What Good Are Formal Models?

Edward A. Lee

Professor of the Graduate School

Invited Talk

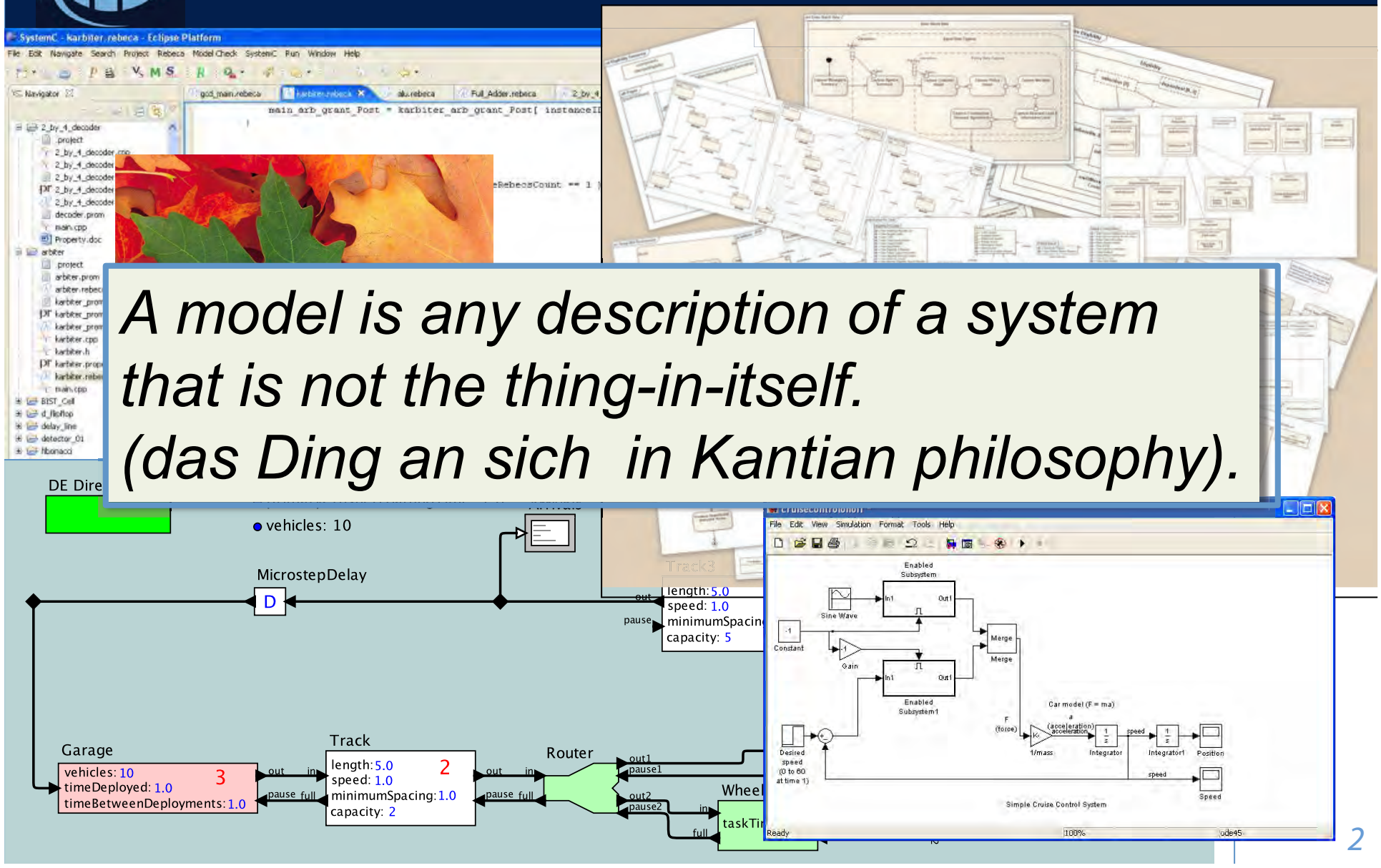
(with contributions from *Marjan Sirjani*, MDH, Sweden)

Formal Aspects of Component Software (FACS)

October 10-12, 2018, Pohang, South Korea

University of California at Berkeley

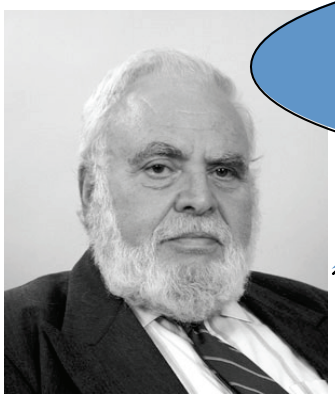
In This Talk: Focus on Models



A model is any description of a system that is not the thing-in-itself. (das Ding an sich in Kantian philosophy).

Challenges

- Confusing the map and the territory
- Choosing a modeling paradigm
- Understanding the purpose of the model



Solomon Wolf Golomb

You will never strike oil by drilling through the map!

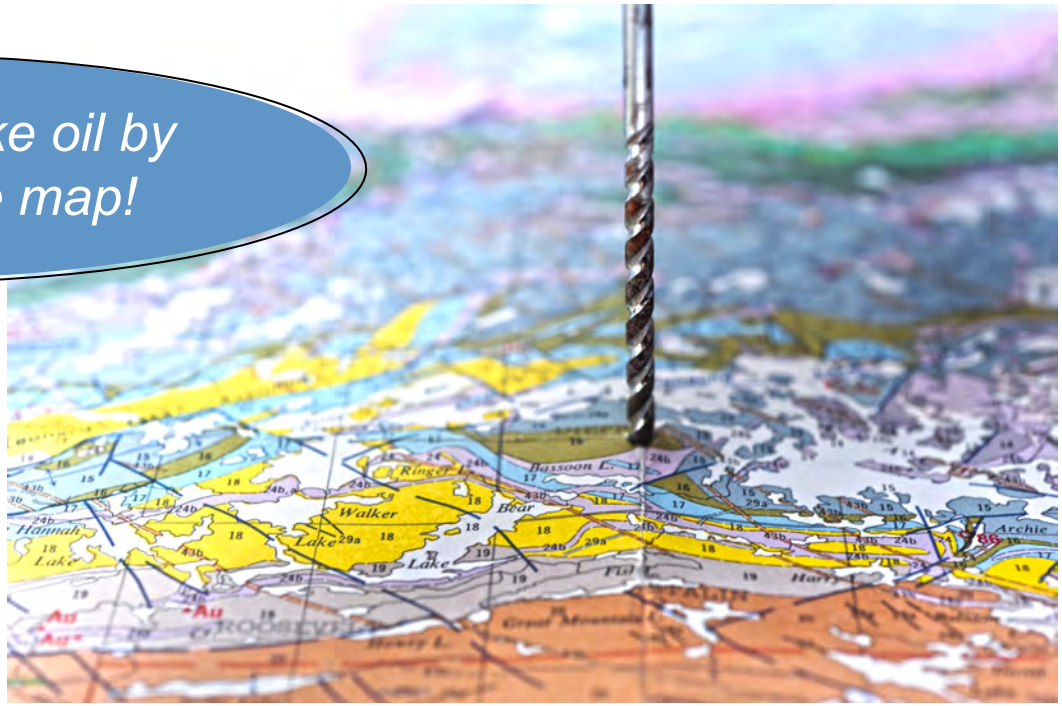
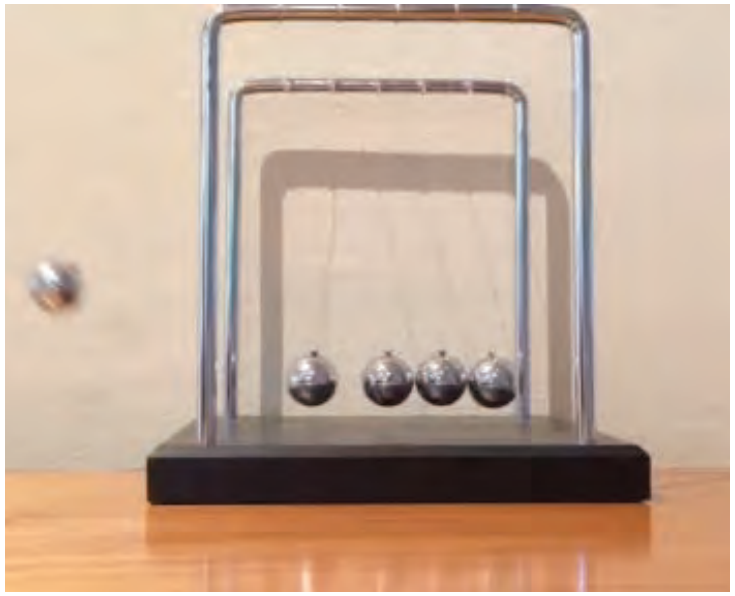


Photo by Rusi Mchedlishvili

Models vs. Reality

$$x(t) = x(0) + \int_0^t v(\tau)d\tau$$
$$v(t) = v(0) + \frac{1}{m}\int_0^t F(\tau)d\tau.$$

The model



The target (the thing being modeled).

In this example, the *modeling universe* is calculus and Newton's laws.

