A Science of Programming Language Design?

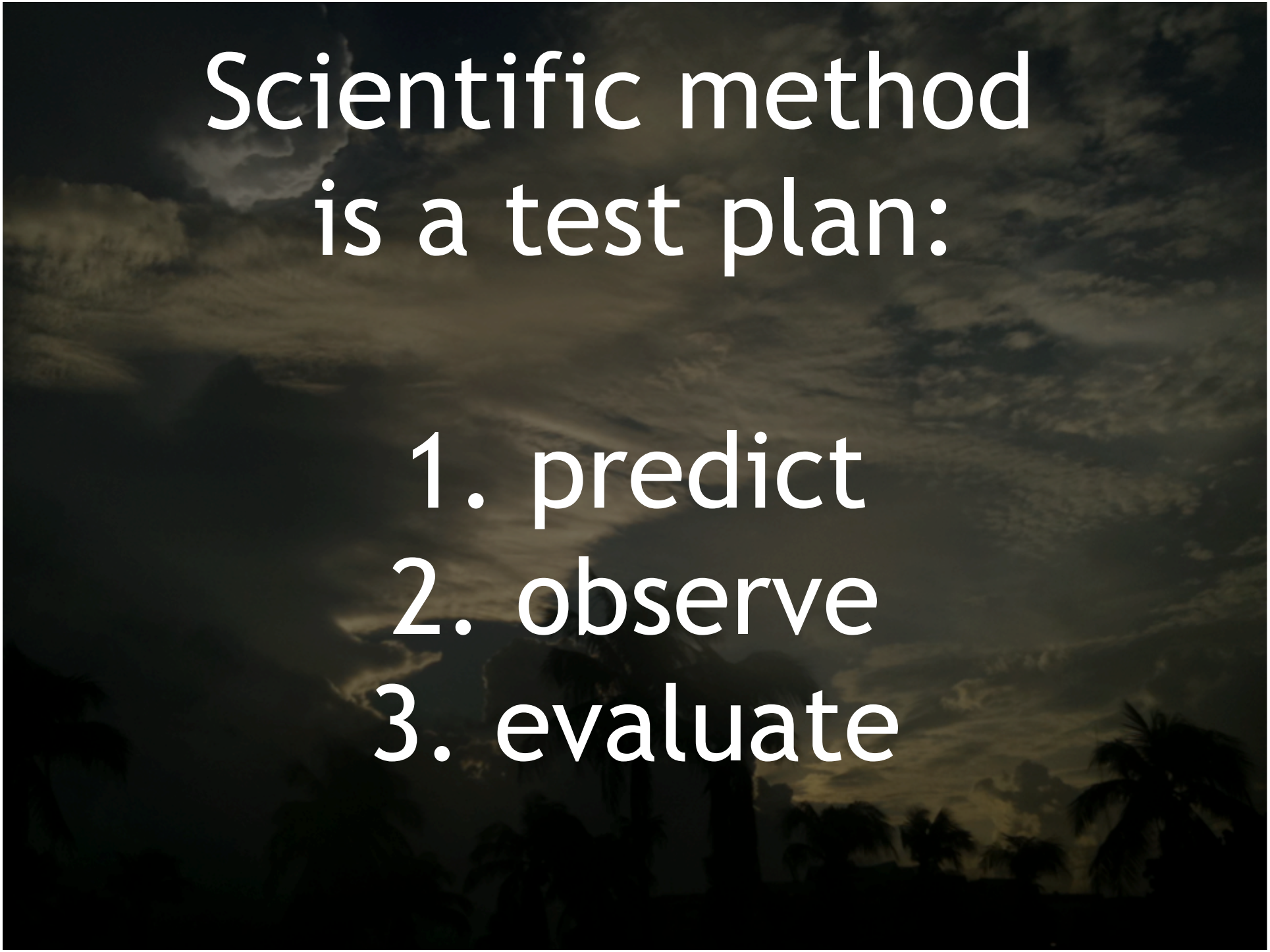
William Cook, UT Austin

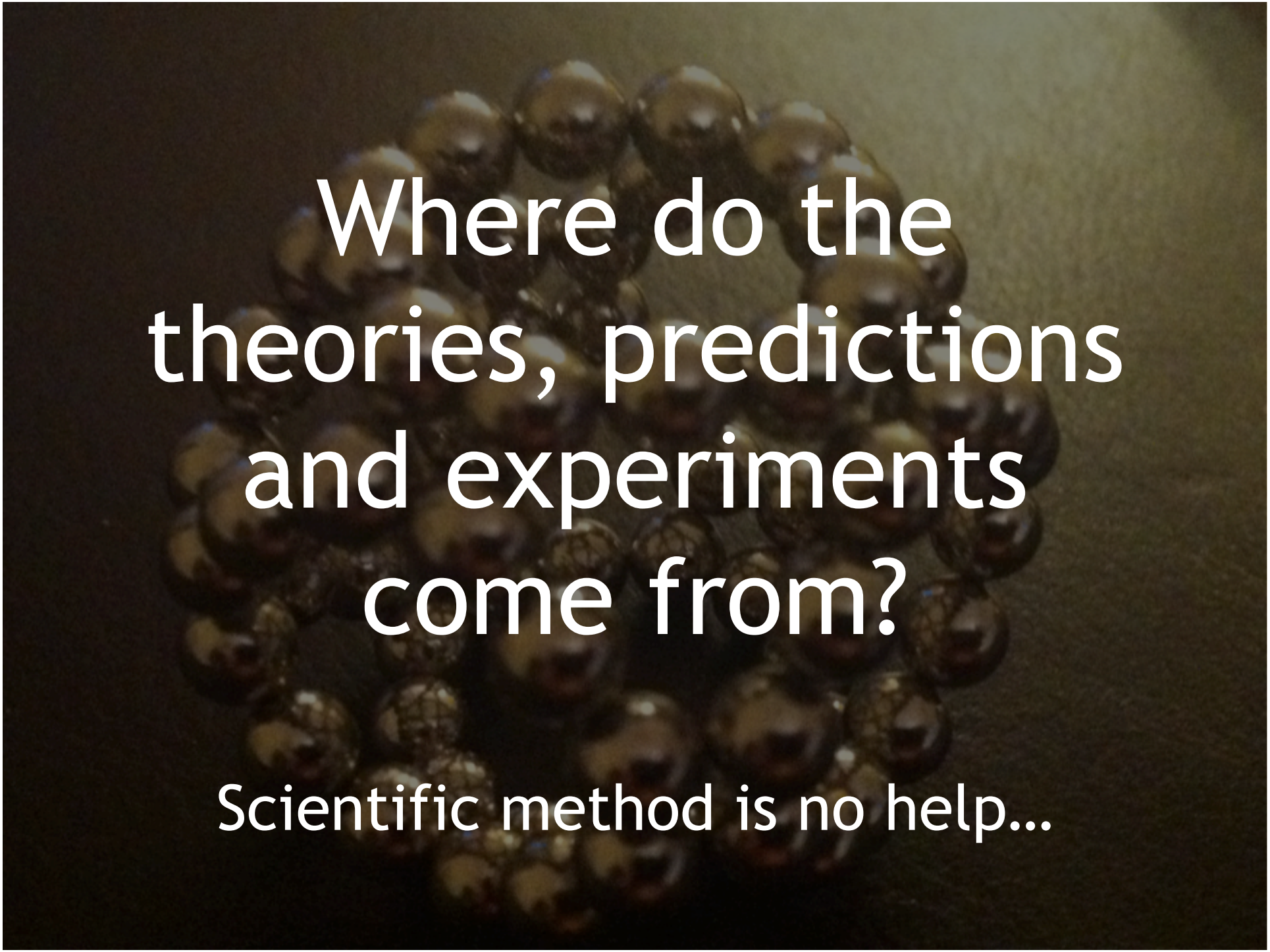
PLATEAU 2012

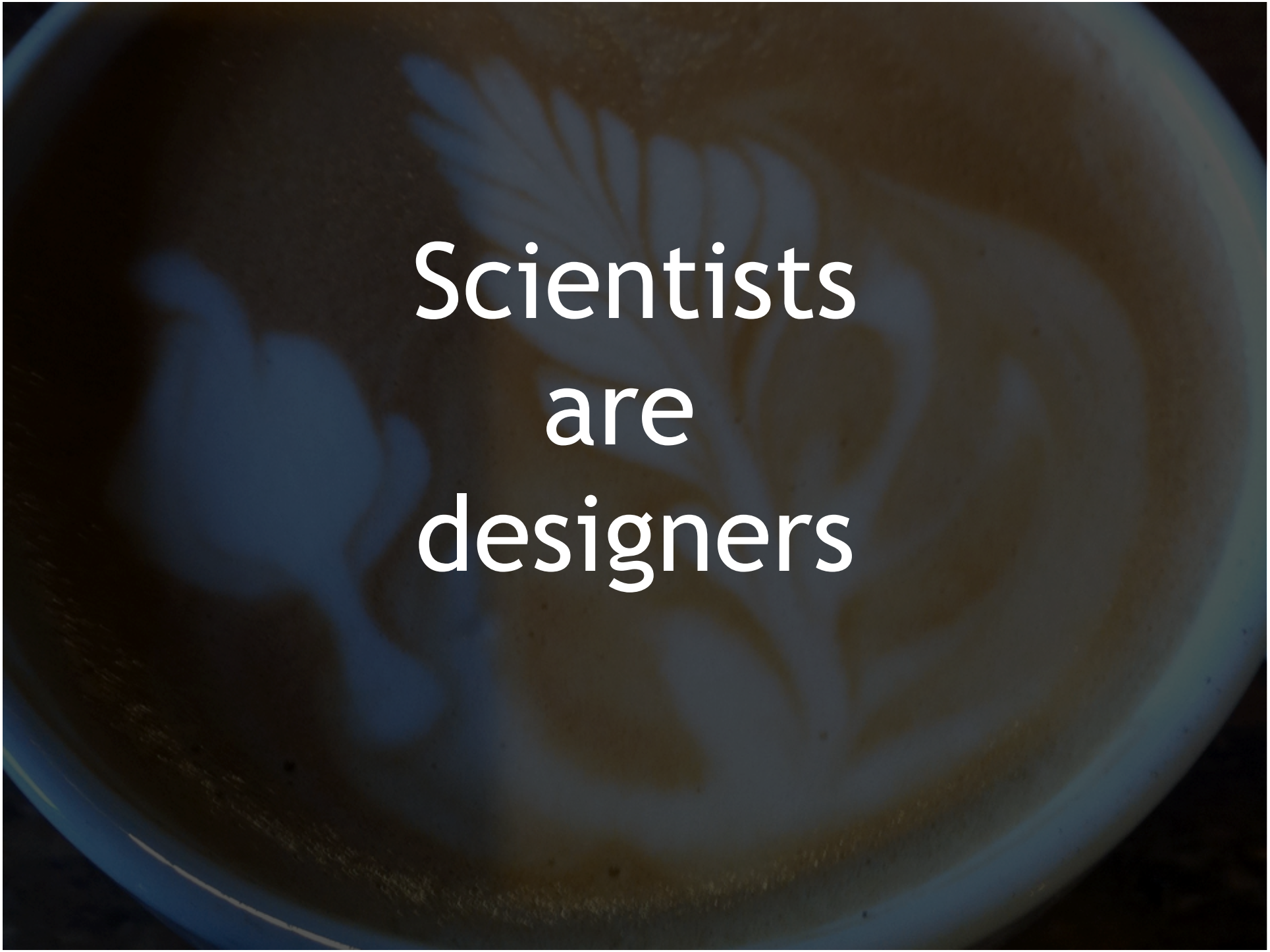

Science
create and evaluate
testable models


Design
create artifact satisfying need or desire


Science
testable model
Design
satisfy desire

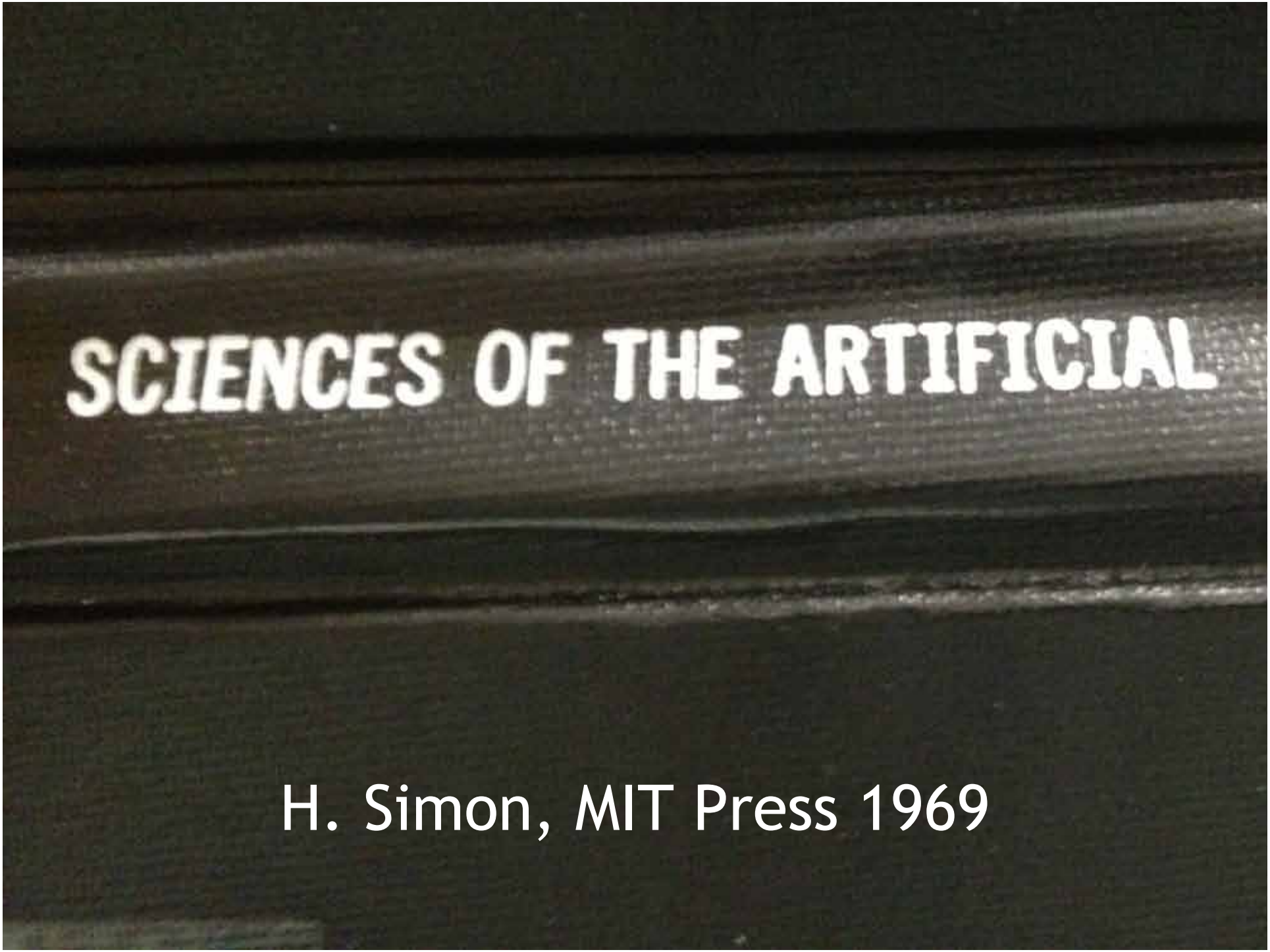

Scientific method
is a test plan:
1. predict
2. observe
3. evaluate


Where do the theories, predictions and experiments come from?
Scientific method is no help…

