Assurance Cases and Software: Is there any evidence[1]?

1: Apologies to John McDermid for stealing from "Software Safety: Where is the evidence?"

Mats P. E. Heimdahl

University of Minnesota Software Engineering Center
Department of Computer Science and Engineering
University of Minnesota
4-192 EE/CS; 200 Union Street SE
Minneapolis, MN 55455

UNIVERSITY OF MINNESOTA

Software Engineering Center

Domains of Concern

Regulation and Approval Today

Process Based Standards

1. Follow these steps
2. Produce these documents
3. Hope for the best

Does Current Regulation Work?

It Is Not Working (as well as it could)

- Do not necessarily lead to desired quality
  - Aircraft accidents and mishaps that should not happen
  - Excessive number of medical device recalls
  - Security breaches are rampant
- Rigid standards inhibit adoption of new tools and techniques
- Questionable correlation between prescribed activities and failure rate
- Very costly?

It Clearly Helps

- Certification and approval promotes a "quality" culture
  - Helps justify the cost
  - Balances "get it done" with "get it right"
- Enforces rigorous process
  - But limits innovation
- Self selection of engineers and developers

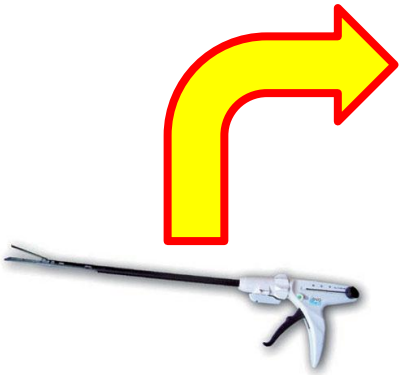
- **It is the culture, not the standard or regulation, that produces quality products**

Recent Reports

- **Software for Dependable Systems: Sufficient Evidence?**
  - Daniel Jackson, Martyn Thomas, and Lynette I. Millett, Editors, Committee on Certifiably Dependable Software Systems, National Research Council.
- **Medical Devices and the Public's Health: The FDA 510(k) Clearance Process at 35 Years**
  - Committee on the Public Health Effectiveness of the FDA 510(k) Clearance Process: Institute of Medicine.

FDA 510(k) Process

- Demonstrate that your new device is “substantially equivalent” to a previous predicate device already on the market



Certification

- *The process of assuring that a product or process has certain stated properties, which are then recorded in a certificate.*
  - Certification usually involves assurance by an independent party, although the term is also used analogously for customer (second-party) and developer (first-party) assurance.

Adopted from NRC Report:
Software for Dependable Systems: Sufficient Evidence?

Goals

- Explicit claims of dependability
  - A system cannot exhibit all desi
    under all conditions; be
    assumptions, envi
- Based on
- P
- acilitate innovation

Claim, Evidence, and Argument

- Explicit Claims
  - State explicitly what properties (safety, security, reliability, performance, etc.) the system must possess and under which assumptions
- Supporting Evidence
  - Results of observing, analysing, testing, simulating and estimating the properties of a system that provide the fundamental information from which safety can be inferred
- High Level Arguments
  - Explanation of how the available evidence can be reasonably interpreted as indicating acceptable dependability

Argument without Evidence is **unfounded**

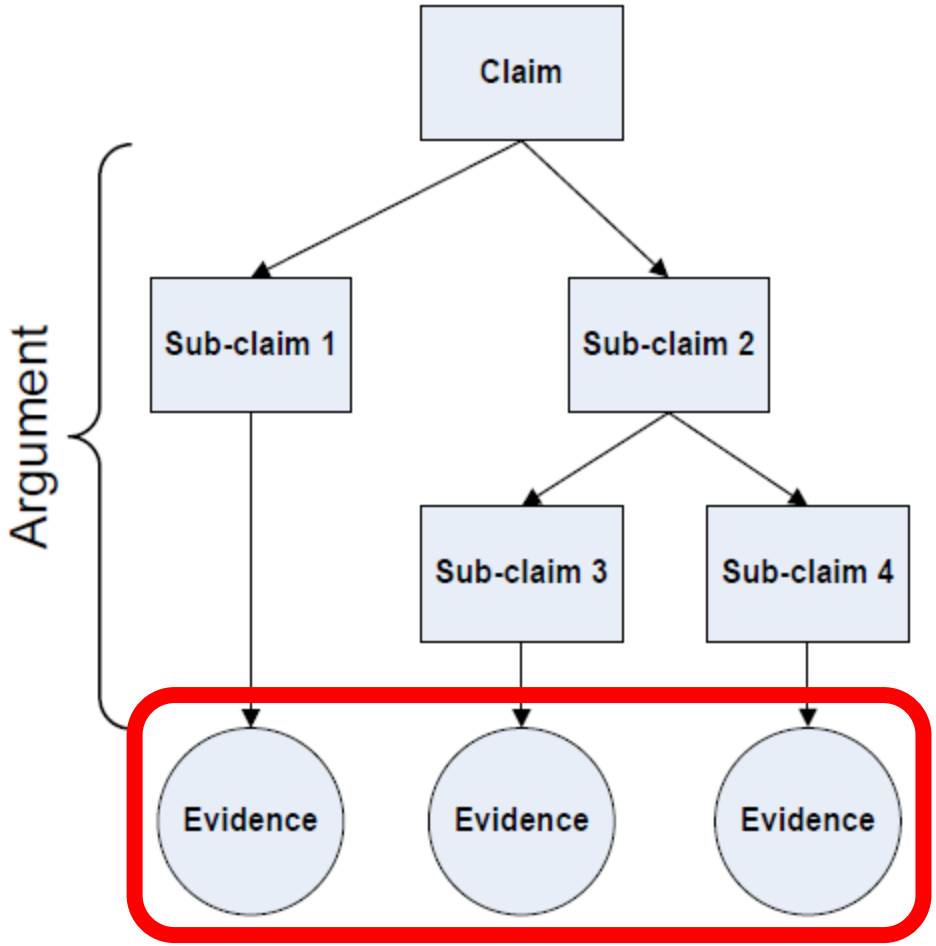
Evidence without Argument is **unexplained**

- Tim Kelly, 2008

Assurance Cases

To construct an assurance case we need to:

- make an explicit set of claims about the system
- produce the supporting evidence
- provide a set of arguments that link the claims to the evidence
- make clear the assumptions and judgments underlying the arguments
- allow different viewpoints and levels of detail.