Faithfulness is how well the model and its target match

A Model

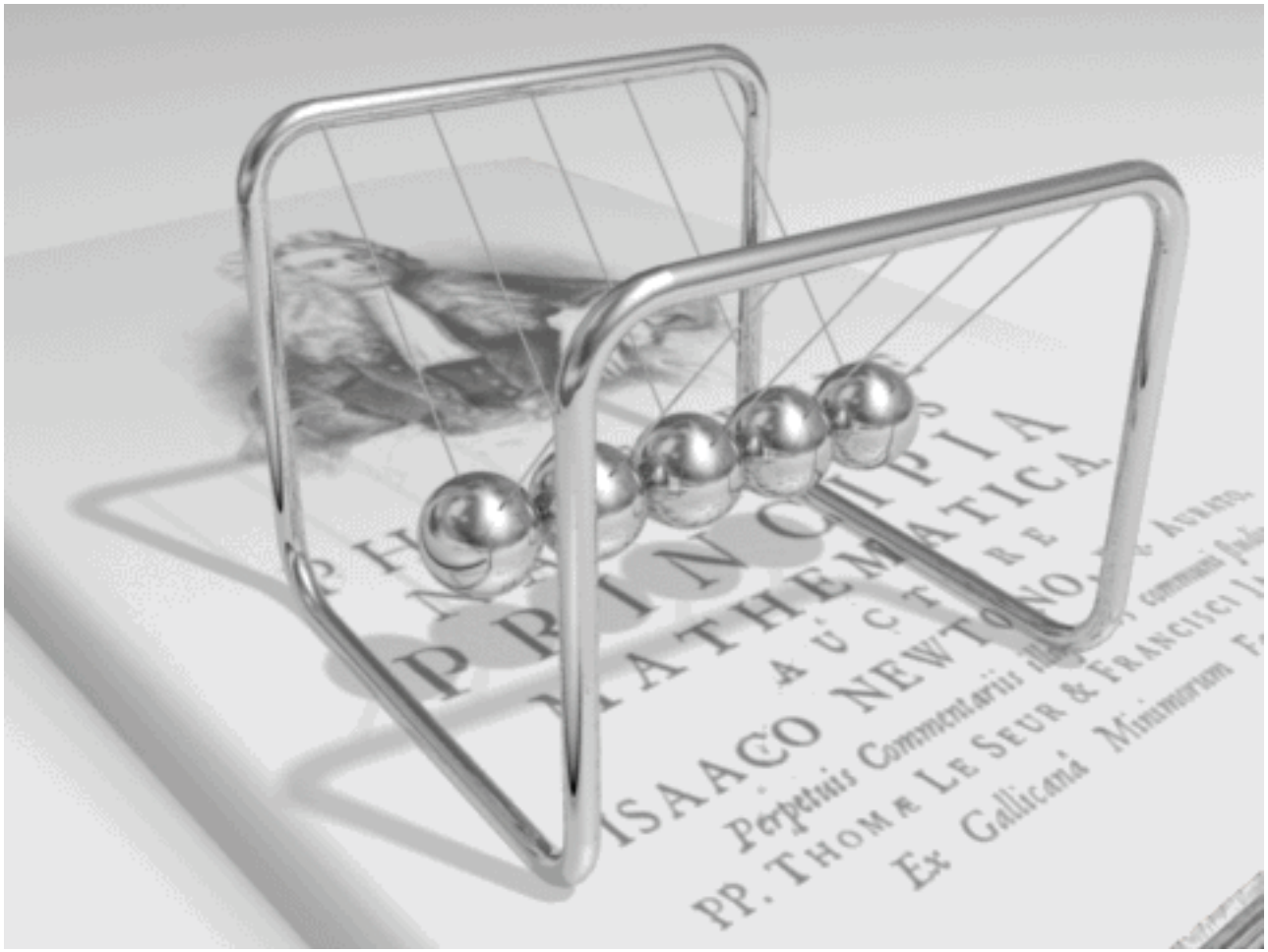
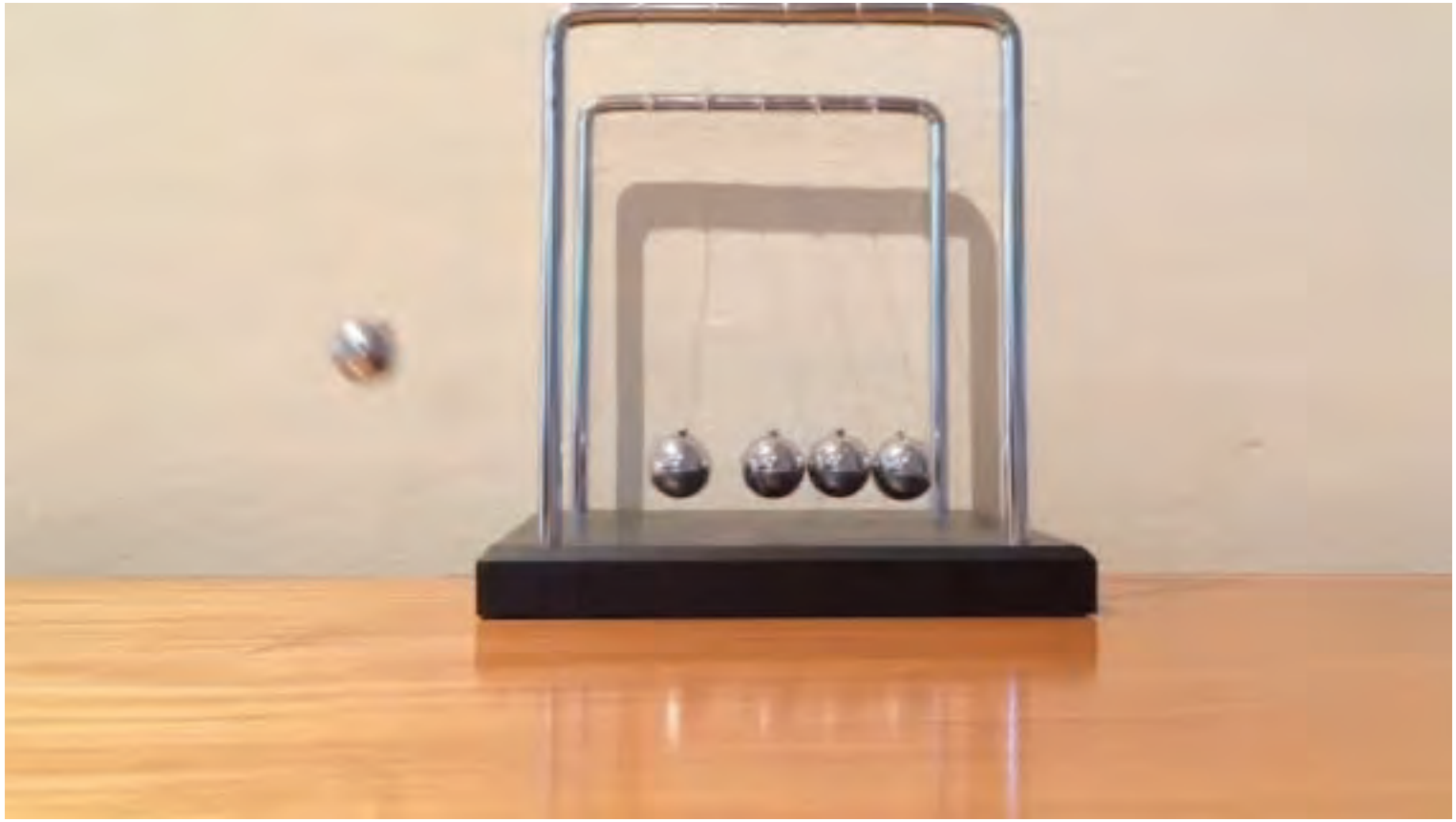


Image by Dominique Toussaint, GNU Free Documentation License, Version 1.2 or later.

A Physical Realization



The Value of Models

- In *science*, the value of a *model* lies in how well its behavior matches that of the physical system.

- In *engineering*, the value of the *physical system* lies in how well its behavior matches that of the model.

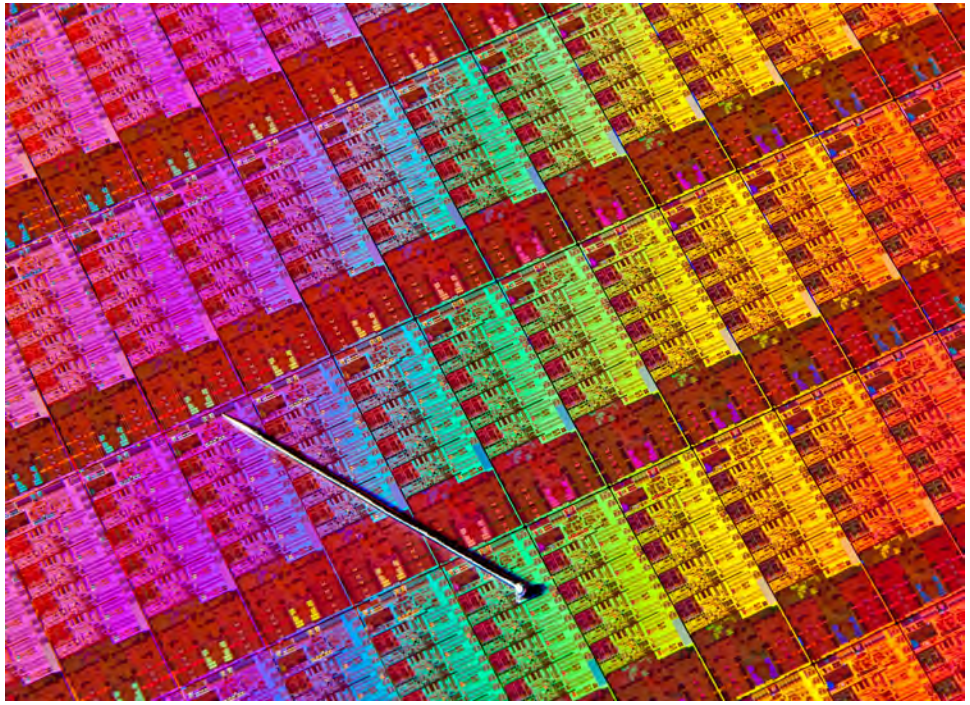
A scientist asks, “Can I make a model for this thing?”
An engineer asks, “Can I make a thing for this model?”