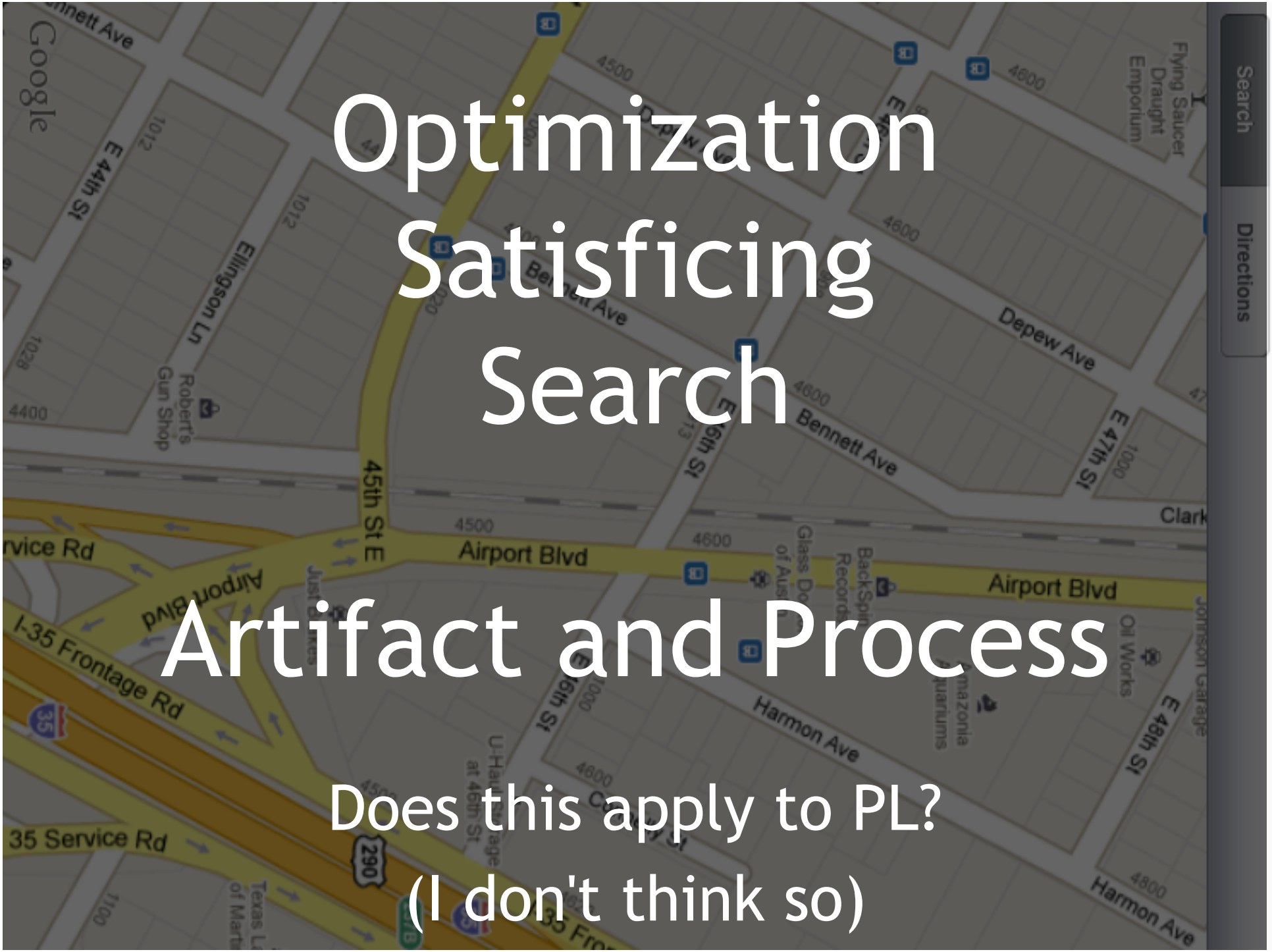

Scientists
are
designers

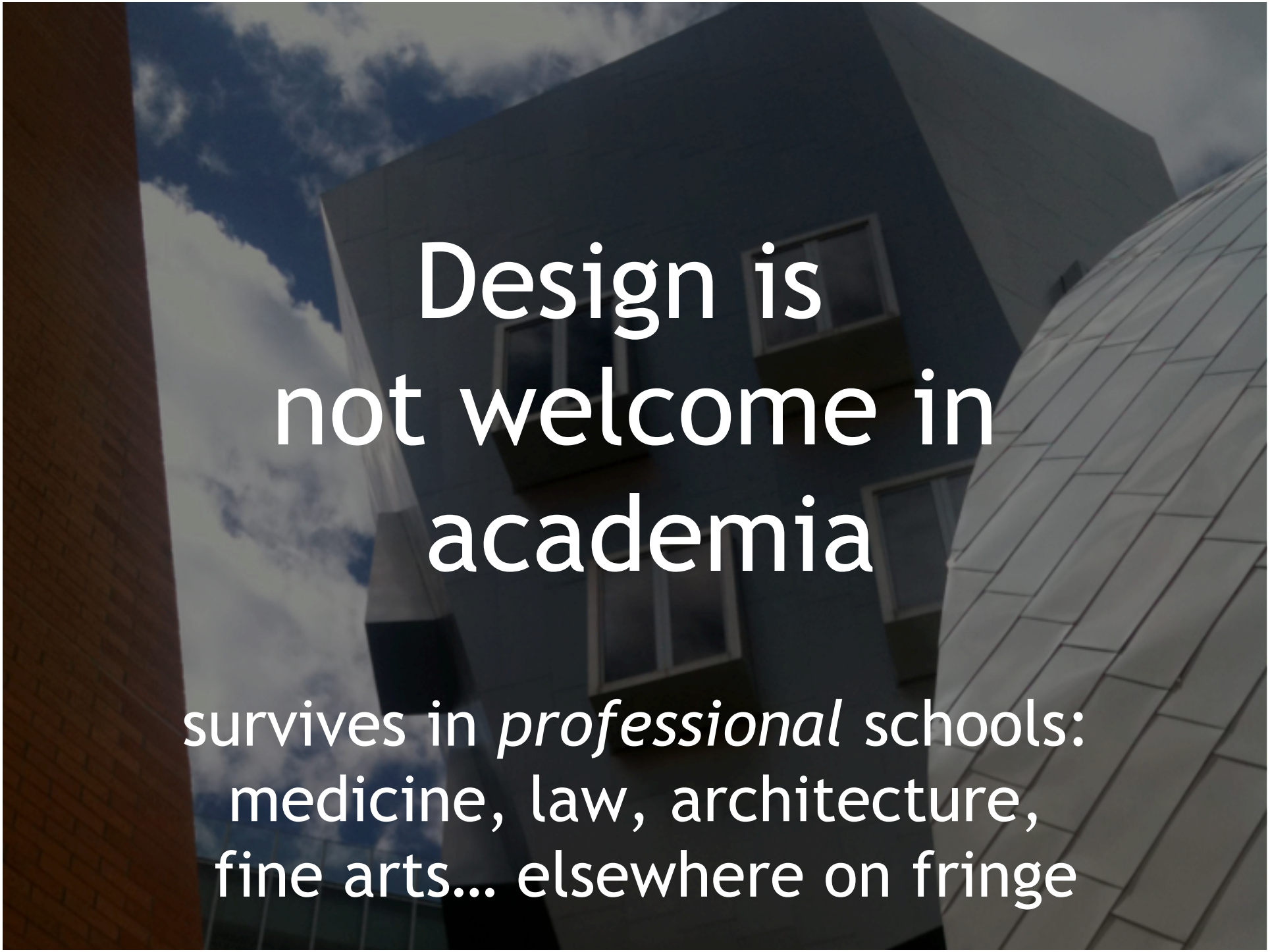

Scientists
design
theories and
experiments

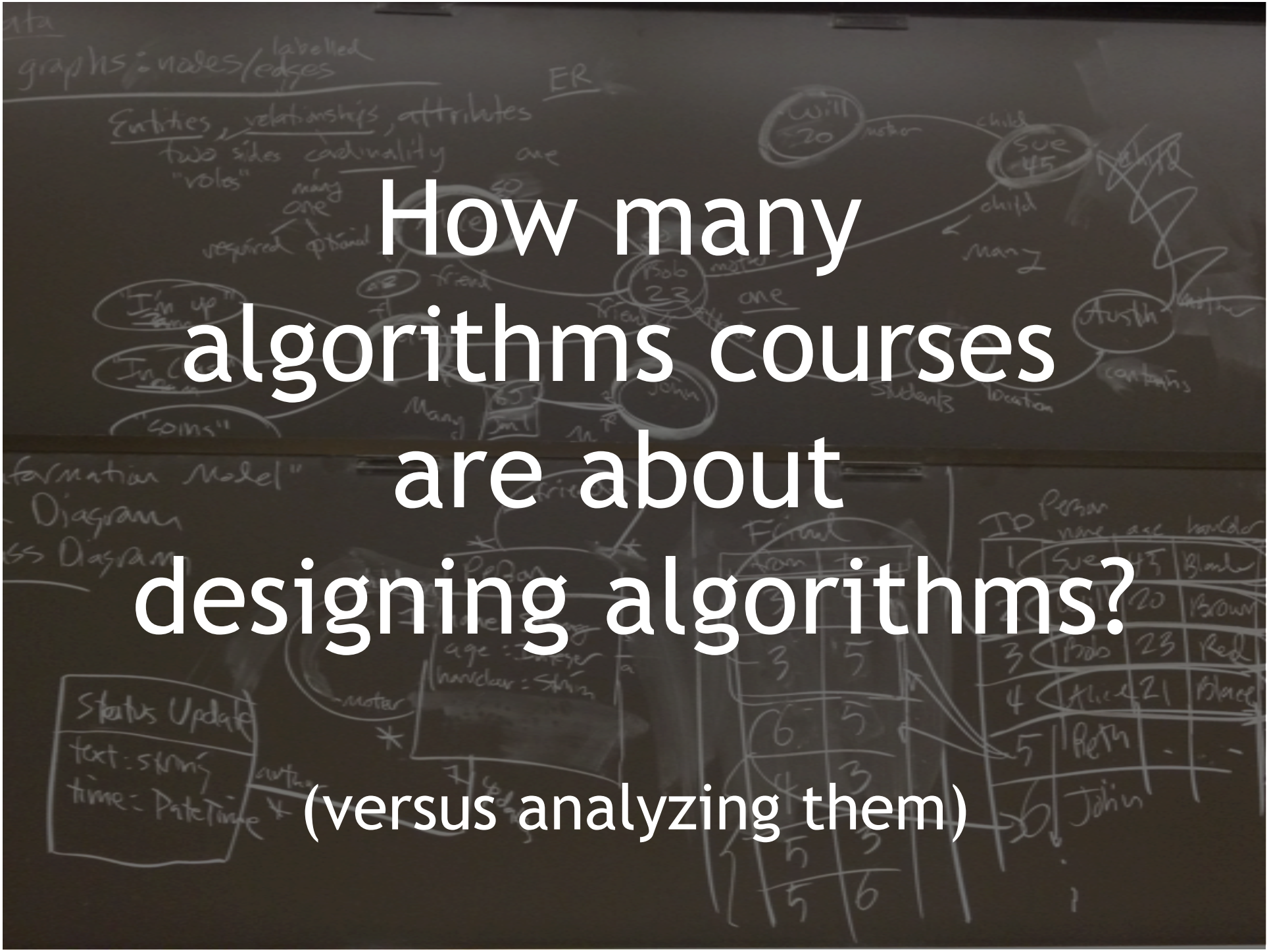
In other words,
the process scientists
use to do science
is not scientific



H. Simon, MIT Press 1969

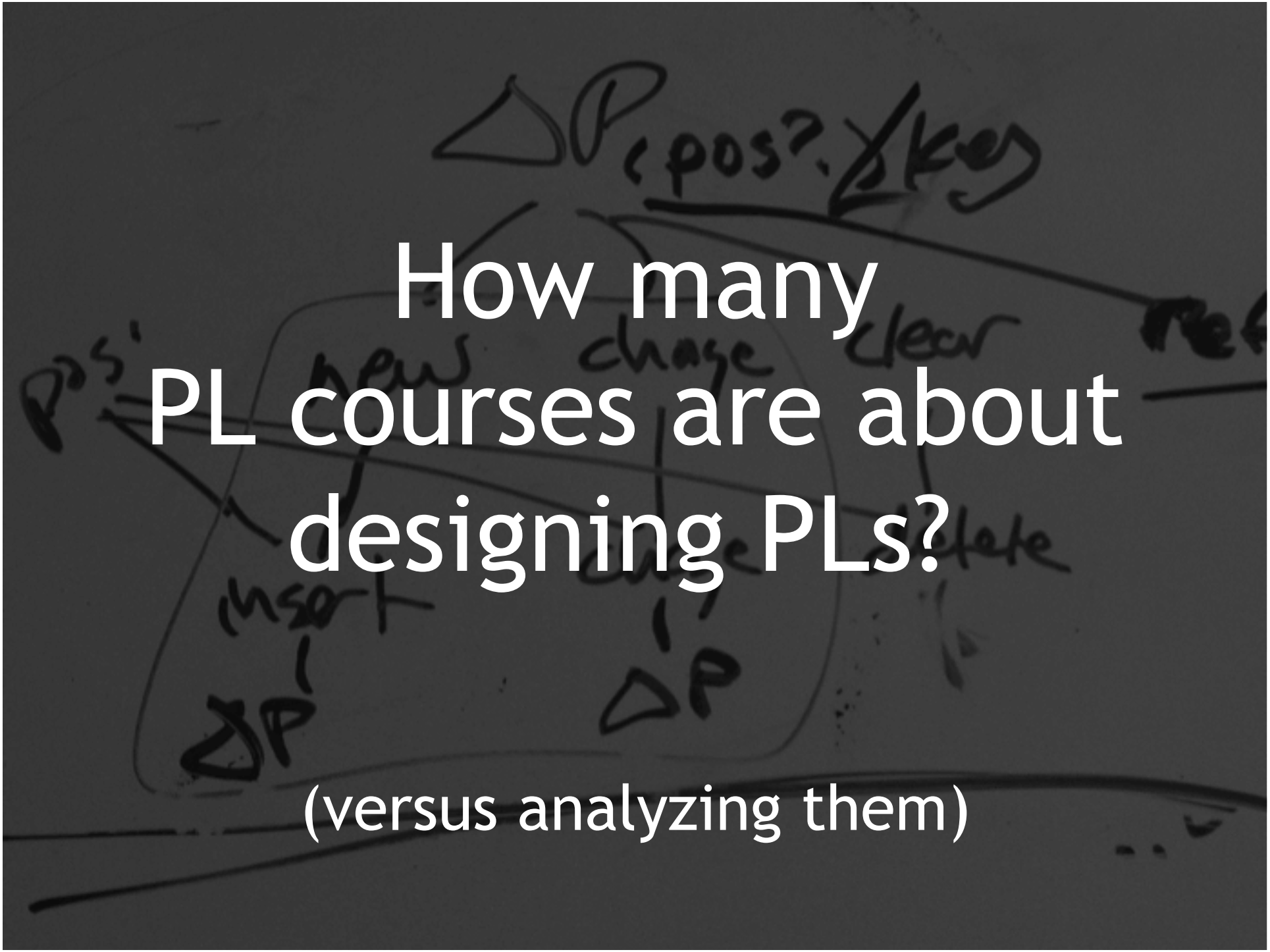

Optimization
Satisficing
Search
Artifact and Process
Does this apply to PL?
(I don't think so)
Search
Directions
iPad
12:42 PM
Google
Airport Blvd
Bennett Ave
Depew Ave
45th St E
I-35 Frontage Rd
35 Service Rd
Harmon Ave
E 47th St
E 46th St
E 45th St
Ellingson Ln
Robert's Gun Shop
Flying Saucer Draught Emporium
Oil Works


Design is
not welcome in
academia
survives in *professional* schools:
medicine, law, architecture,
fine arts… elsewhere on fringe



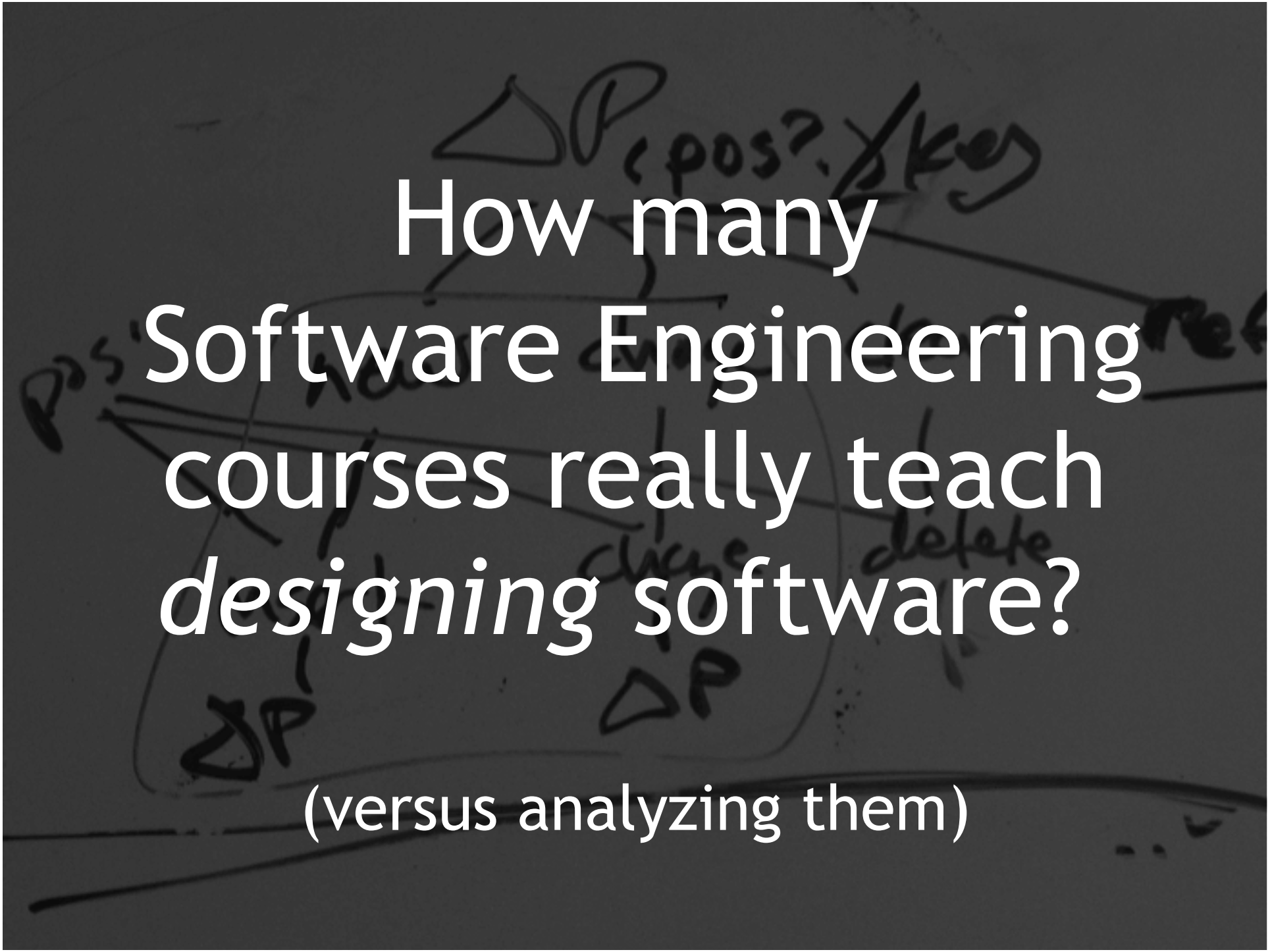
How many algorithms courses are about designing algorithms?