McDermid:

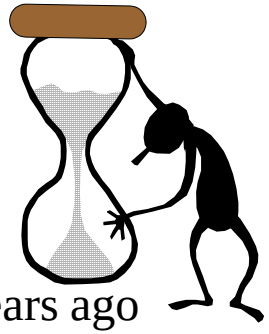
"Software Safety: Where is the evidence?"

- Bring the Evidence!!
- What Evidence????

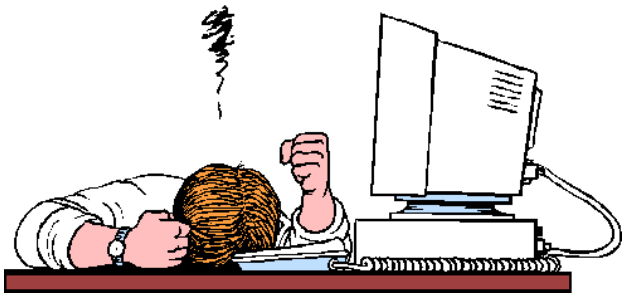
Software meets its safety requirements

1. ~~Inspection~~
2. Testing
3. Formal Verification

What About Testing??



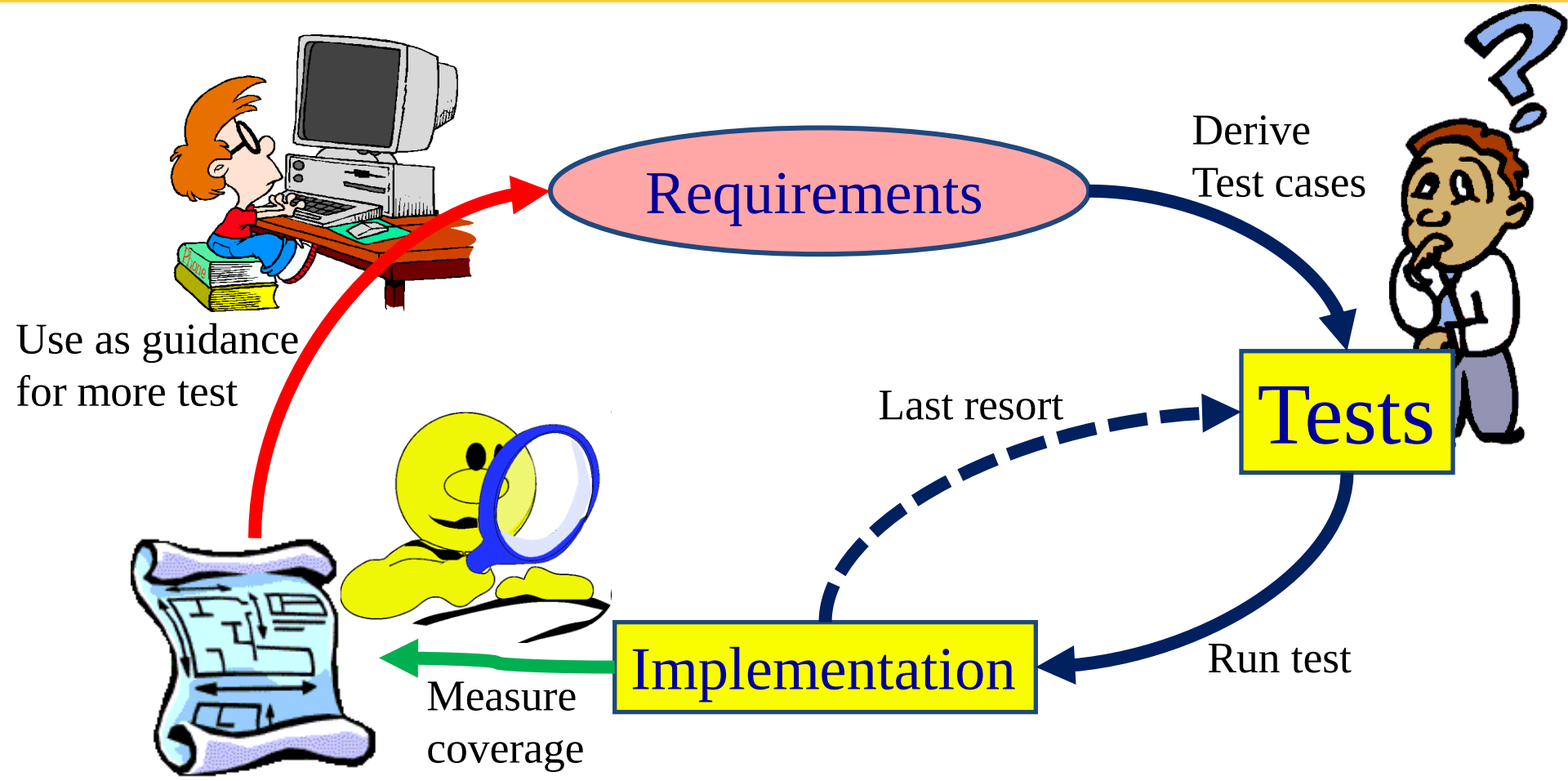
- Statistical Testing
  - **Does not work**
    - Butler and Finelli 20 years ago
    - R. W. Butler and G. B. Finelli. "The Infeasibility of Quantifying the Reliability of Life-Critical Real Time Software"