Model Faithfulness

- To a *scientist*, the model is flawed.
- To an *engineer*, the realization is flawed.

Engineering is about making the thing match the model rather than the other way around.

Consider Chip Design



Intel Haswell, each with 1.4 billion transistors

A piece of silicon that doesn't behave like the model is just beach sand.

The Value of Simulation

"Simulation is doomed to succeed."

Could this statement be confusing engineering models for scientific ones?

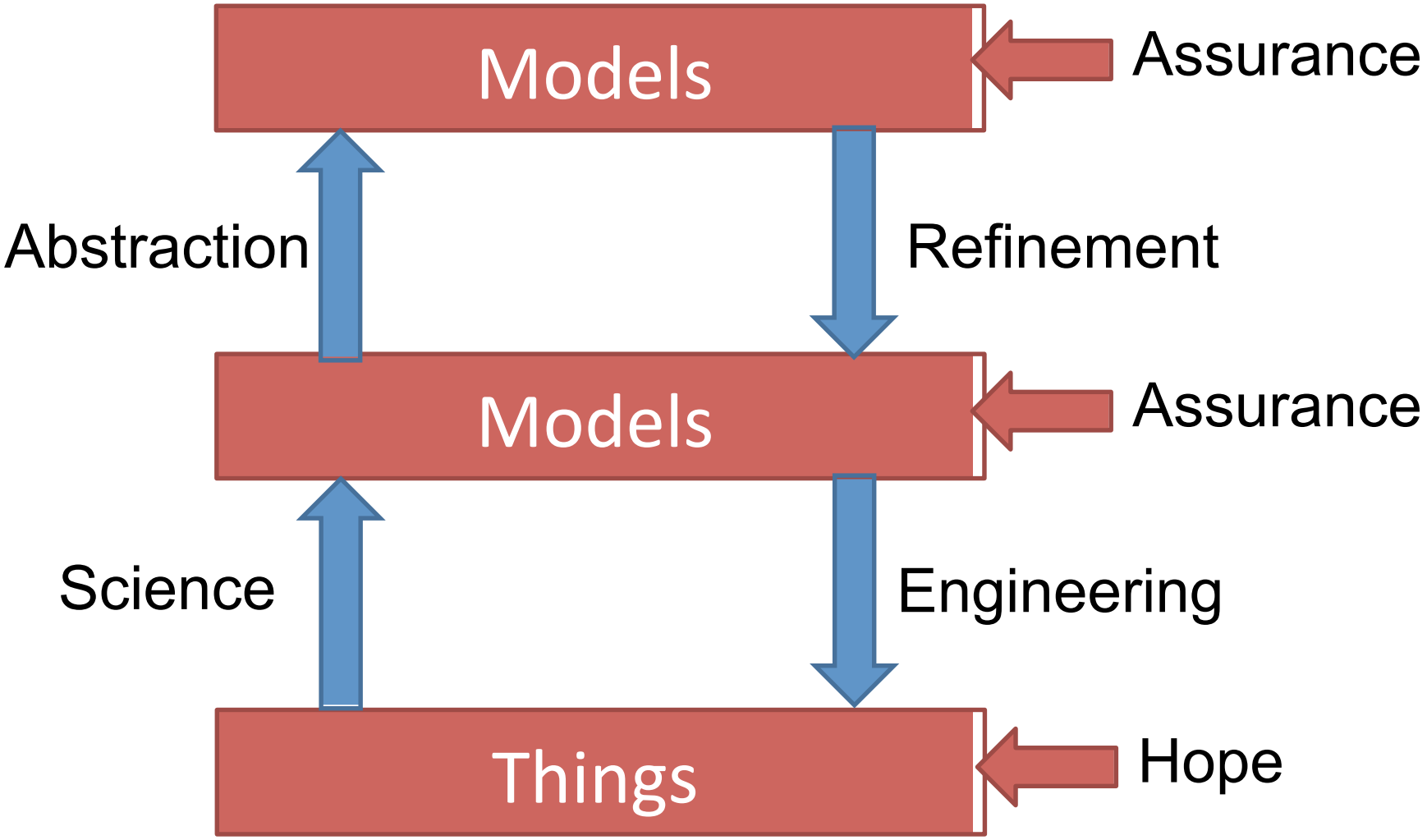
Abstraction and Refinement

- An abstraction **A** of **B** is sound if every *property of interest* that is true for **A** is also true for **B**

- If **A** is a sound abstraction of **B**, then **B** is a refinement of **A**

A simulation (of model **A**) is "doomed to succeed" in that it will not reveal properties of a refinement (**B**) nor of the thing-in-itself that are not also properties of **A**.

Models and Things (vs. Models and Models)



Useful Models and Useful Things

"Essentially, all models are wrong,
but some are useful."

Box, G. E. P. and N. R. Draper, 1987: *Empirical Model-Building and Response Surfaces*. Wiley Series in Probability and Statistics, Wiley.

"Essentially, all system implementations
are wrong, but some are useful."

Lee and Sirjani, "What good are models," FACS 2018.

Faithfulness

- A scientific model is faithful if every property it exhibits is also a property of the system being modeled.

(Like soundness, but a relation between a model and a thing-in-itself, not between models.)

Changing the Question

Is the question whether our models describe the thing in itself (faithfully)?

Or

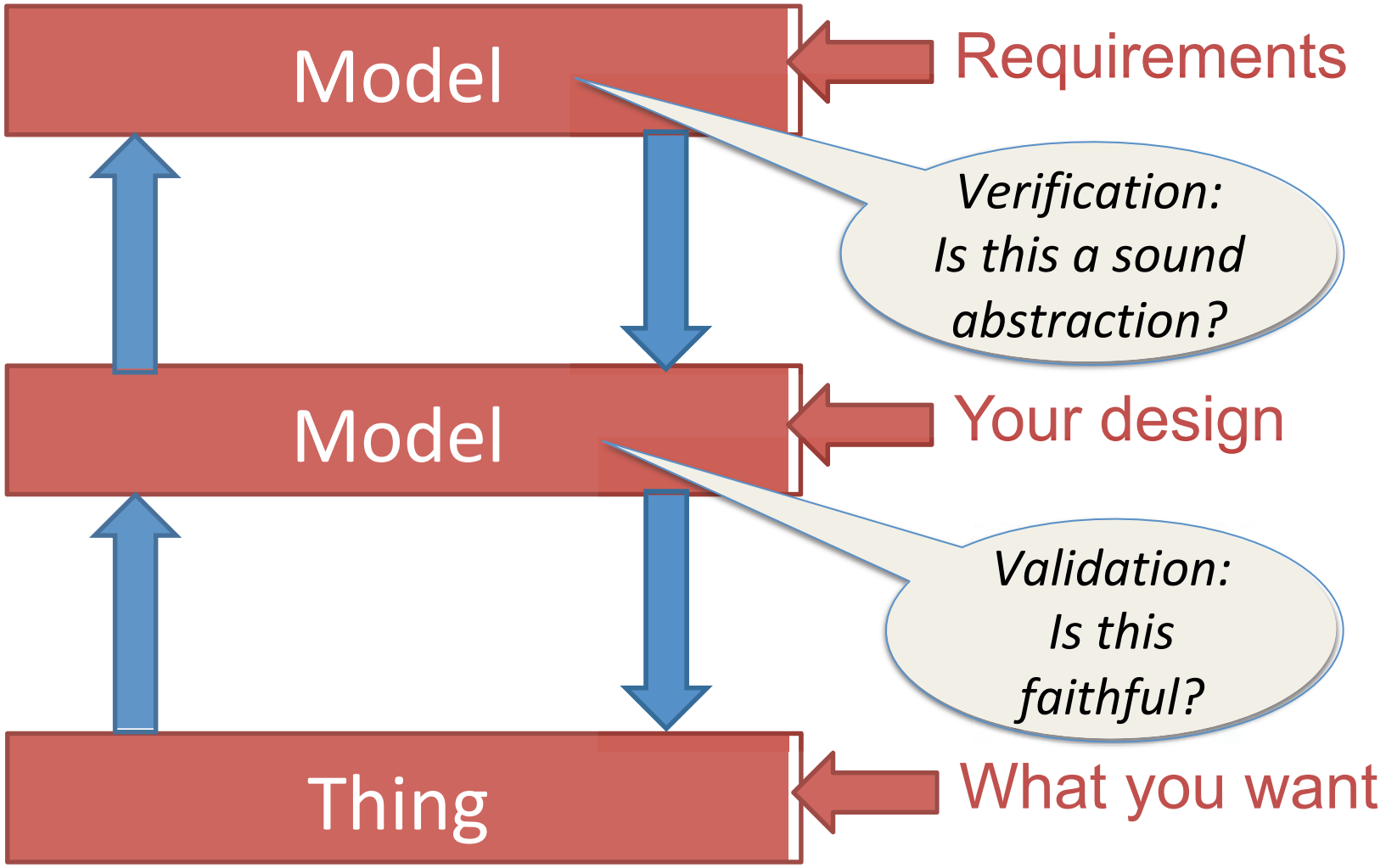
Is the question whether we can build a thing-in-itself where behavior matches that of our models (with high probability)?