(versus analyzing them)

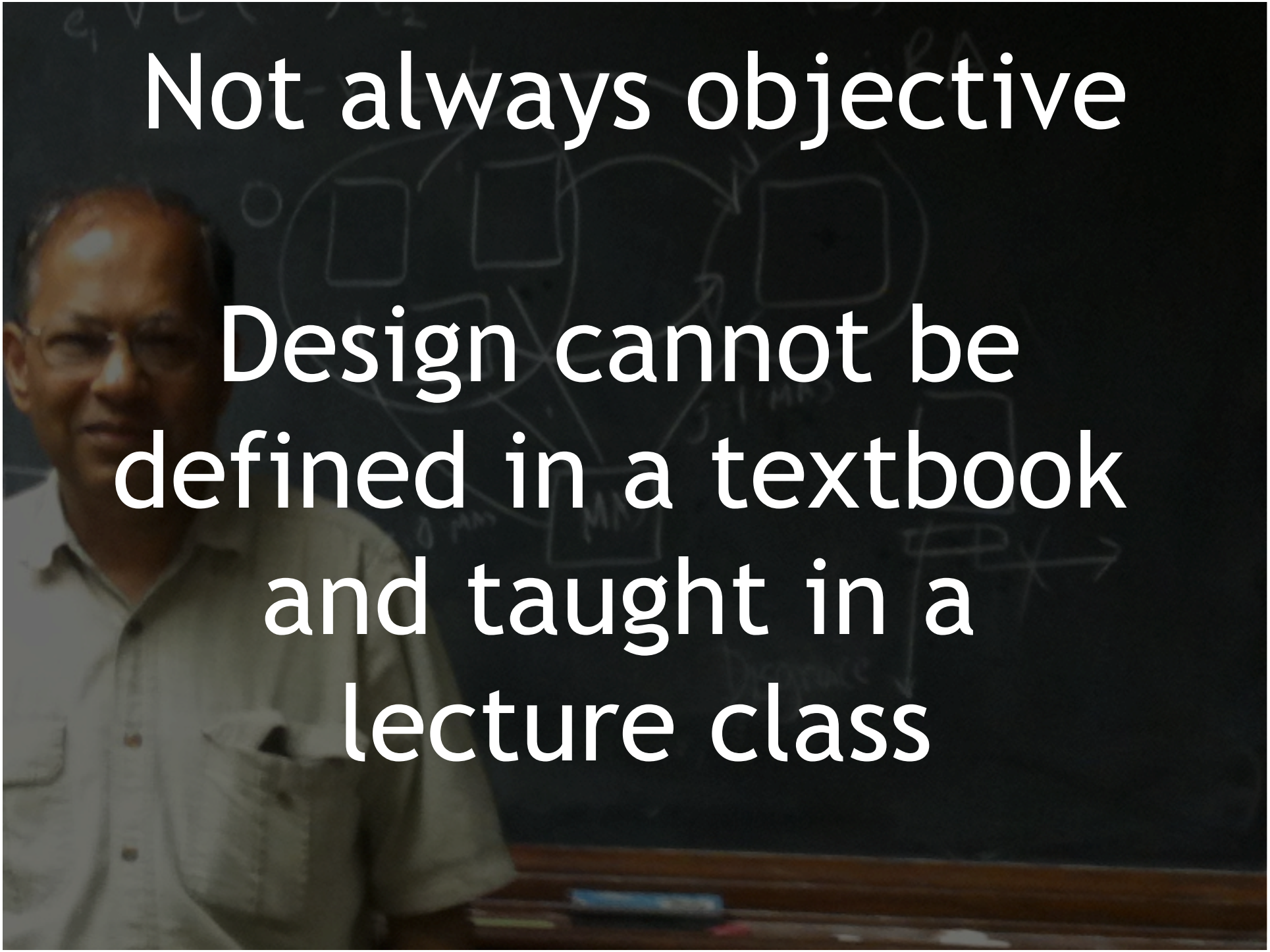


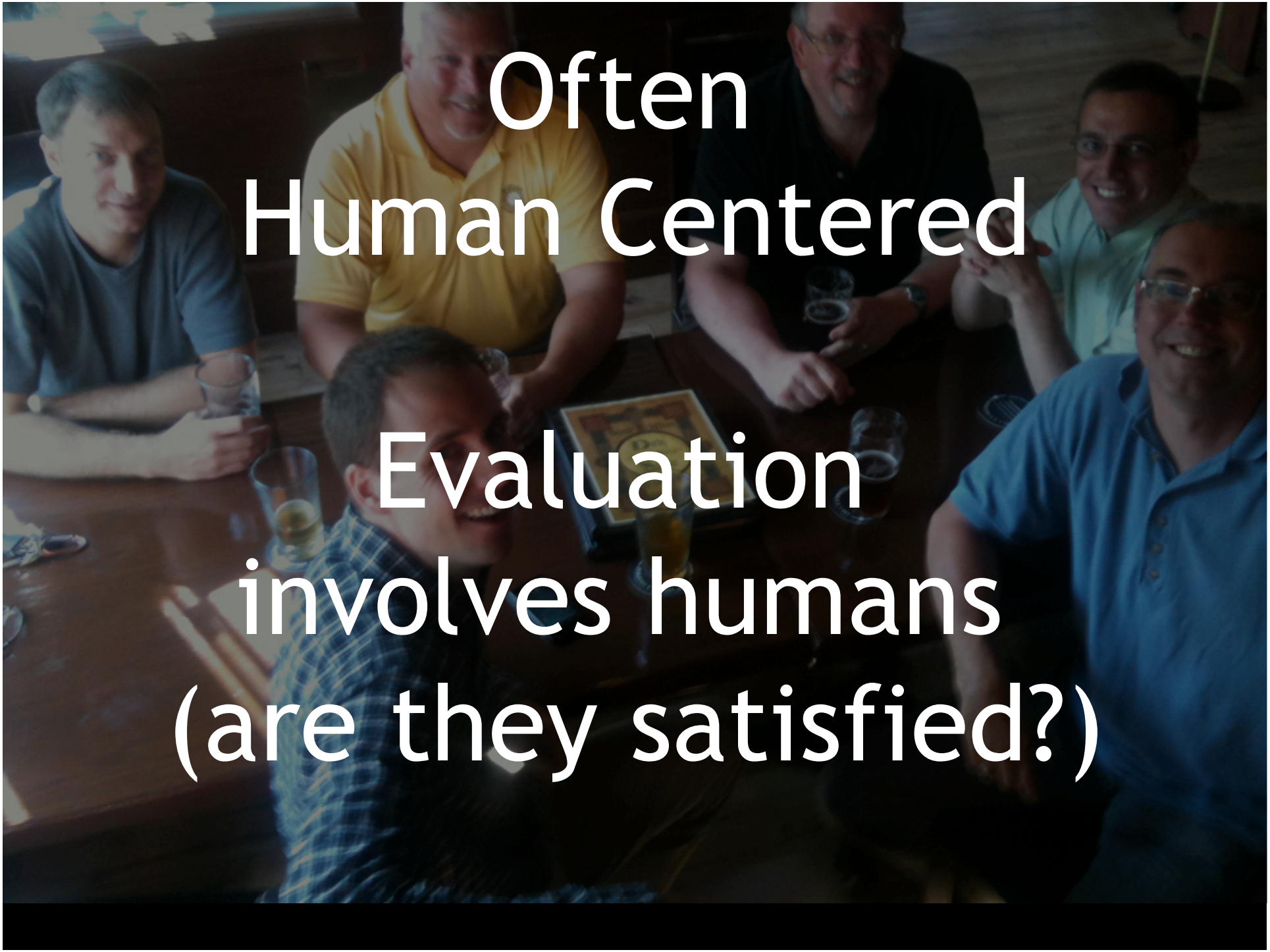
How many PL courses are about designing PLs?

(versus analyzing them)

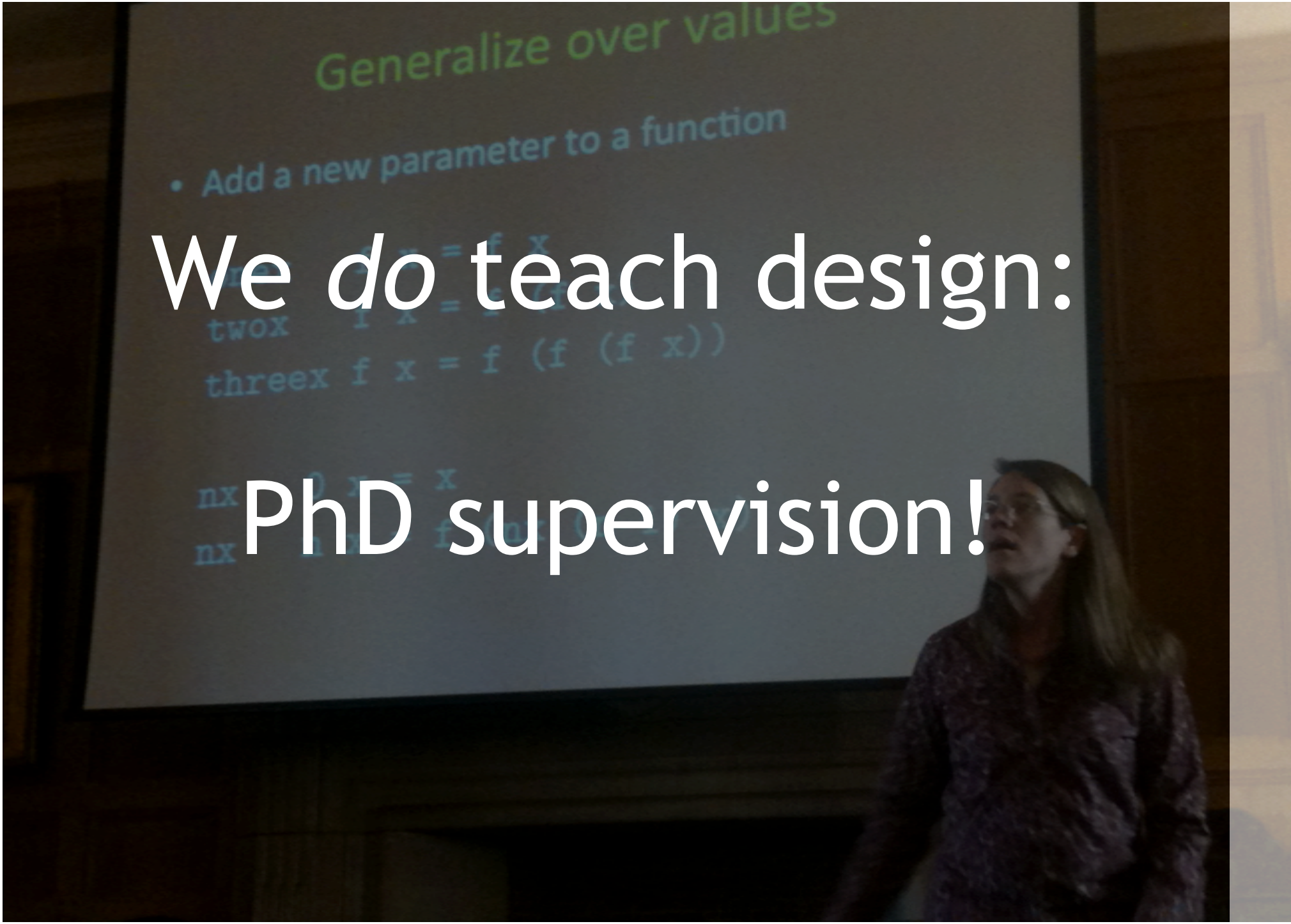

How many
Software Engineering
courses really teach
designing software?
(versus analyzing them)


Not Repeatable
Many design
problems are unique


Not always objective
Design cannot be defined in a textbook and taught in a lecture class


Often
Human Centered
Evaluation
involves humans
(are they satisfied?)


but...