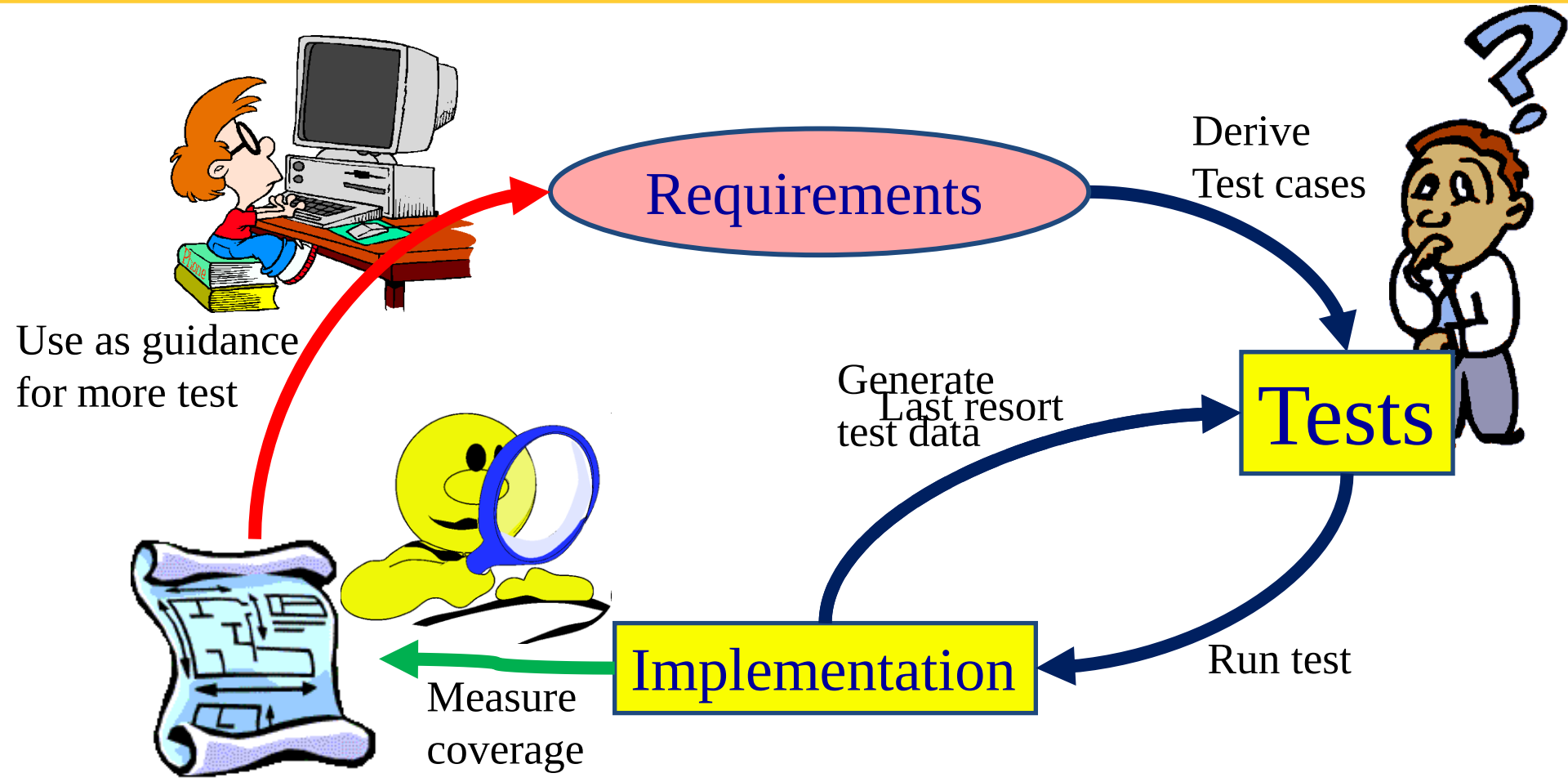
- Coverage Criteria
  - **Does not work** (yet)
    - As will be shown

- Engineering Judgment
  - Assisted by coverage measures
  - Not objective!!!

