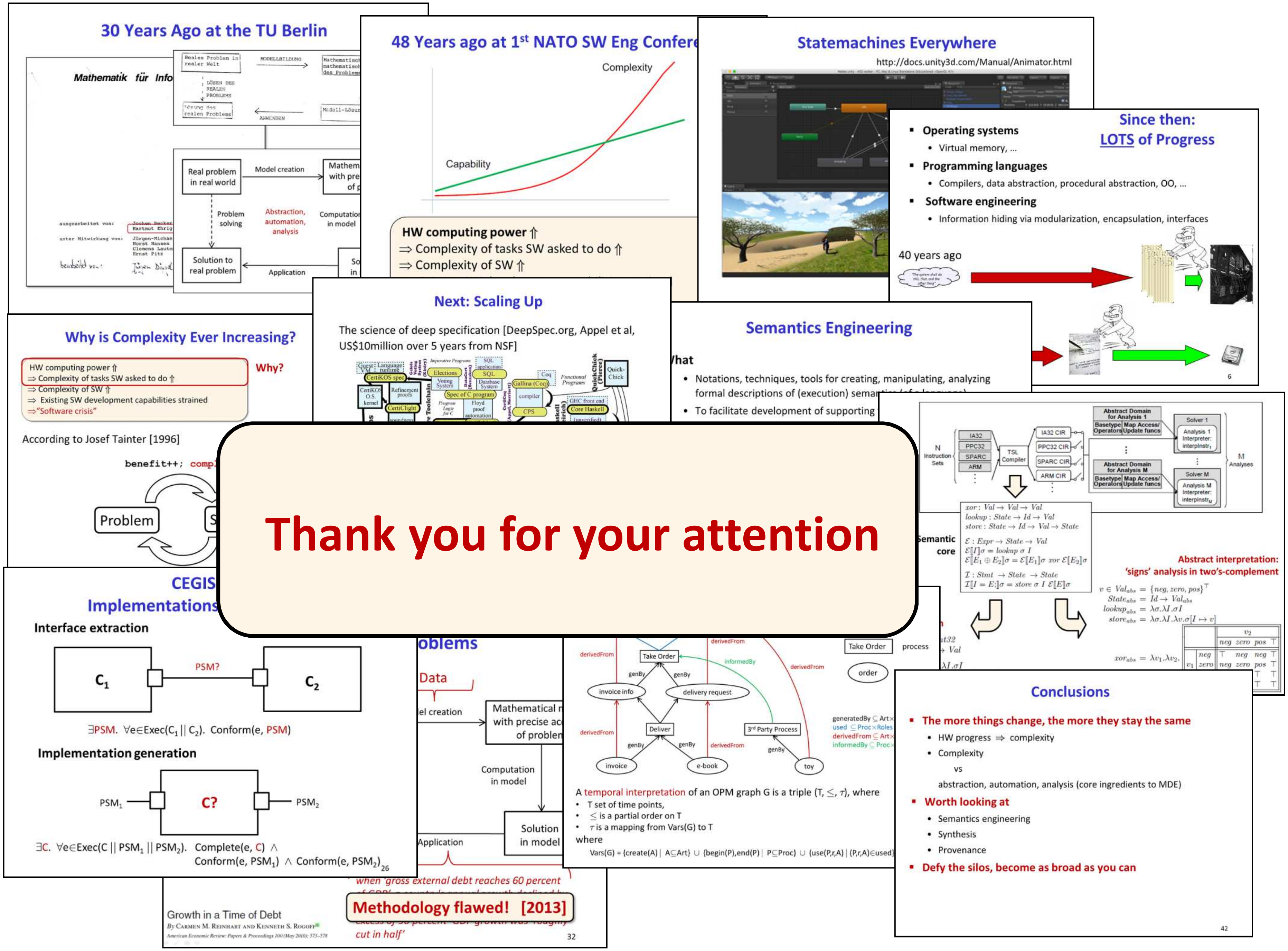
    (core ingredients not just to MDE)



- **Worth looking at**
  - Semantics engineering
  - Synthesis (see other keynotes)
  - Provenance

  } In the context of DSLs many challenges become more manageable

- **How can STAF contribute to more 'repeatable' science?**
- **Defy the silos, become as broad as you can**



Thank you for your attention
30 Years Ago at the TU Berlin
Mathematik für Info
Real problem in real world
Model creation
Problem solving
Abstraction, automation, analysis
Solution to real problem
Application
48 Years ago at 1st NATO SW Eng Confere
Complexity
Capability
HW computing power ⇑
⇒ Complexity of tasks SW asked to do ⇑
⇒ Complexity of SW ⇑
Statemachines Everywhere
http://docs.unity3d.com/Manual/Animator.html
Since then: LOTS of Progress
Operating systems
Virtual memory, …
Programming languages
Compilers, data abstraction, procedural abstraction, OO, …
Software engineering
Information hiding via modularization, encapsulation, interfaces
40 years ago
Why is Complexity Ever Increasing?
Why?
Existing SW development capabilities strained
⇒"Software crisis"
According to Josef Tainter [1996]
Problem
Next: Scaling Up
The science of deep specification [DeepSpec.org, Appel et al, US$10million over 5 years from NSF]
Semantics Engineering
Notations, techniques, tools for creating, manipulating, analyzing formal descriptions of (execution) semantics
To facilitate development of supporting
Abstract interpretation: 'signs' analysis in two's-complement
CEGIS Implementations
Interface extraction
PSM?
∃PSM. ∀e∈Exec(C1 || C2). Conform(e, PSM)
Implementation generation
C?
∃C. ∀e∈Exec(C || PSM1 || PSM2). Complete(e, C) ∧ Conform(e, PSM1) ∧ Conform(e, PSM2)
Methodology flawed! [2013]
Growth in a Time of Debt
A temporal interpretation of an OPM graph G is a triple (T, ≤, τ), where
T set of time points,
≤ is a partial order on T
τ is a mapping from Vars(G) to T
Conclusions
The more things change, the more they stay the same
HW progress ⇒ complexity
Complexity vs abstraction, automation, analysis (core ingredients to MDE)
Worth looking at
Semantics engineering
Synthesis
Provenance
Defy the silos, become as broad as you can