Verification and Validation

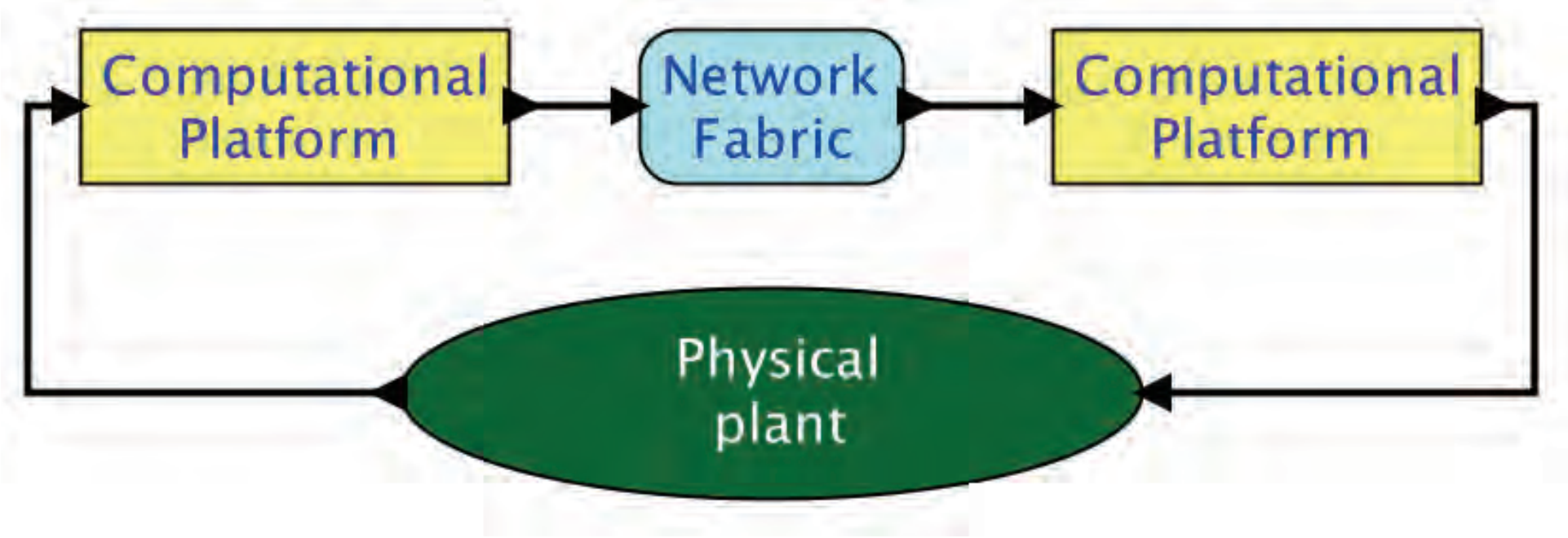
Per Boehm:

- Am I building the product right? (verification)
- Am I building the right product? (validation)

Verification and Validation



Cyber Physical Systems



What kinds of models should we build?

Software as a Model

Physical System

Model

```
/** Reset the output receivers, which are the inside receivers of
 *  the output ports of the container.
 *  @exception IllegalActionException If getting the receivers fails.
 */
private void _resetOutputReceivers() throws IllegalActionException {
    List<IOPort> outputs = ((Actor) getContainer()).outputPortList();
    for (IOPort output : outputs) {
        if (_debugging) {
            _debug("Resetting inside receivers of output port: "
                    + output.getName());
        }
        Receiver[][] receivers = output.getInsideReceivers();
        if (receivers != null) {
            for (int i = 0; i < receivers.length; i++) {
                if (receivers[i] != null) {
                    for (int j = 0; j < receivers[i].length; j++) {
                        if (receivers[i][j] instanceof FSMReceiver) {
                            receivers[i][j].reset();
                        }
                    }
                }
            }
        }
    }
}
```

Single-threaded imperative programs are deterministic models

Physics as a Model

Physical System

Image: Wikimedia Commons

Model

$$\dot{\mathbf{x}}(t) = \dot{\mathbf{x}}(0) + \frac{1}{M}\int_0^t \mathbf{F}(\tau)d\tau$$

Differential Equations are deterministic models

A major problem for CPS: combinations of deterministic models are nondeterministic

```
void initTimer(void) {
    SysTickPeriodSet(SysCtlClockGet() / 1000);
    SysTickEnable();
    SysTickIntEnable();
}
volatile uint timer_count = 0;
void ISR(void) {
    if(timer_count != 0) {
        timer_count--;
    }
}
int main(void) {
    SysTickIntRegister(&ISR);
    .. // other init
    timer_count = 2000;
    initTimer();
    while(timer_count != 0) {
        ... code to run for 2 seconds
    }
    ... // other code
}
```

$$\dot{\mathbf{x}}(t) = \dot{\mathbf{x}}(0) + \frac{1}{M}\int_0^t \mathbf{F}(\tau)d\tau$$

Image: Wikimedia Commons

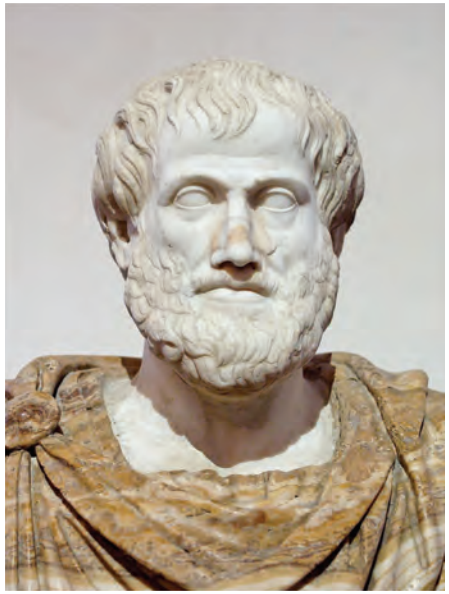
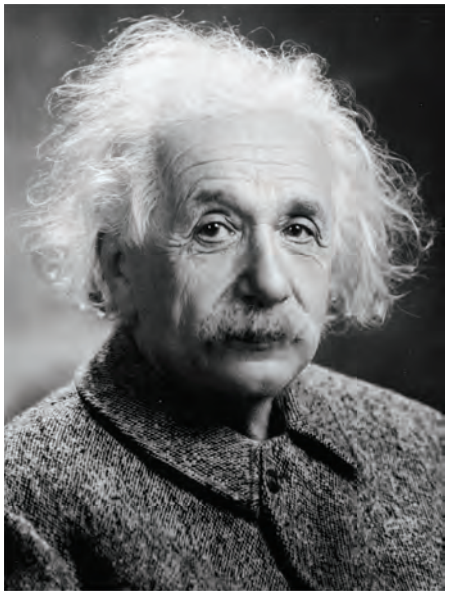
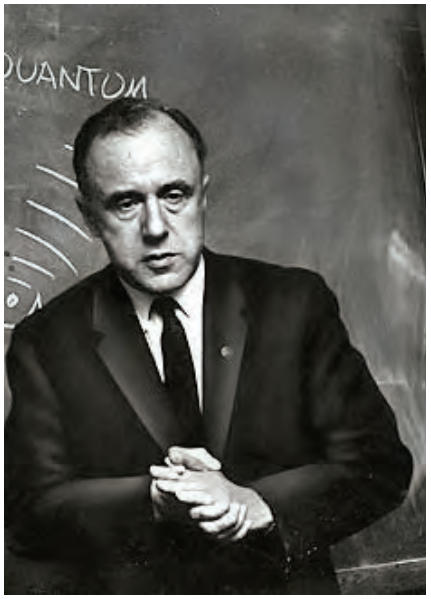
Consider Avionics

What is assurance?

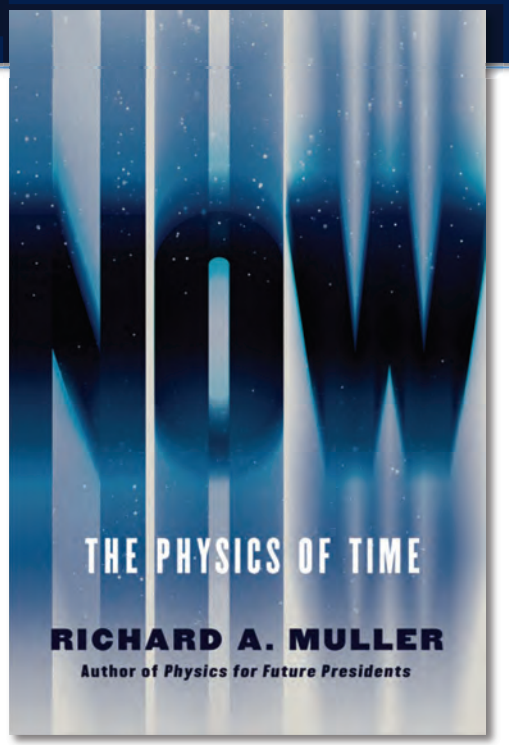
- Software is correct?
- Compiler is correct?
- Microprocessor is correct?