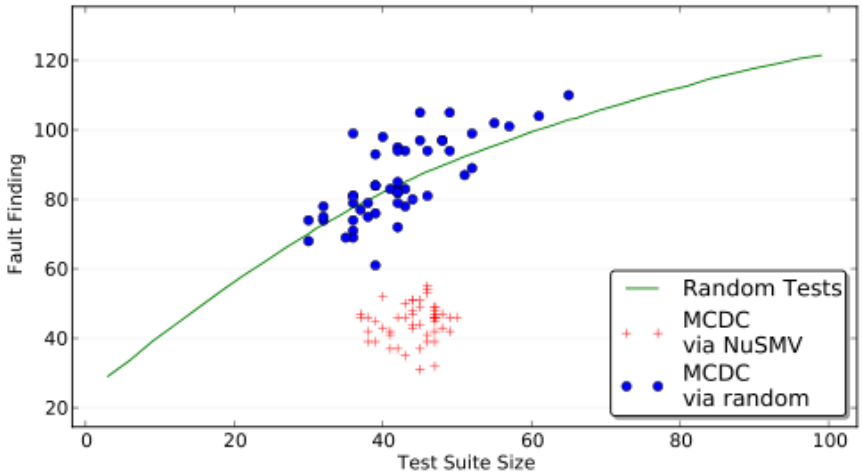
MCDC as Intended in DO-178B



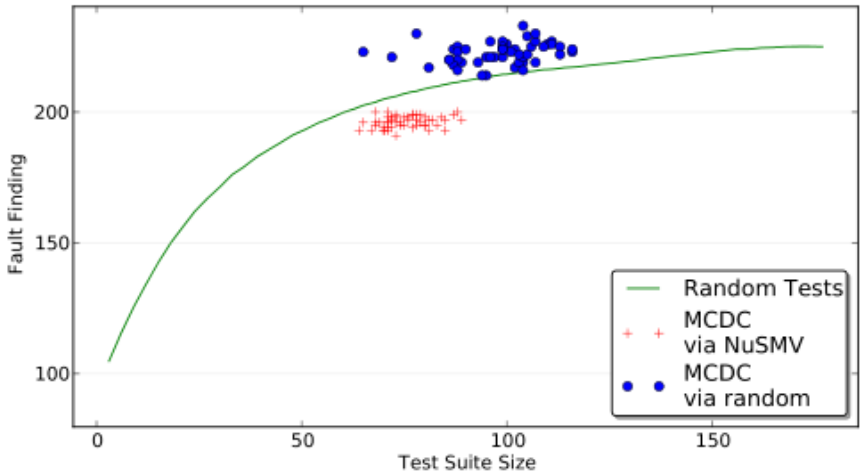
MCDC with Automation



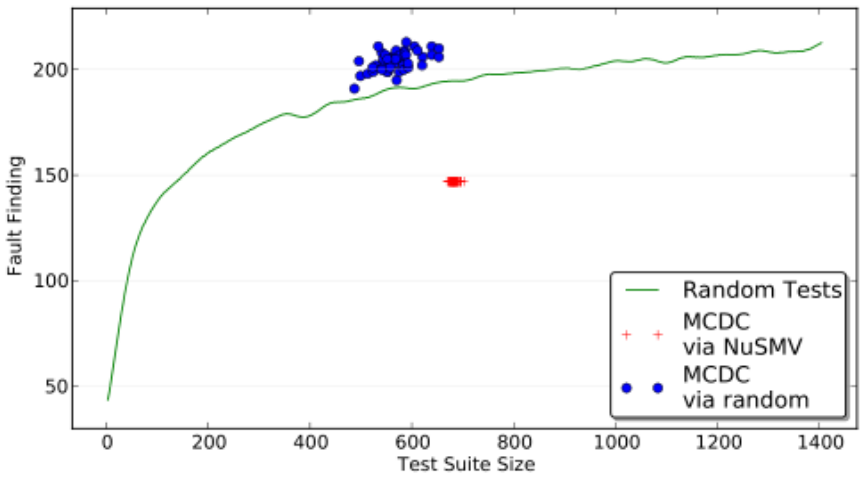
MCDC Effectiveness is Poor



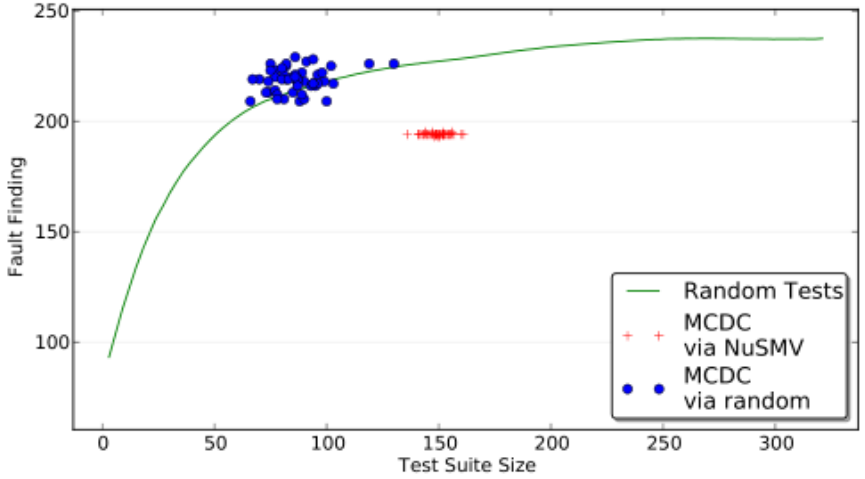
(a) DWM_1: MCDC



(b) DWM_2: MCDC

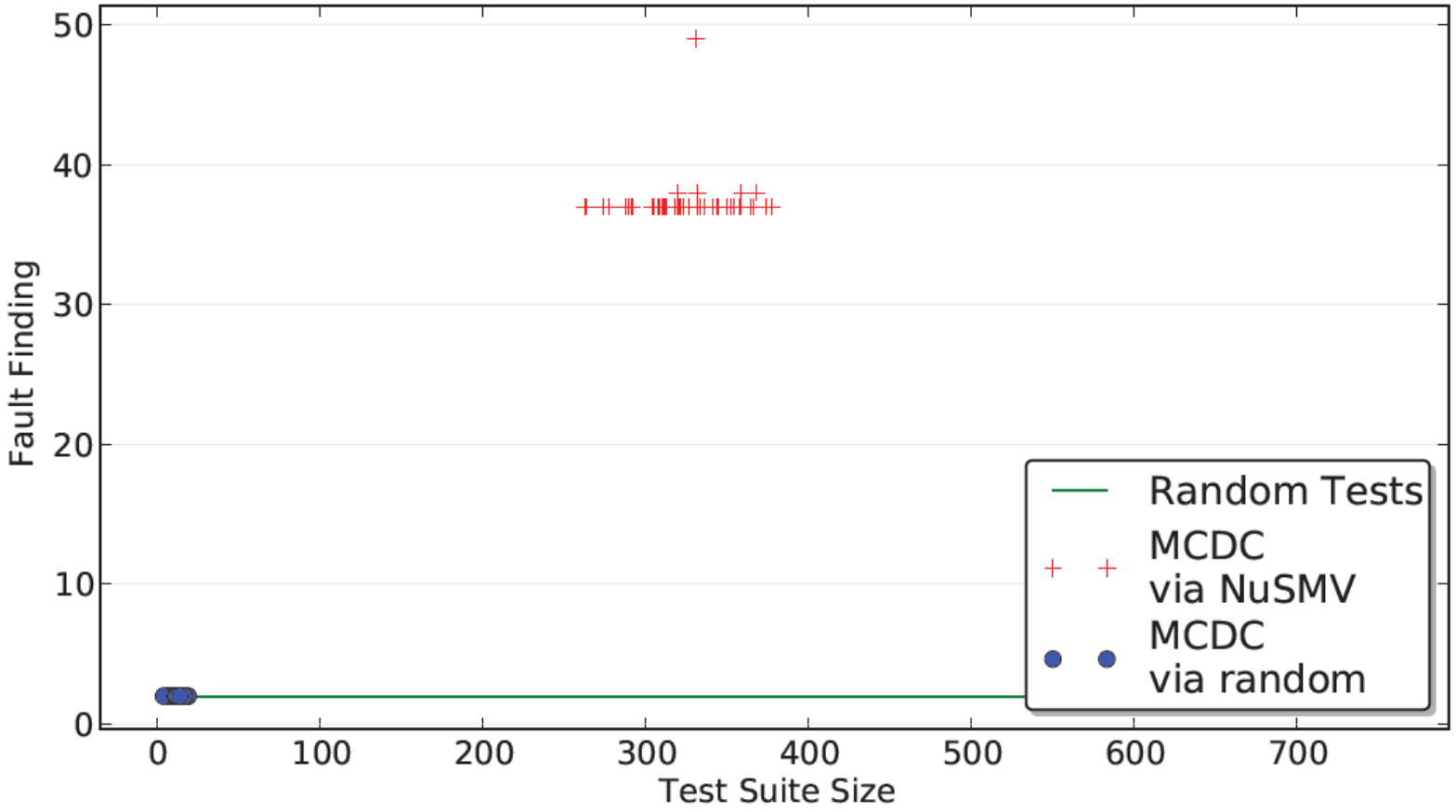


(c) Vertmax_Batch: MCDC



(d) Latctl_Batch: MCDC

Except When it is Not

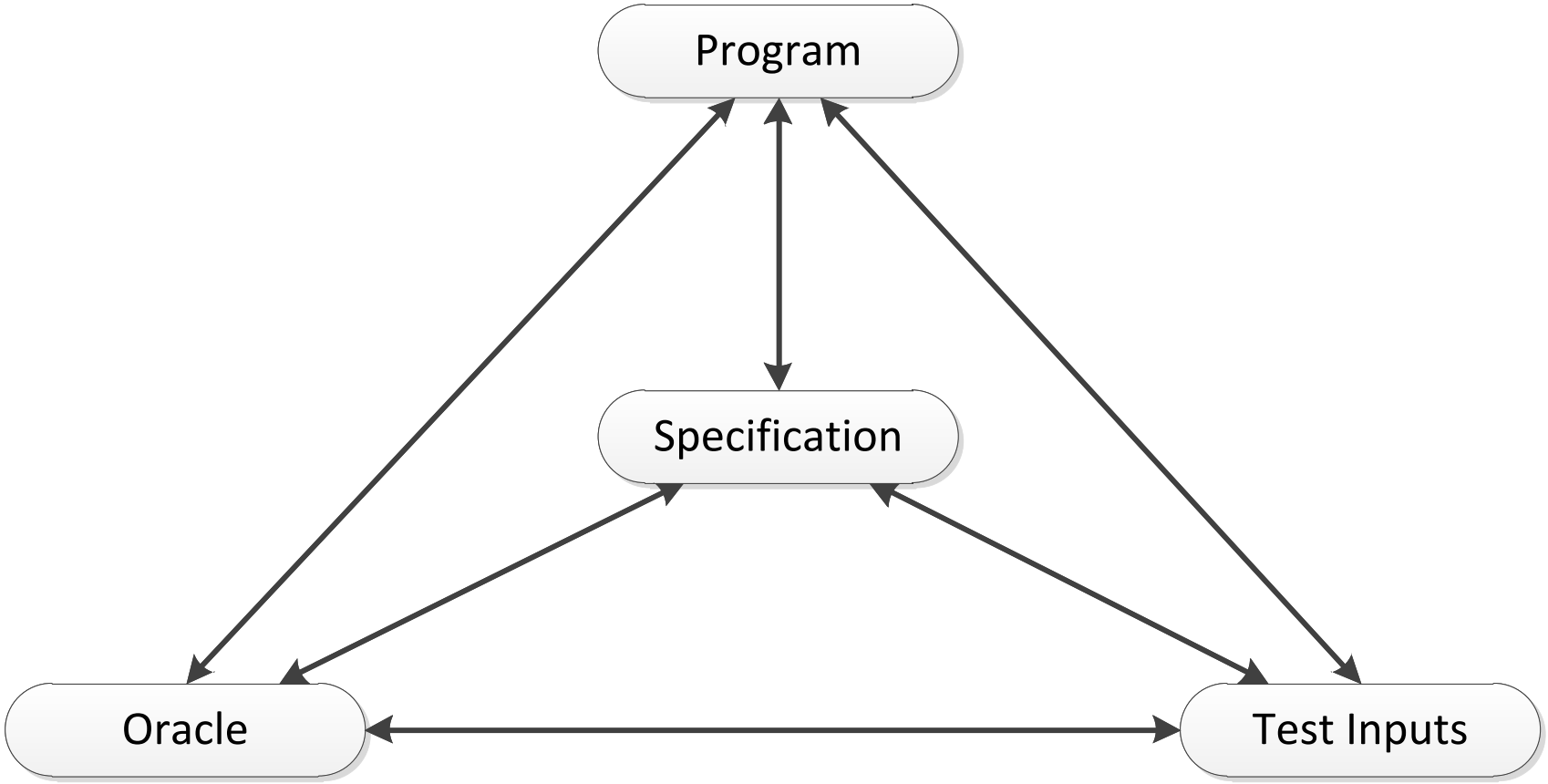


Effect of Program Structure of # Faults Found (MCDC)

Oracle	IV			Outputs Only		
Inline Level	NonInlined	Inlined	Rel. Imp.	NonInlined	Inlined	Rel. Imp.
DWM_1	79.9%	87.9%	10.0%	69.1%	82.5%	19.4 %
DWM_2	63.7%	86.1%	35.2%	56%	84.6%	51.8%
DWM_3	5.7%	90.6%	1489%	1.6%	90.6%	5940%
Latctl_Batch	69.3%	86.5%	24.8%	60.1%	79.2%	32.9%
Vertmax_Batch	76.7%	85.5%	11.5%	75.9%	84.7%	11.6%
WBS	77.3%	77.4%	0.1%	55.4%	56.3%	1.6%
Sensor Voting	28.4%	33.3%	17.6%	25.9%	30.9%	19.3%

Table VI. Percentage of mutants caught by reduced inlined and non-inlined test suites.