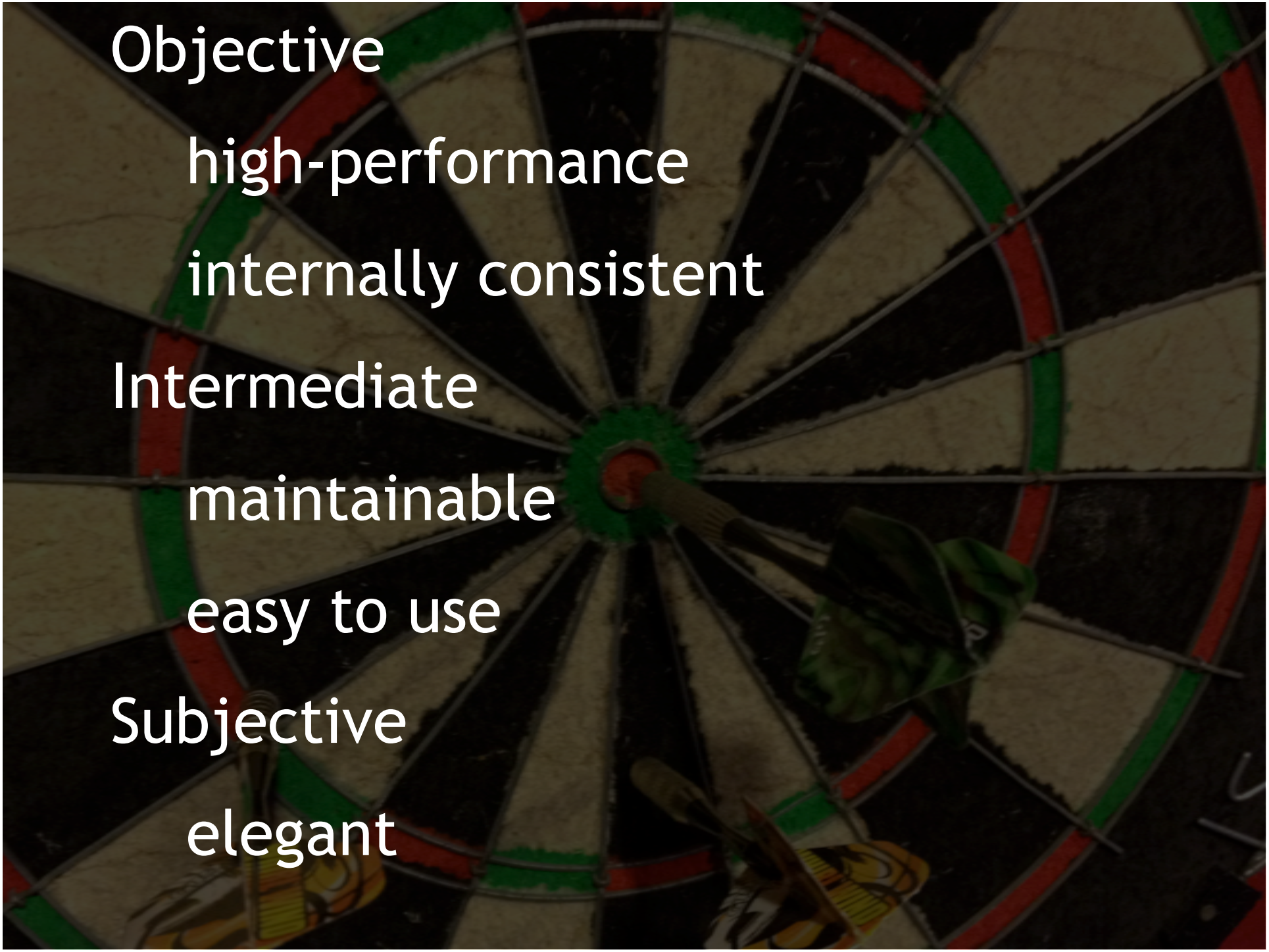

Generalize over values
• Add a new parameter to a function
twox
threex f x = f (f (f x))
We *do* teach design:
PhD supervision!


Apprenticeship
Practice
Critique
Reflect


How do we know
good design?


Good Design
Satisfies the human desire or need


easy to use
high-performance
maintainable
elegant
internally consistent


Objective
high-performance
internally consistent
Intermediate
maintainable
easy to use
Subjective
elegant

Wicked Problems

No test for solutions

Cannot enumerate possible solutions

Every problem is unique, no learning

Defining "wicked problem" is a wicked problem

My Take

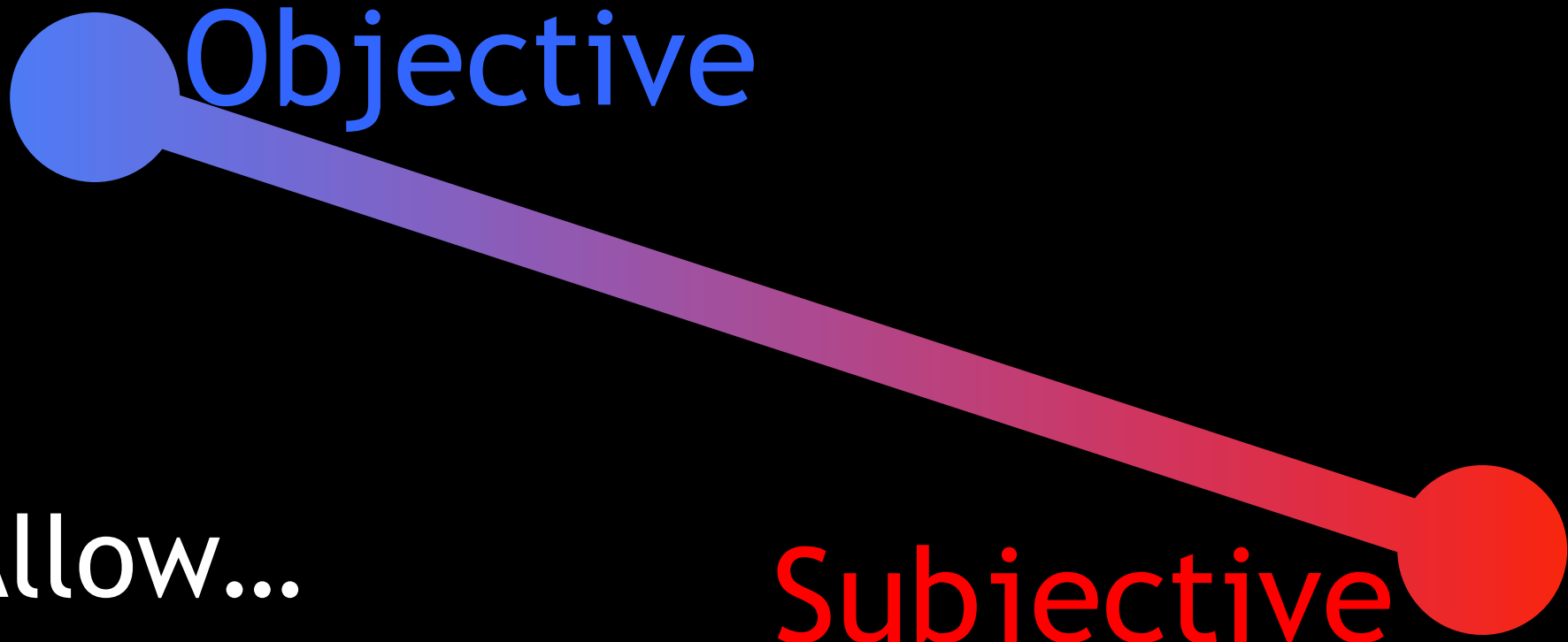
Many things we
really care about…

are not
easy/possible
to measure

Industrial experimentation is our current evaluation mechanism

Academia should embrace design

Spectrum of Criteria



Allow…
discussion
of entire spectrum

User Studies
Repository Mining

are great
but not only options

Need to expand the range of acceptable "tests" for validity

Acceptable Evidence

- Controlled User Study
- Case study
- Historical data mining
- Reasoned argument
- Benchmark design problem
- Structured critique
- Detailed comparisons

Call to Action:

Formalize
PL design paper
review criteria

Other terms besides "scientific"

Academically rigorous

Scholarly

IFIP
Working Group 2.16
on
Language Design

approved last year


design
case
studies

AppleScript

- We did do user studies
- Weren't sure how to do it!
- They didn't influence the language much
- We still ended up with "partial success"

Understanding

Objects

First-class behaviors (dual of ADTs)

Inheritance

Open recursion (not just for objects)

See R. Gabriel "The Structure of a Programming Language Revolution", Onward! 2012

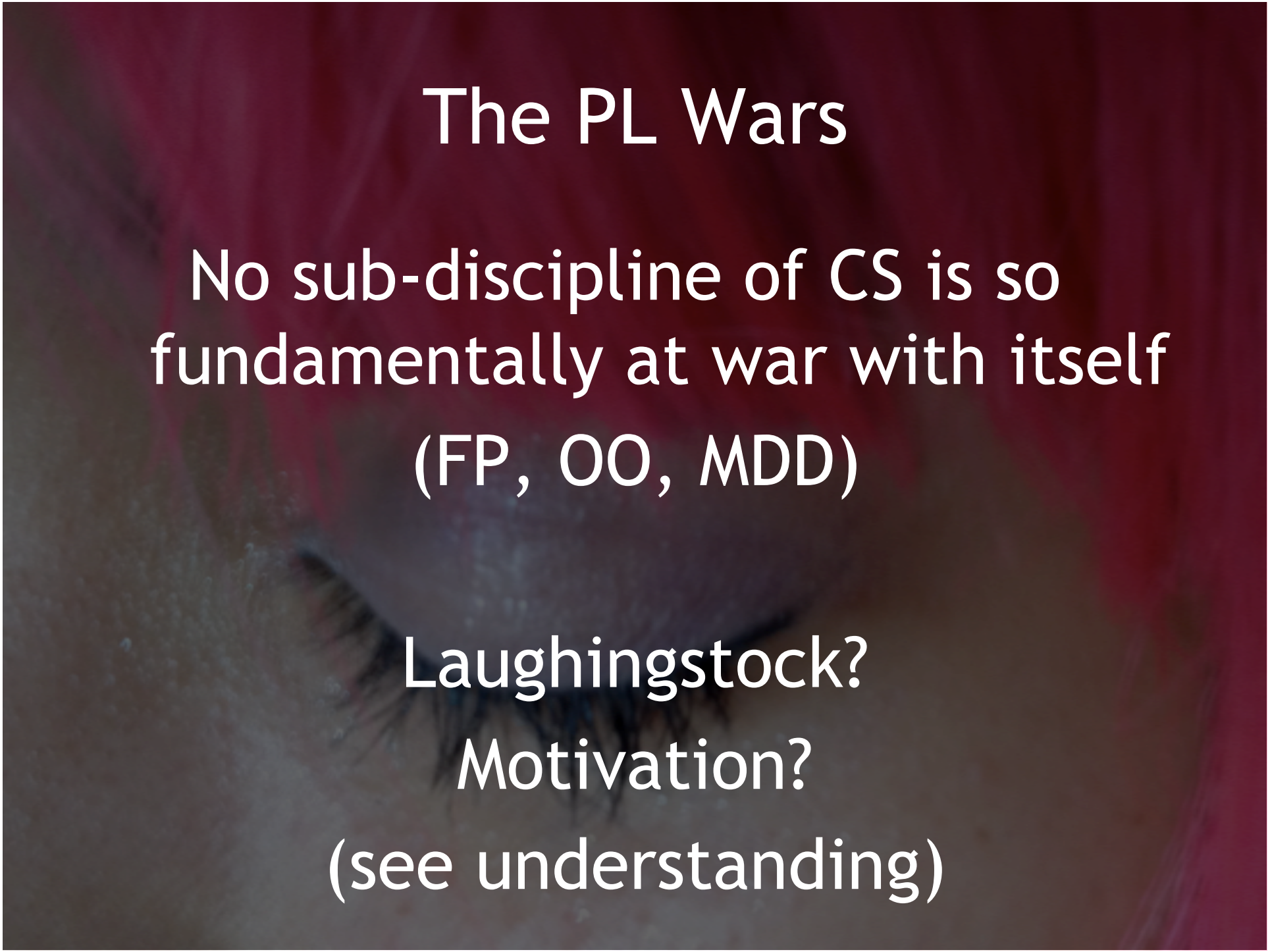
Semantics

Denotational over operational

Operational wins

- typing proofs
- concurrency

Featherweight Java tells you what inheritance does, not what it means


The PL Wars
No sub-discipline of CS is so fundamentally at war with itself
(FP, OO, MDD)
Laughingstock?
Motivation?
(see understanding)

Choose Good Examples

Remote Method Call

local.print(remote.proc(inputs))

Conclusions:

- marshall data
- create remote proxies
- serialize objects

Choose Good Examples

Multiple Remote Method Calls

local.print(remote.$proc_1$($inputs_1$))

local.print(remote.$proc_2$($inputs_2$))