Correct execution of *correct* software provides little assurance.

What is Time?

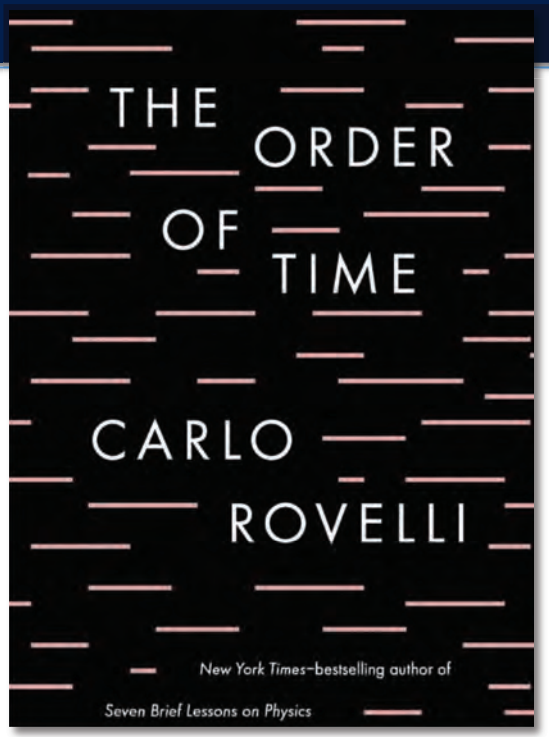
Change	Smooth	Relative	Discrete
			
Aristotle	Newton	Einstein	Wheeler

What is Time?



2016

Muller: Gives a theory of time that requires big black holes to collide somewhere near us to test it.



2018

Rovelli: “The nature of time is perhaps the greatest remaining mystery.”

How can we build models based on something we do not understand

?

Do we want scientific models or engineering models?

Real-Time Software

Scientific Models:

- Pipeline hazards
- Cache effects
- Interrupts
- DRAM variability
- Mutexes
- Interference
- Dynamic voltage/ frequency
- …

Engineering Models:

- Time
- Periodic tasks
- Deadlines

Do we have technologies today that deliver faithful physical realizations of such models?

Models in Time

Assume that "time" is about how a model *changes*.

Change may be:

1. Discrete: indivisible, atomic, an *event*.
2. Continuous: flow, motion

Time in Software

Physical System

Model

```
1  void initTimer(void) {
2      SysTickPeriodSet(SysCtlClockGet() / 1000);
3      SysTickEnable();
4      SysTickIntEnable();
5  }
6  volatile uint timer_count = 0;
7  void ISR(void) {
8      if(timer_count != 0) {
9          timer_count--;
10     }
11 }
12 int main(void) {
13     SysTickIntRegister(&ISR);
14     .. // other init
15     timer_count = 2000;
16     initTimer();
17     while(timer_count != 0) {
18         ... code to run for 2 seconds
19     }
20     ... // other code
21 }
```

Time in software is sequences of discrete steps

Time in Physical Systems

Physical System

Image: Wikimedia Commons

Model

$$\dot{\mathbf{x}}(t) = \dot{\mathbf{x}}(0) + \frac{1}{M}\int_0^t \mathbf{F}(\tau)d\tau$$

Time is the Newtonian continuum.

Relating Sequences to Newtonian Time

In modern microprocessors, determining how long it takes to execute a program is astonishingly difficult.

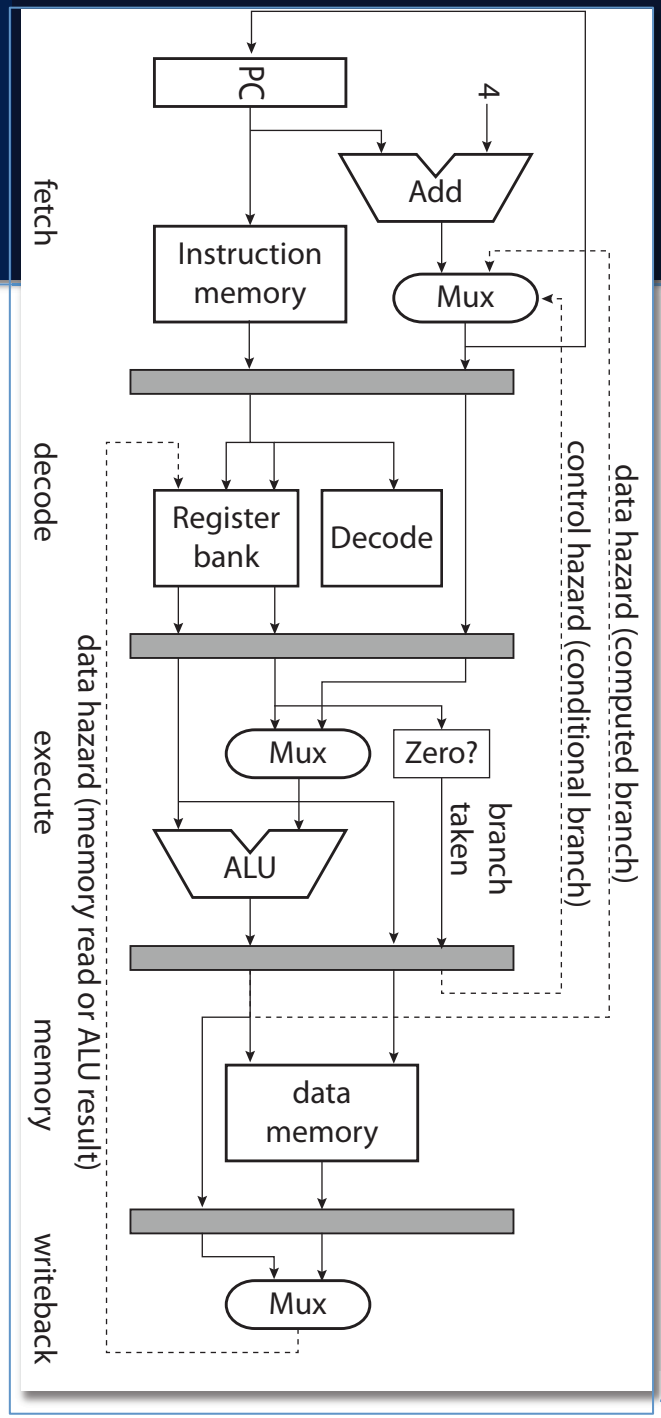
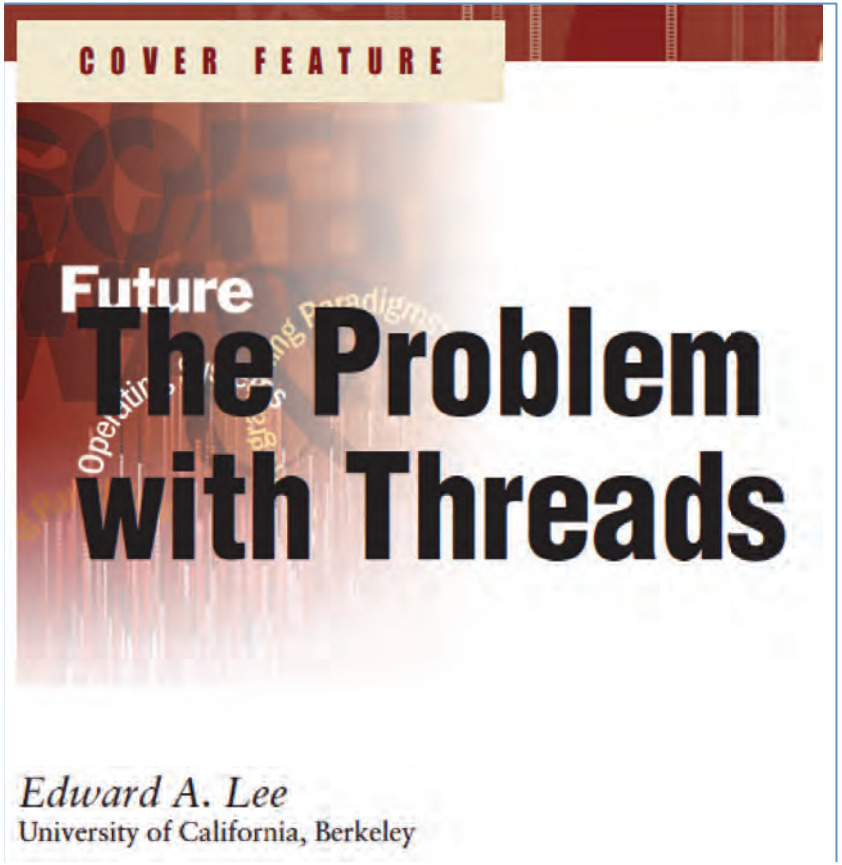


Image from Lee & Seshia, Introduction to Embedded Systems MIT Press, 2017

Messy Time

Time becomes a mess with interrupts and threads



Model

```
1  void initTimer(void) {
2      SysTickPeriodSet(SysCtlClockGet() / 1000);
3      SysTickEnable();
4      SysTickIntEnable();
5  }
6  volatile uint timer_count = 0;
7  void ISR(void) {
8      if(timer_count != 0) {
9          timer_count--;
10     }
11 }
12 int main(void) {
13     SysTickIntRegister(&ISR);
14     .. // other init
15     timer_count = 2000;
16     initTimer();
17     while(timer_count != 0) {
18         ... code to run for 2 seconds
19     }
20     ... // other code
21 }
```

IEEE Computer, May, 2006.