Testing: We Do Not Know What We Are Doing
Testing Artifacts - Relationships

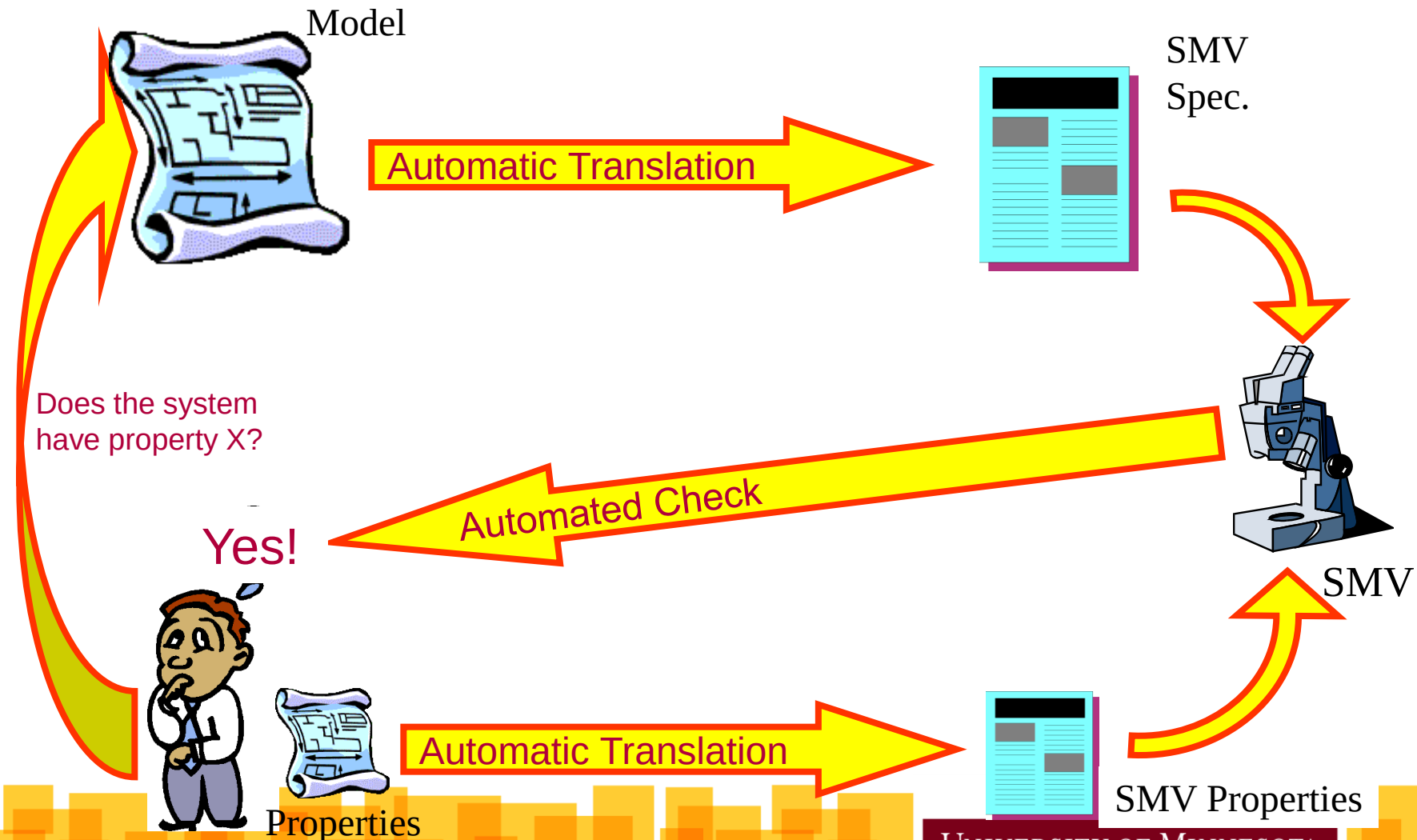


Matt Staats, Michael W. Whalen, and Mats P.E. Heimdahl.
Programs, Tests, and Oracles: The Foundations of Testing Revisited.