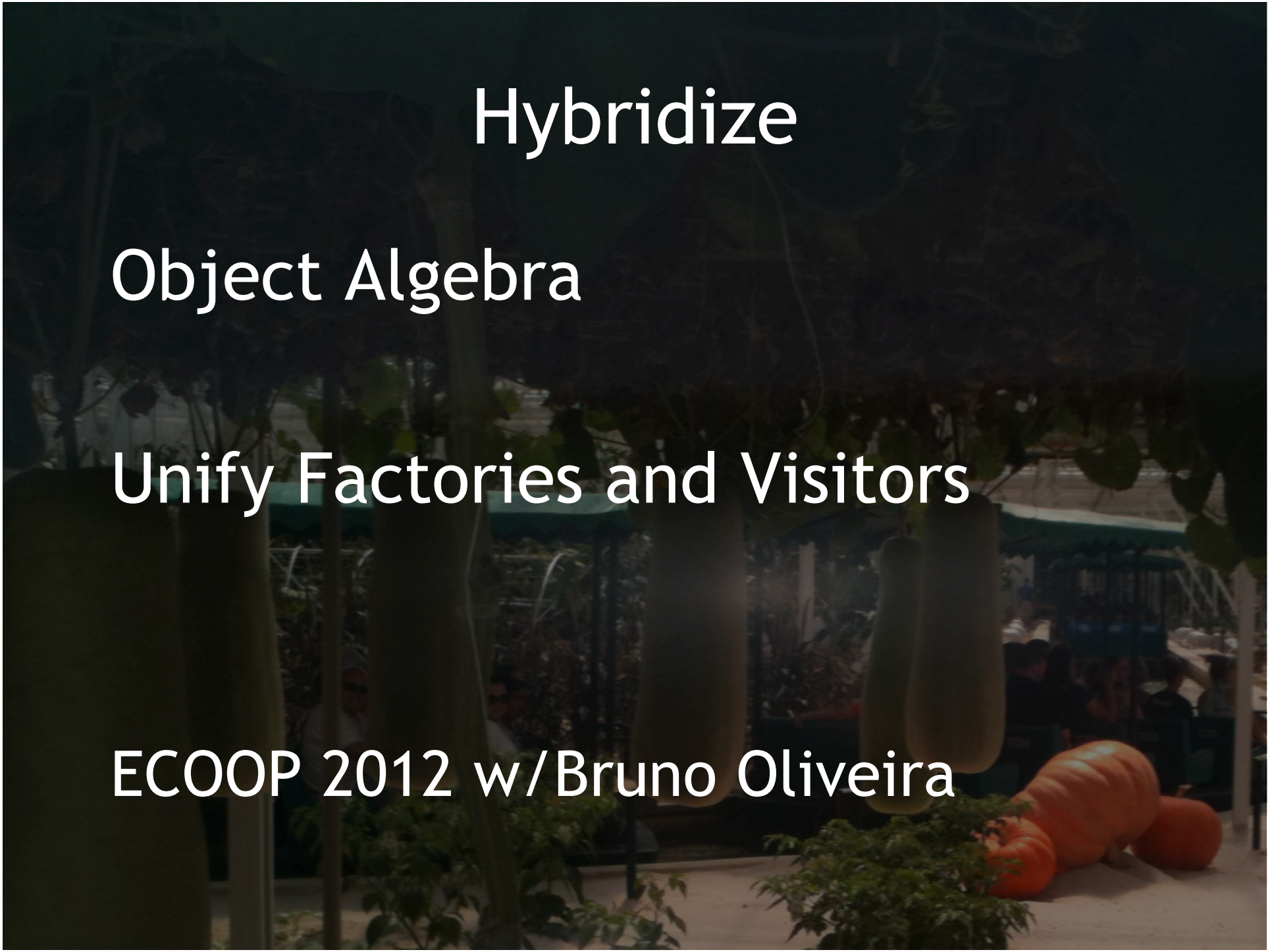
Conclusions:

send multiple calls to server at once

bulk transfer of inputs and results

no serialization, no proxies

"batches" include conditionals and loops


Hybridize
Object Algebra
Unify Factories and Visitors
ECOOP 2012 w/Bruno Oliveira


Ensō
(motivation)
with Tijs van der Storm
Alex Loh (see Onward! 2012)