About Interrupts

"[M]any a systems programmer's grey hair bears witness to the fact that we should not talk lightly about the logical problems created by that feature"

- Edsger Dijkstra (1972)

Current Trends in Real-Time Software

- Model the details
- Analyze the models

Result is expensive, intractable models.

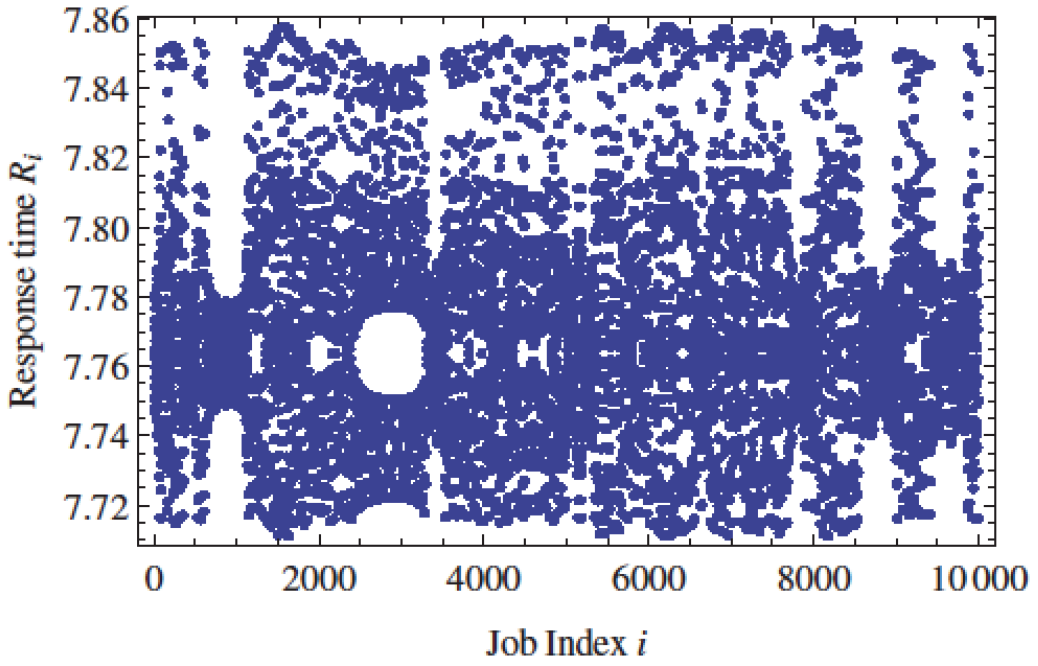


Fig. 15. Response time across jobs for the multi-resource scheduler with $R_s(i-1) = 7.76$ and $R_s(i-2) = 7.74$.

Even deterministic real-time models can lead to chaos.

[Thiele and Kumar, EMSOFT 2015]

Bring the cyber and the physical together a

Boom

```
void initTimer(void) {
    SysTickPeriodSet(SysCtlClockGet() / 1000);
    SysTickEnable();
    SysTickIntEnable();
}
volatile uint timer_count = 0;
void ISR(void) {
    if(timer_count != 0) {
        timer_count--;
    }
}
int main(void) {
    SysTickIntRegister(&ISR);
    .. // other init
    timer_count = 2000;
    initTimer();
    while(timer_count != 0) {
        ... code to run for 2 seconds
    }
    ... // other code
}
```

$$\dot{\mathbf{x}}(t) = \dot{\mathbf{x}}(0) + \frac{1}{M}\int_0^t \mathbf{F}(\tau)d\tau$$

Image: Wikimedia Commons

Changing the Question

Is the question whether our models describe the thing in itself (faithfully)?

Or

Is the question whether we can build a thing-in-itself where behavior matches that of our models (with high probability)?

Better Engineering Models for Real-Time Cyber-Physical Systems

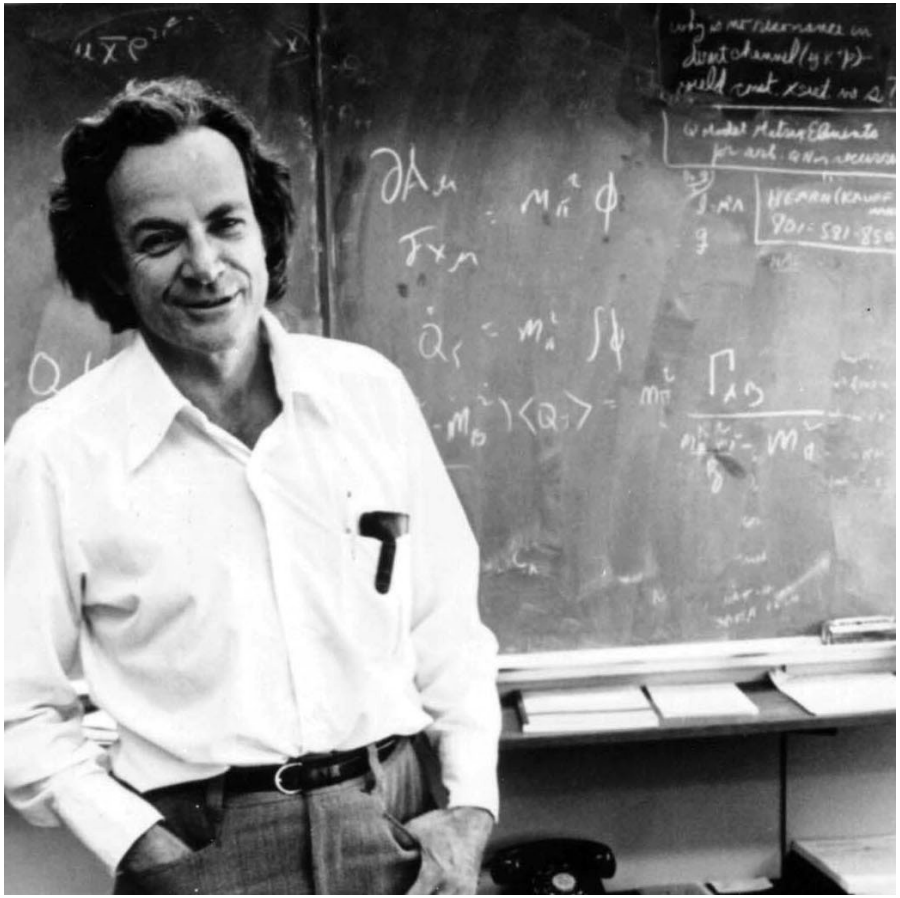
- Models of time
  - "Computing Needs Time," *Comm.of the ACM,* May 2009
- PTIDES: distributed real-time software
  - http://chess.eecs.berkeley.edu/ptides
- PRET: time-deterministic architectures
  - http://chess.eecs.berkeley.edu/pret

Together, these technologies show that deterministic distributed CPS models are practically realizable.

With these, one can, in principle, certify the software (a model) for a distributed real-time system.

Models and Thinking

A quote from Gleik, 1993: *Genius: The Life and Science of Richard Feynman*.



By Briola Giancarlo CC BY 3.0
https://it.wikipedia.org/w/index.php?curid=4508947

Models and Thinking

Models are not representations of our thoughts.

They are our thoughts.

Shaping our Thinking

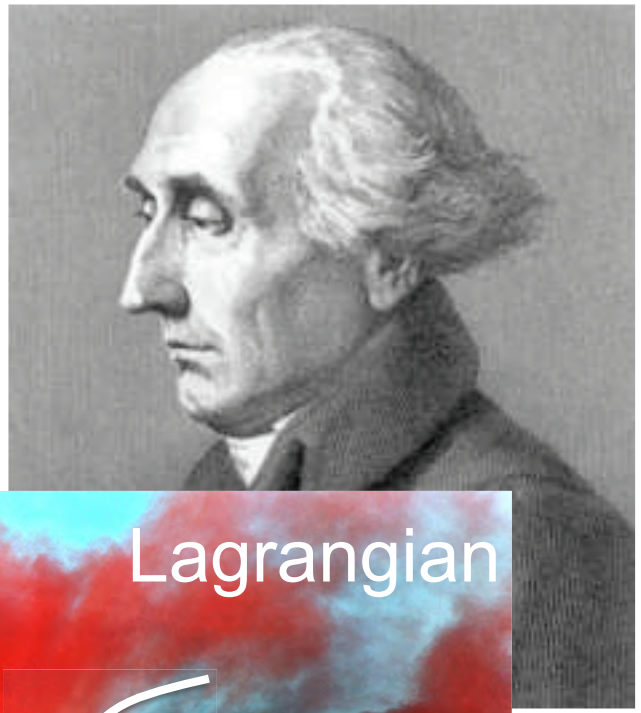
- Model of time
- Model of concurrency
- Model of computation
- Syntax
- Tool capabilities
- Mechanisms for modularity
- Modeling styles and patterns
- …