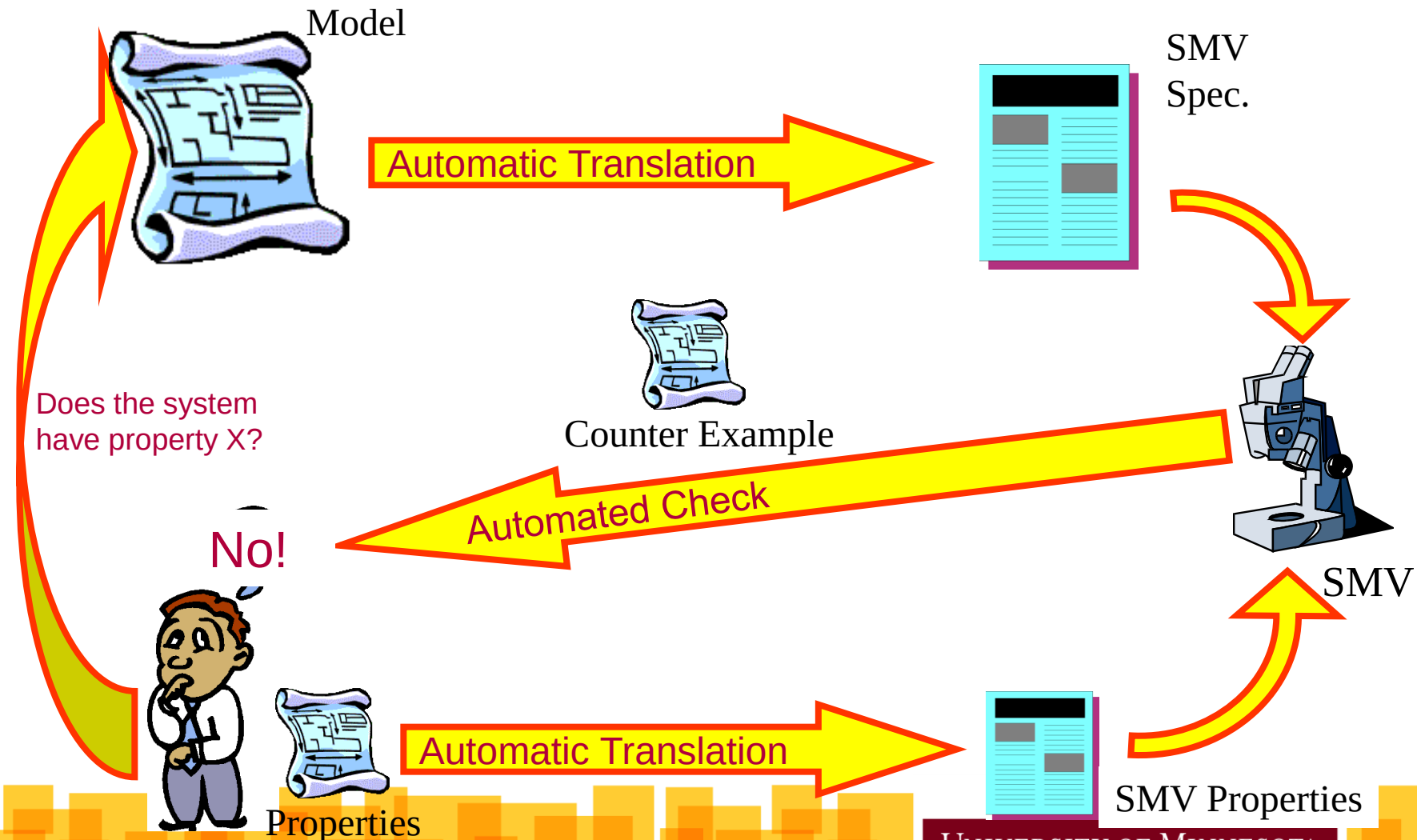
What About Formal Verification?

- We can mathematically prove that our program satisfies the requirements
  - Requirement R is satisfied in model M
    - M models R: $M \models R$
  - Rarely the case
    - R is satisfied in M when M is running in the environment E
    - $M \wedge E \models R$

Model Checking Process



Model Checking Process



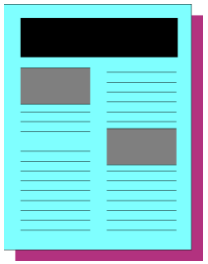
Why?

Guru

Model

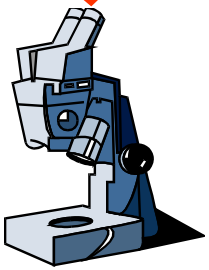
Automatic Translation

SMV Spec.



?

Out to Lunch



SMV

Does the system have property X?

Engineer

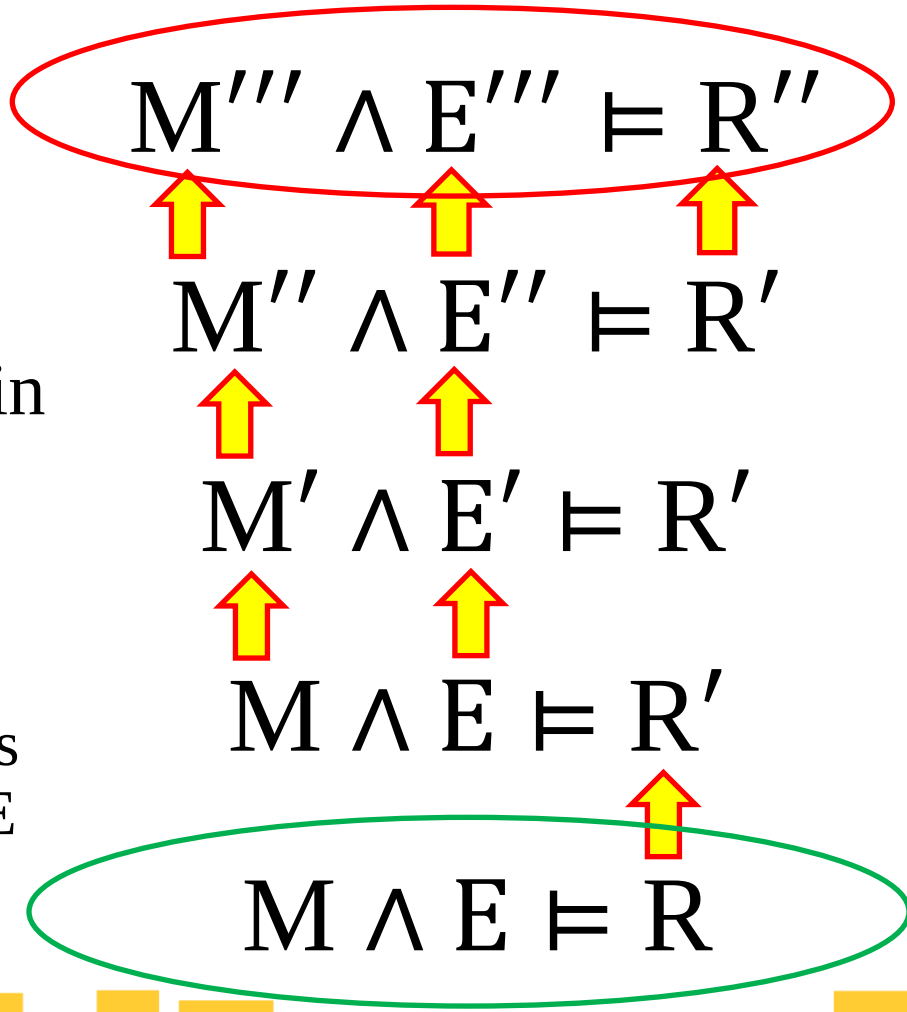
Properties

Automatic Translation

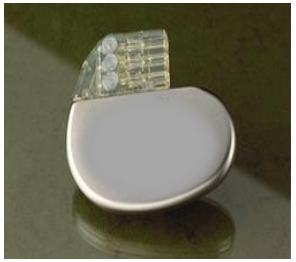
SMV Properties

What About Formal Verification?