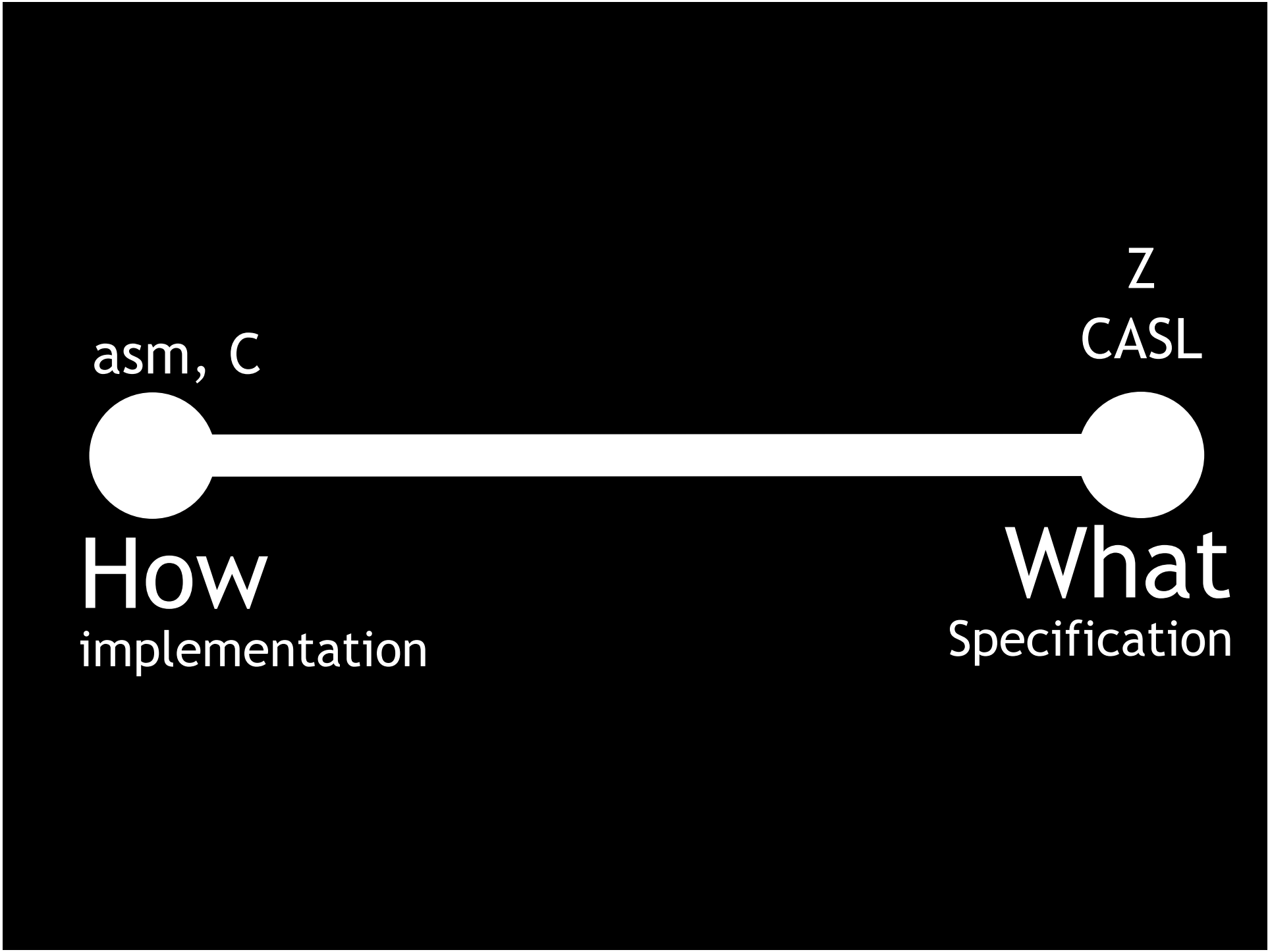
Spectrum of Programming

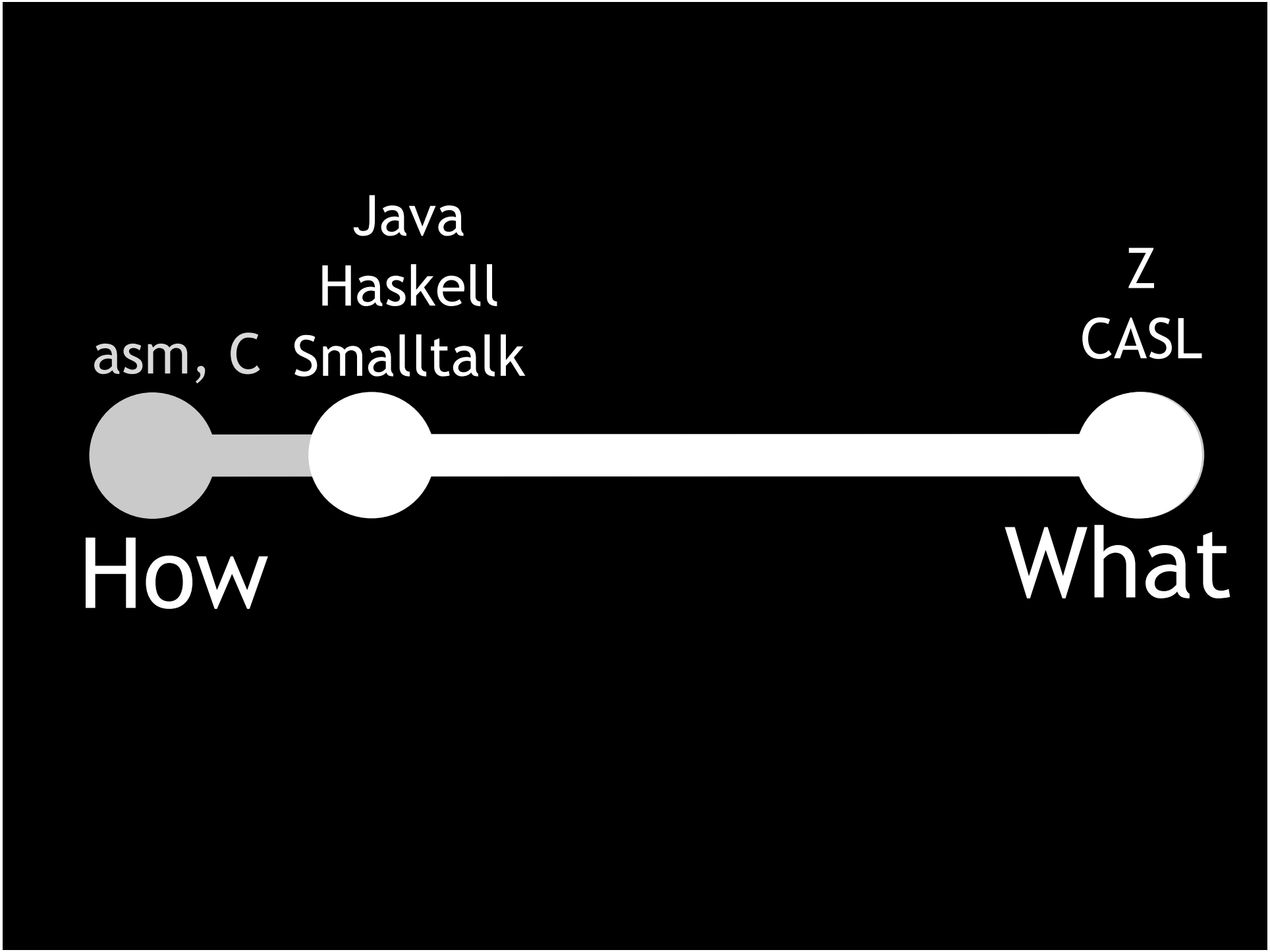
How

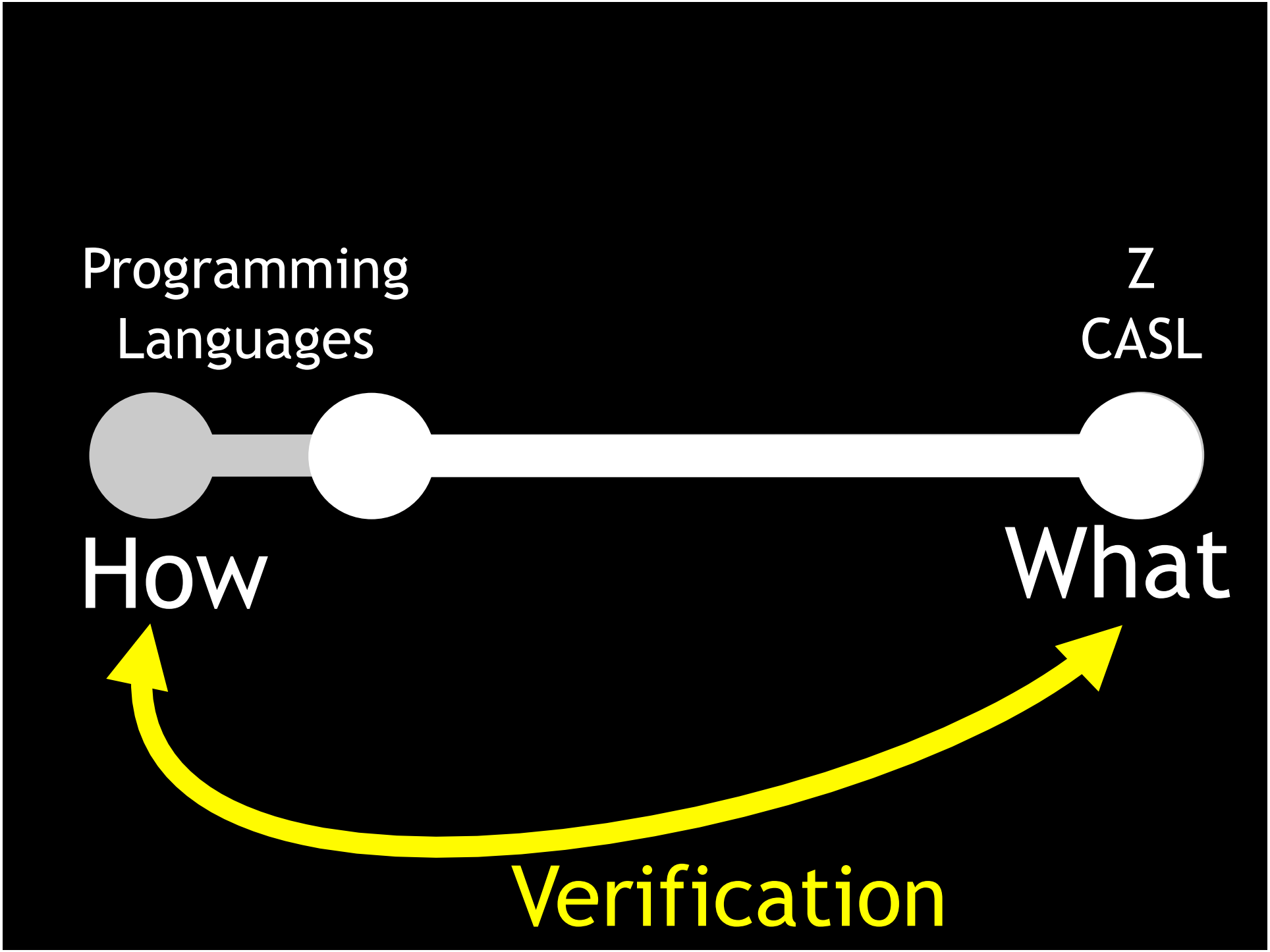
implementation

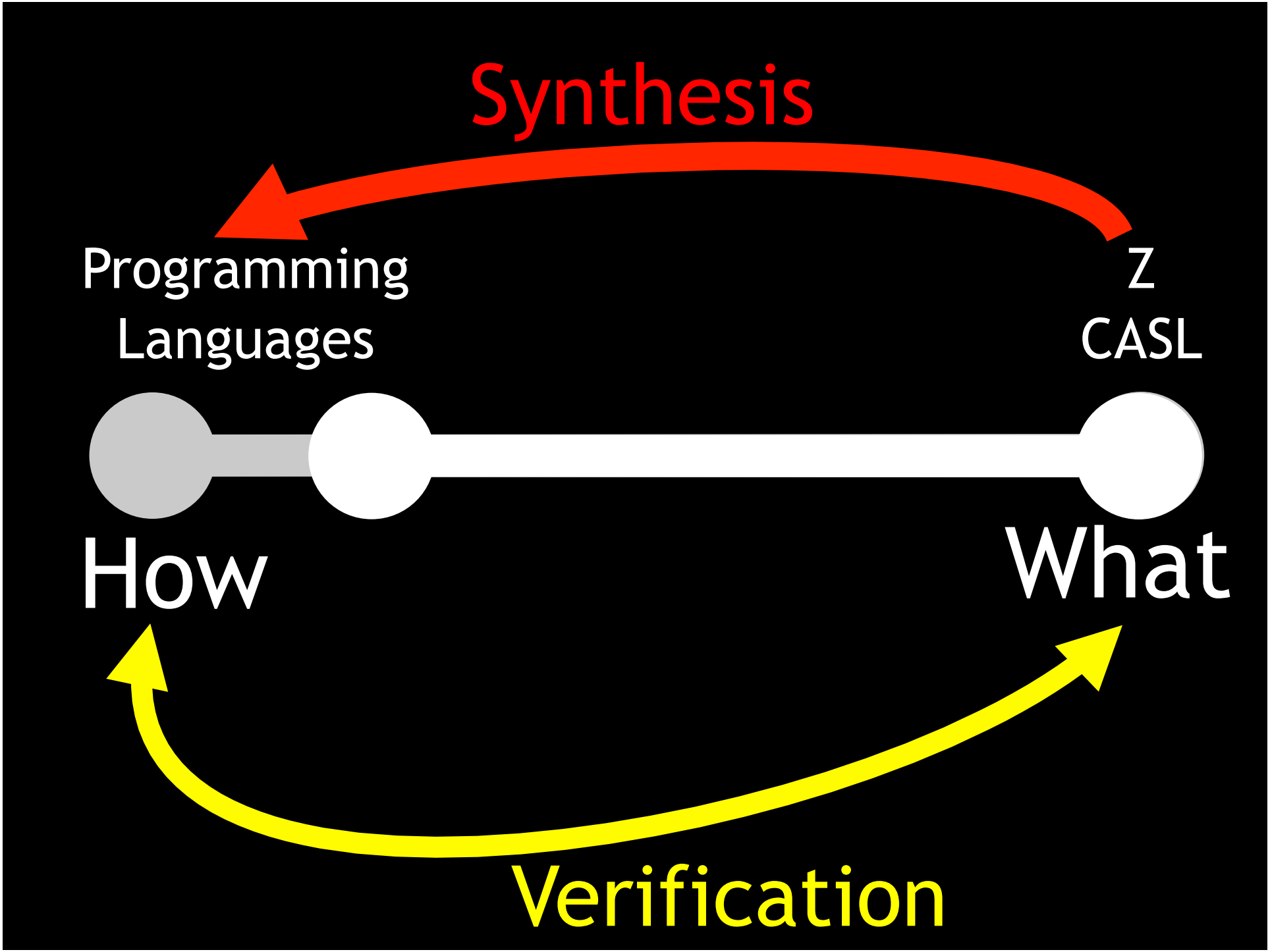
What

Specification


Z
CASL
asm, C
How
implementation
What
Specification

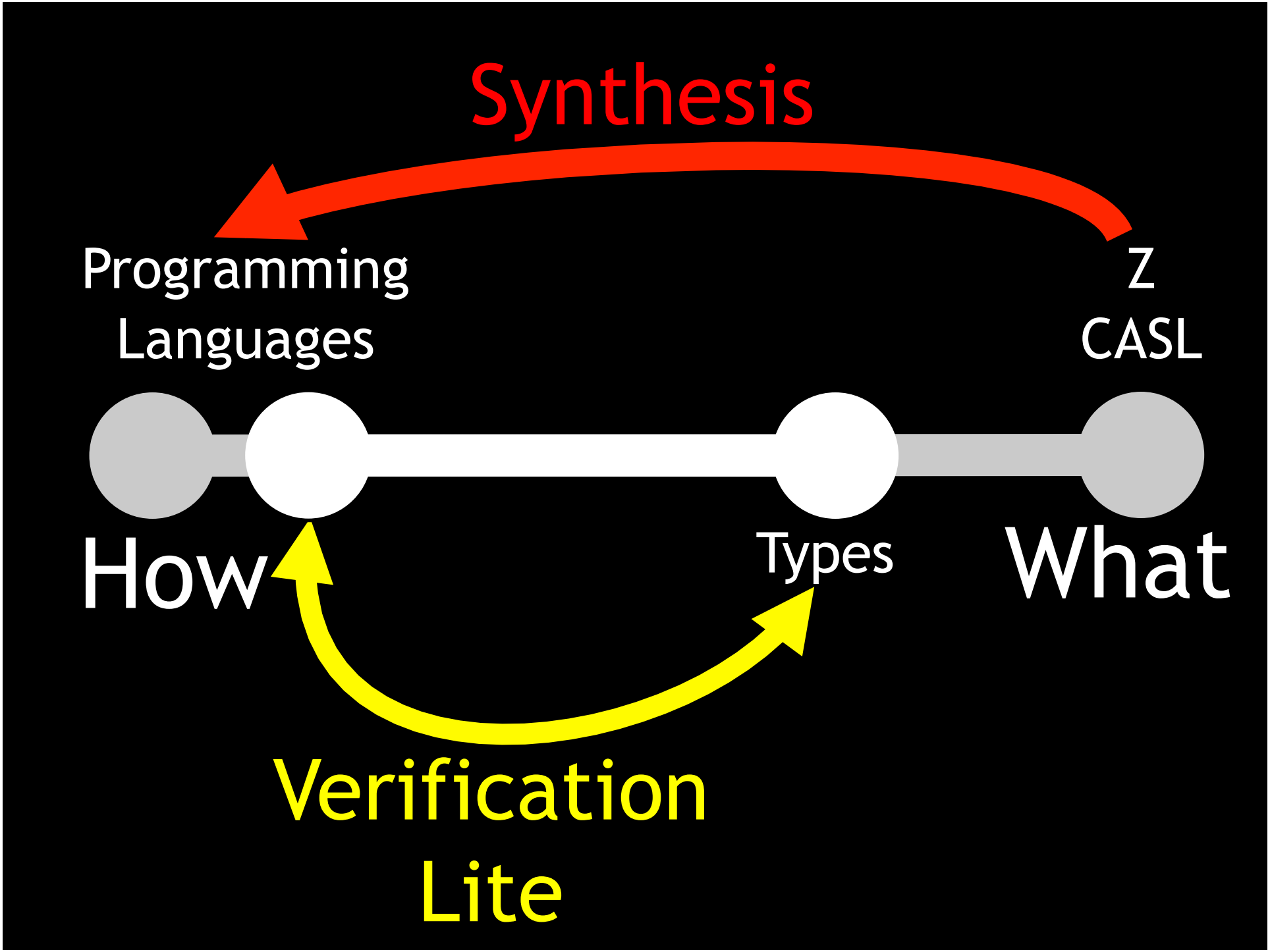

Java
Haskell
Smalltalk
asm, C
Z
CASL
How
What

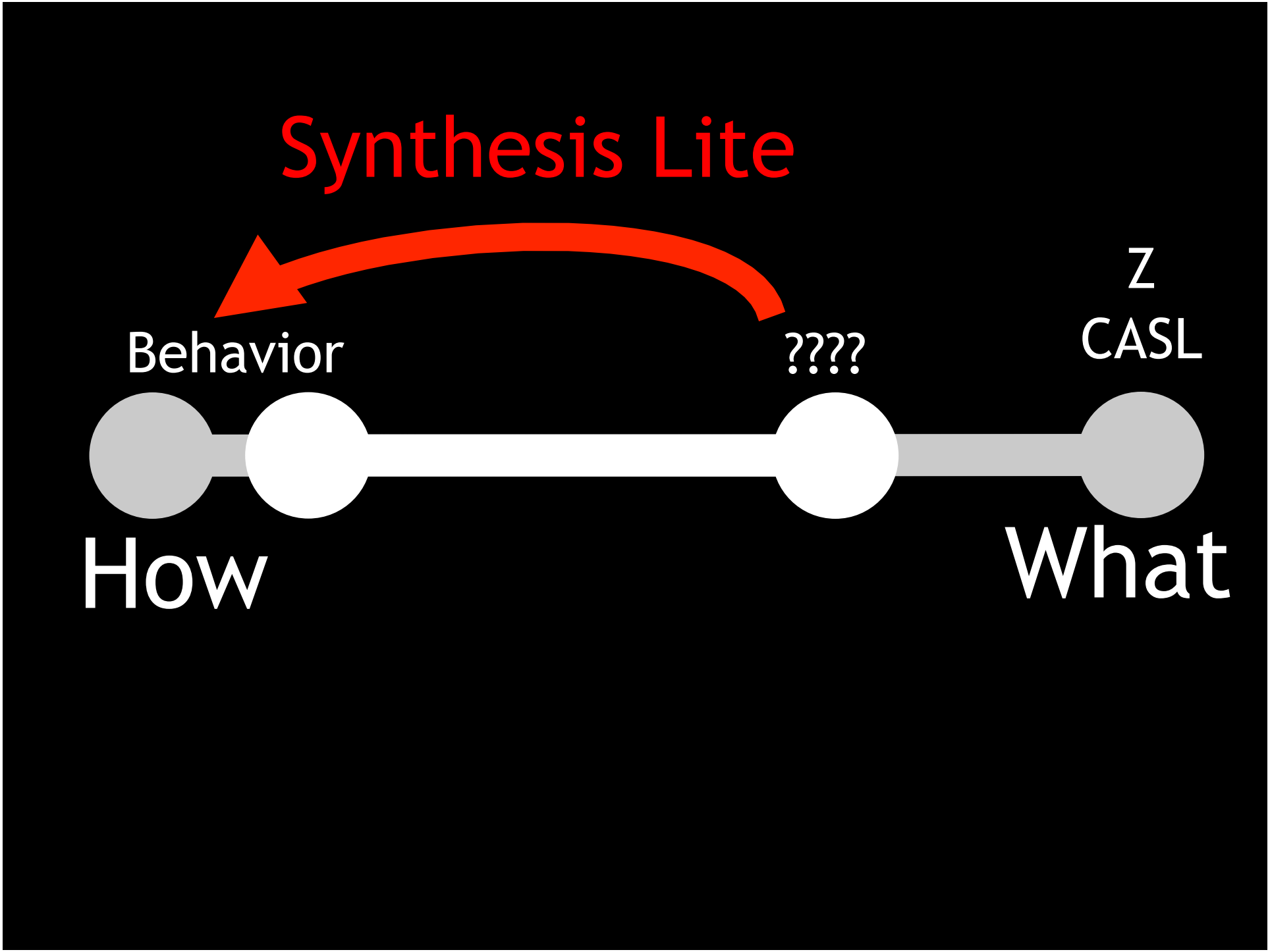

Programming
Languages
Z
CASL
How
What
Verification

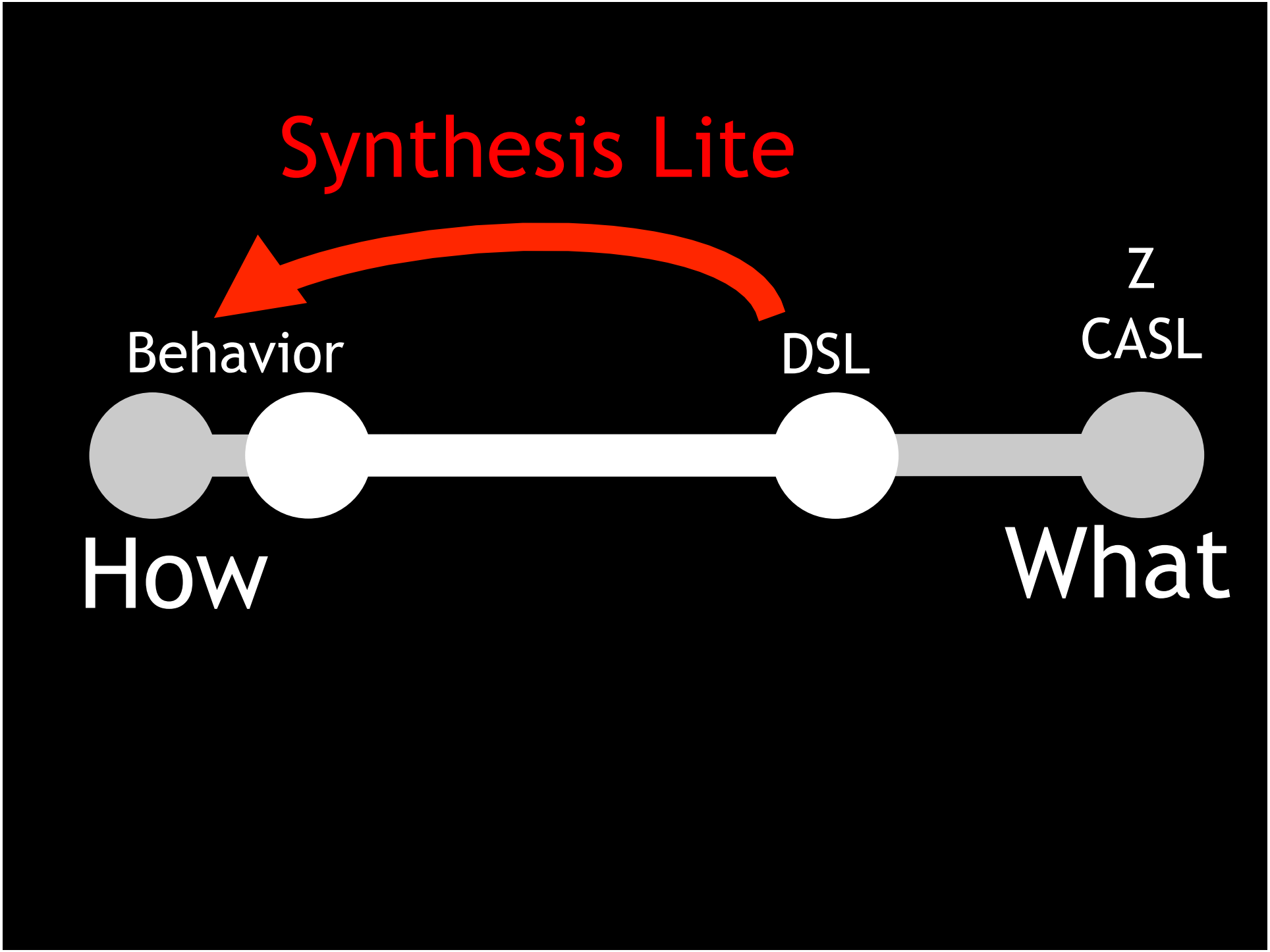

Synthesis
Programming Languages
Z CASL
How
What
Verification

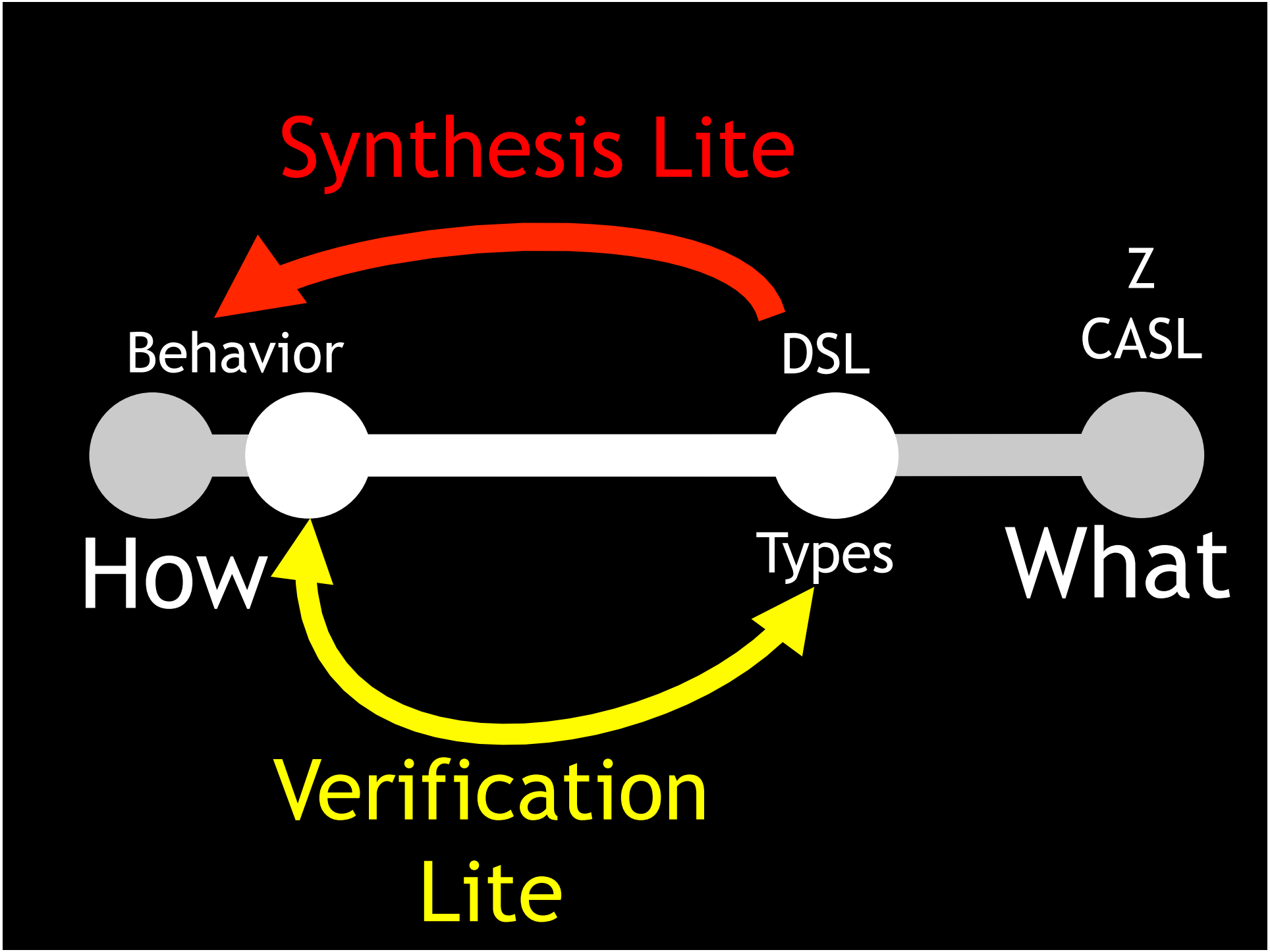