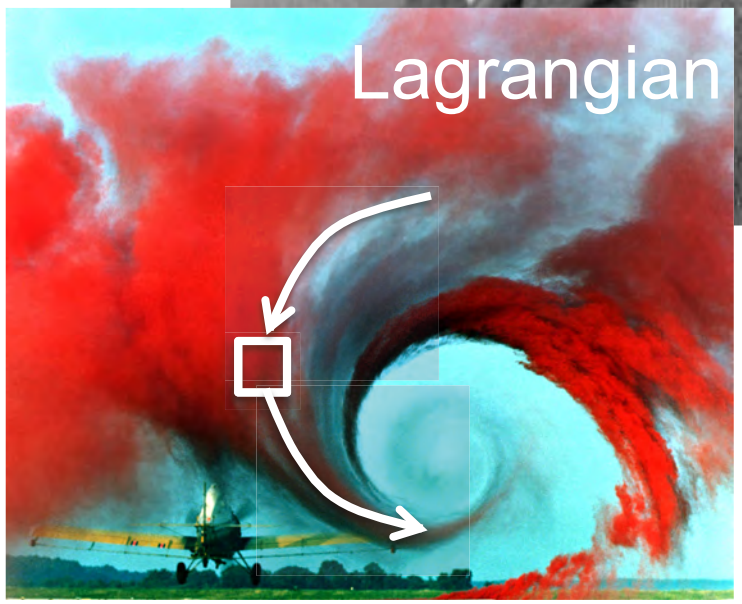
Semantics

Syntax

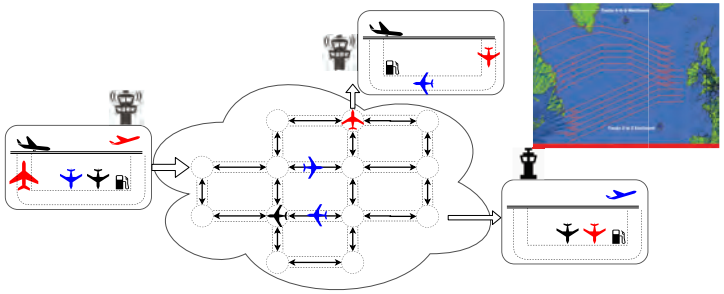
Pragmantics

Example: Modeling Fluid Flow

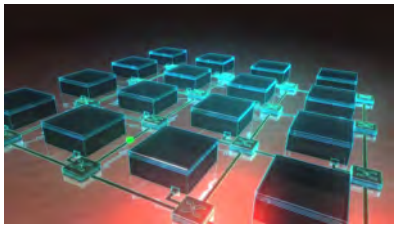




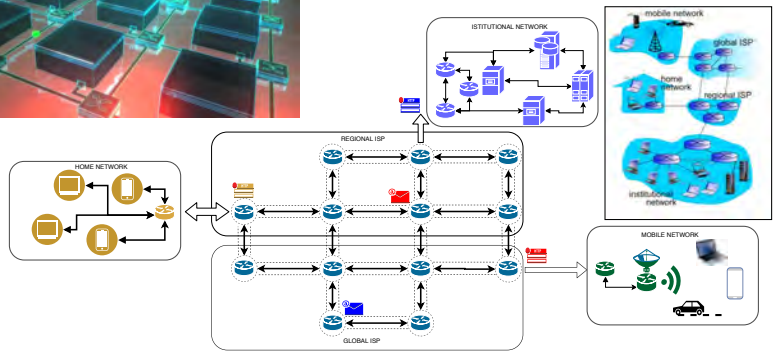
There are Many Flow Management Engineering Problems