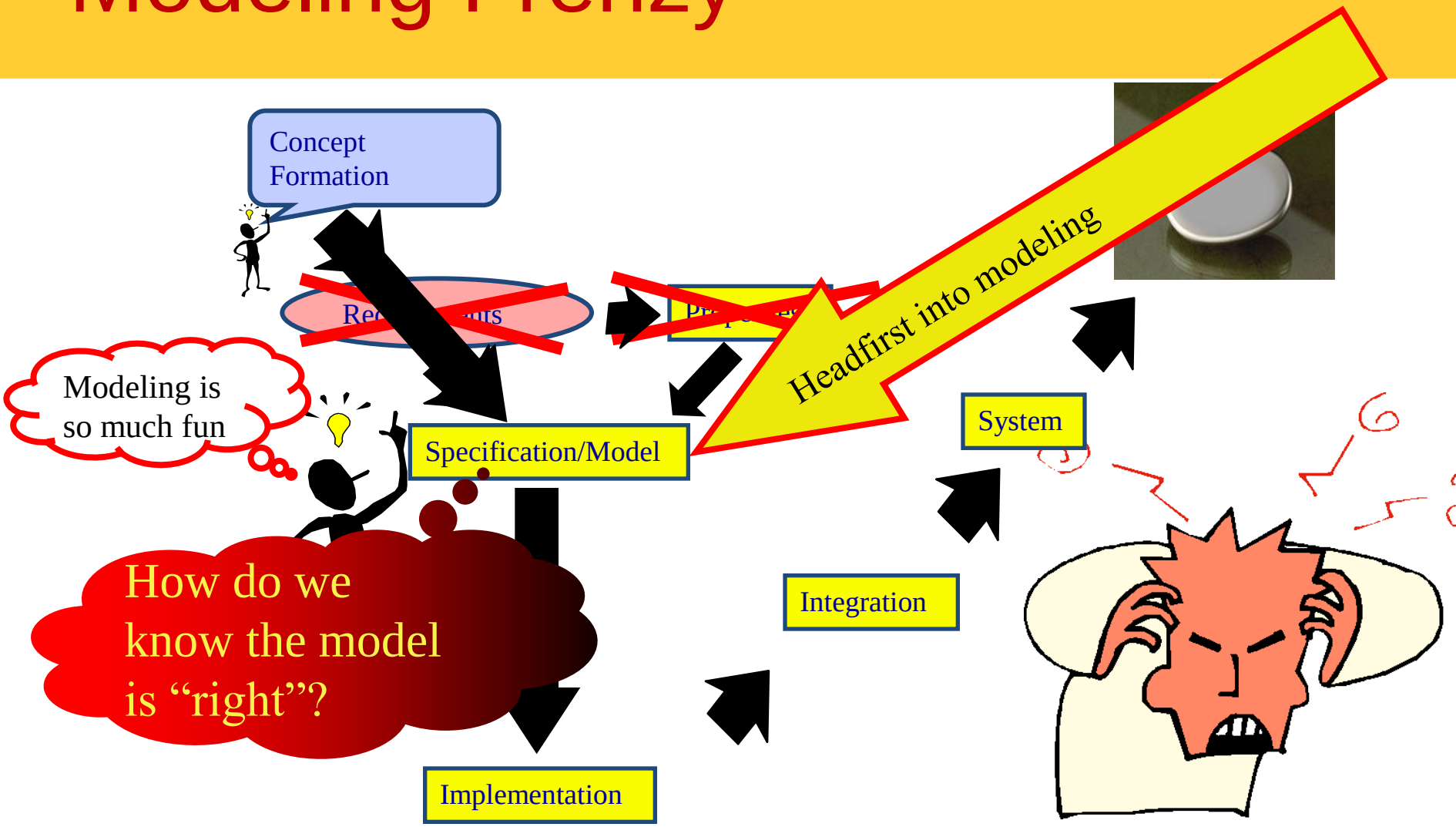
- We can mathematically prove that our program satisfies the requirements
  - Requirement R is satisfied in model M
    - M models R: $M \models R$
  - Rarely the case
    - R is satisfied in M when M is running in the environment E
    - $M \wedge E \models R$



How we **Will** Develop Software (in theory)



Modeling Frenzy



Inappropriate Evidence

- Even perfect tools used inappropriately will harm you
  - Testing tools to generate inappropriate and/or useless tests
  - Verification with inappropriate abstractions, simplifications, and assumptions
- Loss of *collateral validation and verification*
  - Much validation and verification takes place by engineers working hard
    - How much? Nobody knows…

So, What Do We Do?

- Back to basic system safety engineering!!!
  - Design hazards out of your systems
- Automation key to productivity and dependability
  - I am a big supporter of tools and automation
  - There is still a long way to go
  - Improper tool use could be catastrophic
- Fundamental testing research needed
  - Robust test adequacy metrics
  - Understand relationships between development artifacts
- Verification support
  - IVE: Integrated Verification Environments
  - Good training materials
    - Verification methodologies

Infusion Pump

- When the stop button is pressed, the current pump stroke shall be completed prior to stopping the pump.

- We could verify in our software, or…

Not Actual Device

So, What Do We Do?

- Back to basic system safety engineering!!!
  - Design hazards out of your systems
- Automation key to productivity and dependability
  - I am a big supporter of tools and automation
  - There is still a long way to go
  - Improper tool use could be catastrophic
- Fundamental testing research needed
  - Robust test adequacy metrics
  - Understand relationships between development artifacts
- Verification support
  - IVEs: Integrated Verification Environments
  - Good training materials
    - Verification methodologies

Discussion