[Note]

Programming Languages Grand Challenges panel didn't even mention synthesis


Synthesis
Programming Languages
Z CASL
How
Types
What
Verification Lite


Synthesis Lite
Behavior
????
Z
CASL
How
What


Synthesis Lite
Behavior
DSL
Z
CASL
How
What


Synthesis Lite
Behavior
DSL
Z
CASL
How
Types
What
Verification
Lite

Ensō Plan

Integrate and **Extend** DSLs

Standalone, not embedded

Interpret, not compile/transform

Graphical + Textual

Partial evaluation for speed

Data, Grammars, Security, Workflow,
Diagrams, GUIs, WebUI, Synchonization

Prevent Bad

Enable Good

Bug Finding
Race Detection
Type Checking
etc.

Prevent Bad

Enable Good

Bug Finding
Race Detection
Type Checking
etc.

Prevent Bad

Enable Good

New languages?
New features?
For what?

Bug Finding
Race Detection
Type Checking
etc.

Prevent Bad

Advantages:
Measurable
Domain-free

Enable Good

New languages?
New features?
For what?

If somebody comes up with the next big thing after objects… all bets are off

Lets try to do this!

simplicity
is the
result
of hard work

Embrace Design

Don't fall prey to
"science envy"

academic rigor
not rigor mortis

Don't Design Your Programs

Program
Your Designs