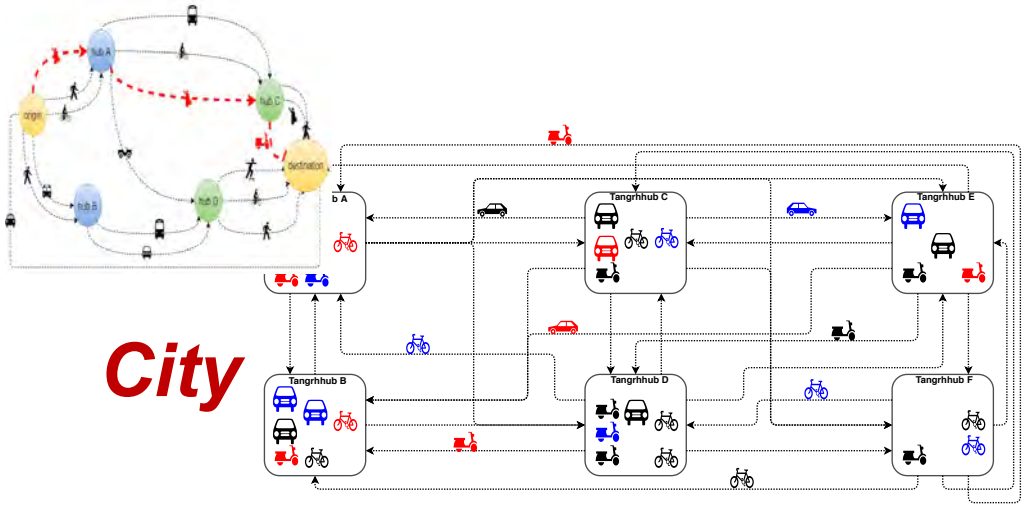


Air Space

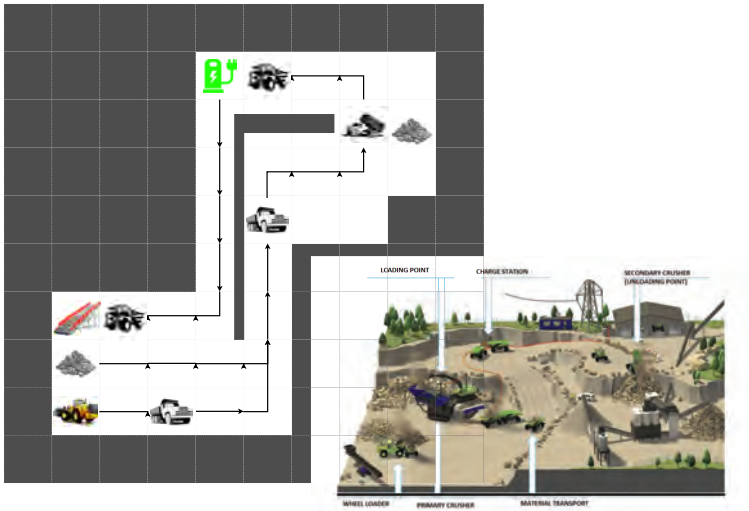


Networks



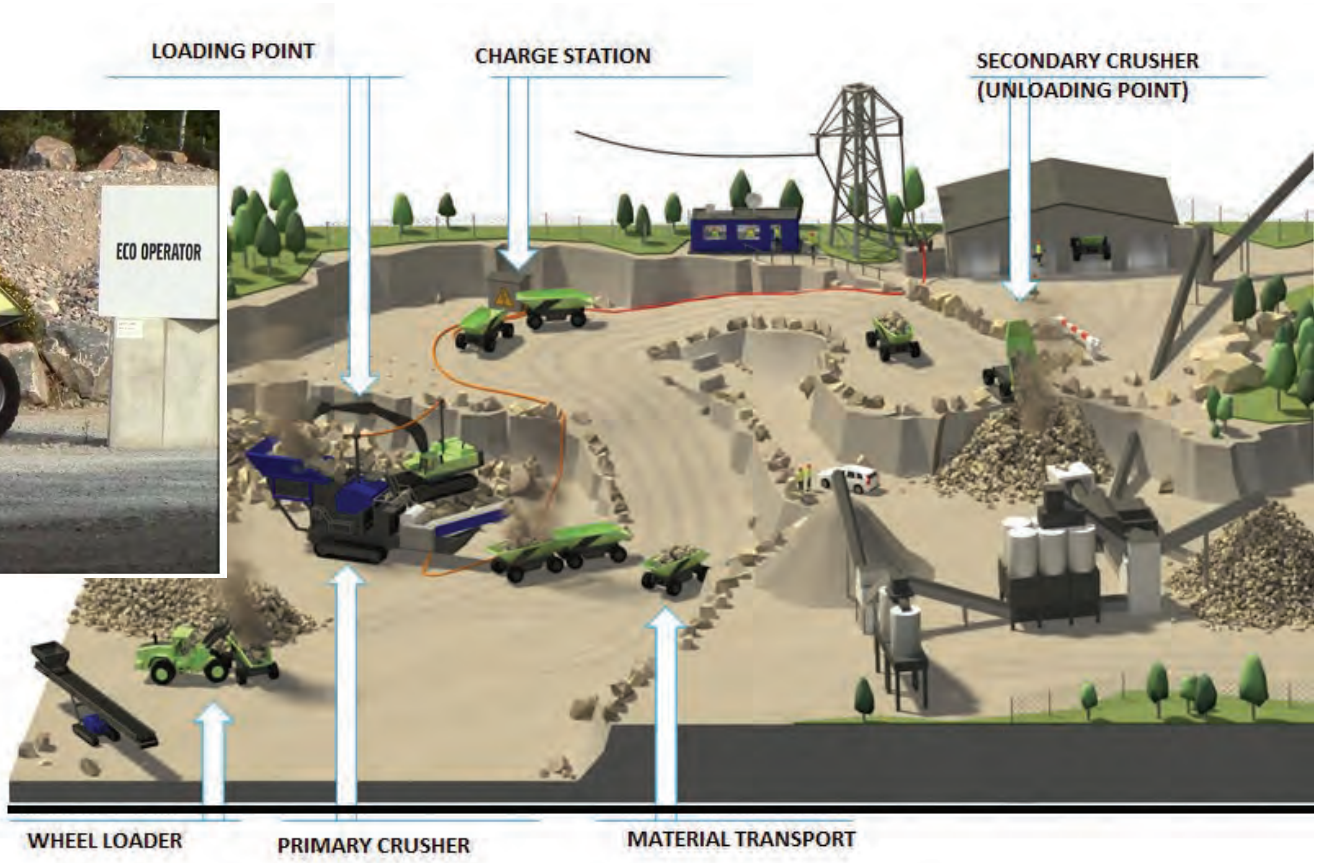


City



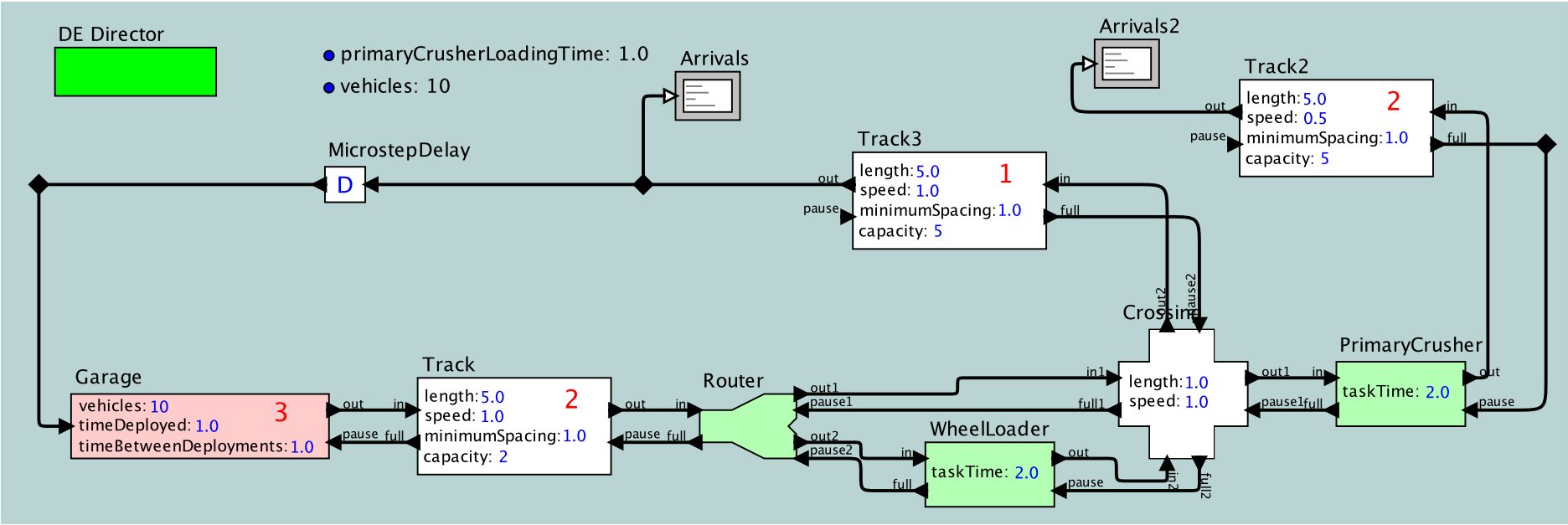
Quarry

Example: An Automated Quarry



Courtesy of Volvo CE

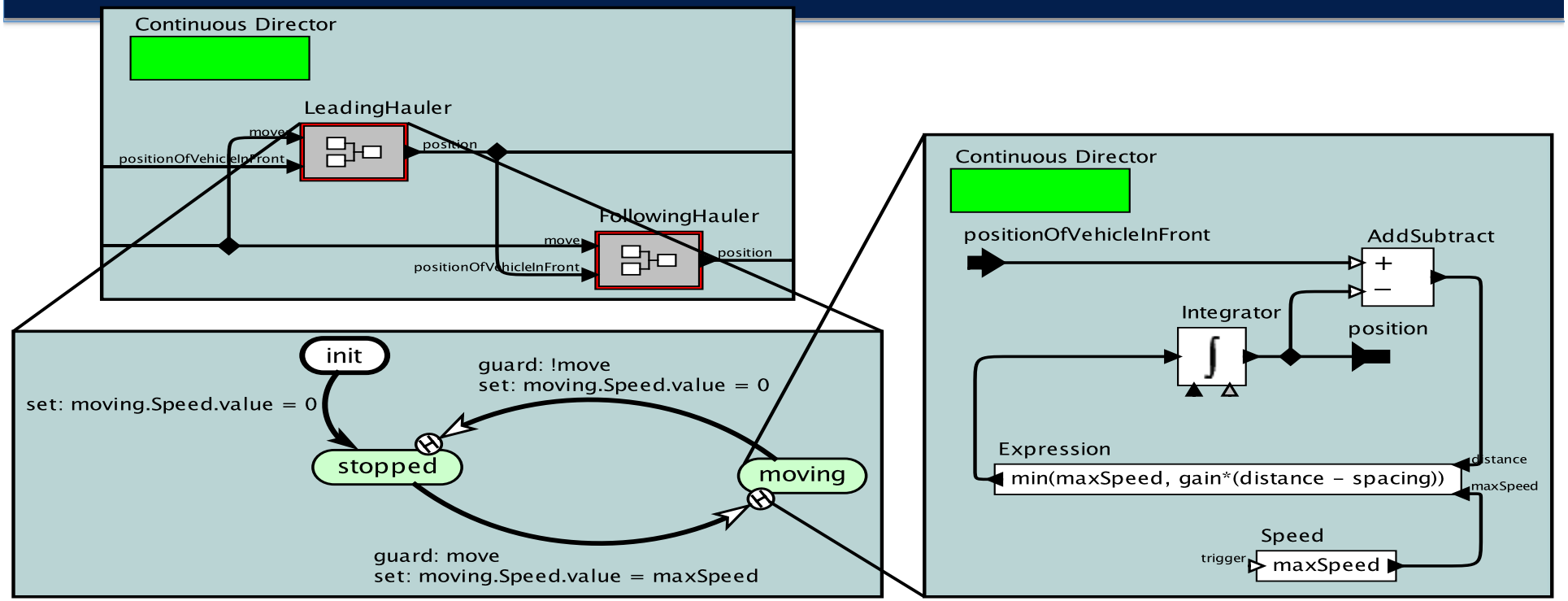
Eulerian Model
Actors are Tracks and Worksites



Use this model to study or design:

- Trajectory planning
- Resource optimization
- Affects of disruptions

Lagrangian Model
Actors are Haulers



Use this model to study or design:

- Collision avoidance
- Sensor performance
- Battery usage

Many Choices

- Discrete events or continuous dynamics?
- Environmental disturbances?
- Mechanical, sensor, actuator imperfections?
- Software timing?
- Network timing?
- …

Fallacies:
- More detail is better
- More “realistic” is better

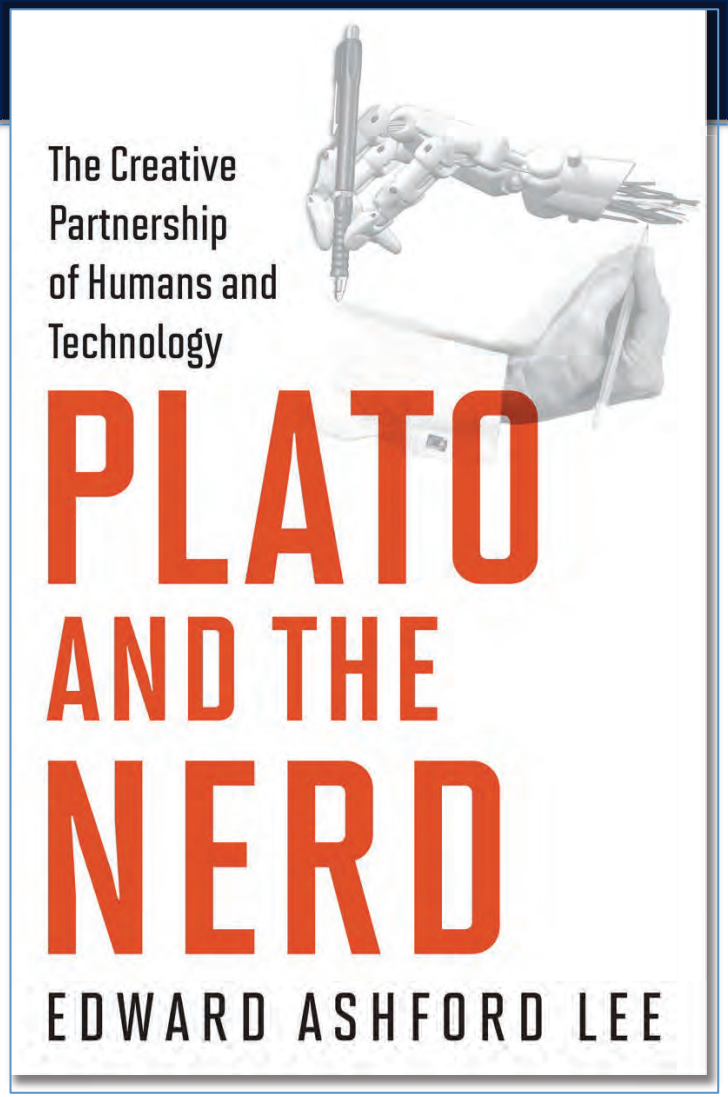
감사합니다

- In *science*, the value of a *model* lies in how well its behavior matches that of the physical system.

- In *engineering*, the value of the *physical system* lies in how well its behavior matches that of the model.

My message:
Do less science and more engineering.

Special thanks to *Marjan Sirjani*.



http://platoandthenerd.org