BLOOM
CALM
PROGRAMMING THE CLOUD

Joe Hellerstein
Peter Alvaro

AGENDA

- Brief research background from the BOOM project
  - http://boom.cs.berkeley.edu
- A taste of CS194-17, “Programming the Cloud” and the `bloom` language
- Some related work

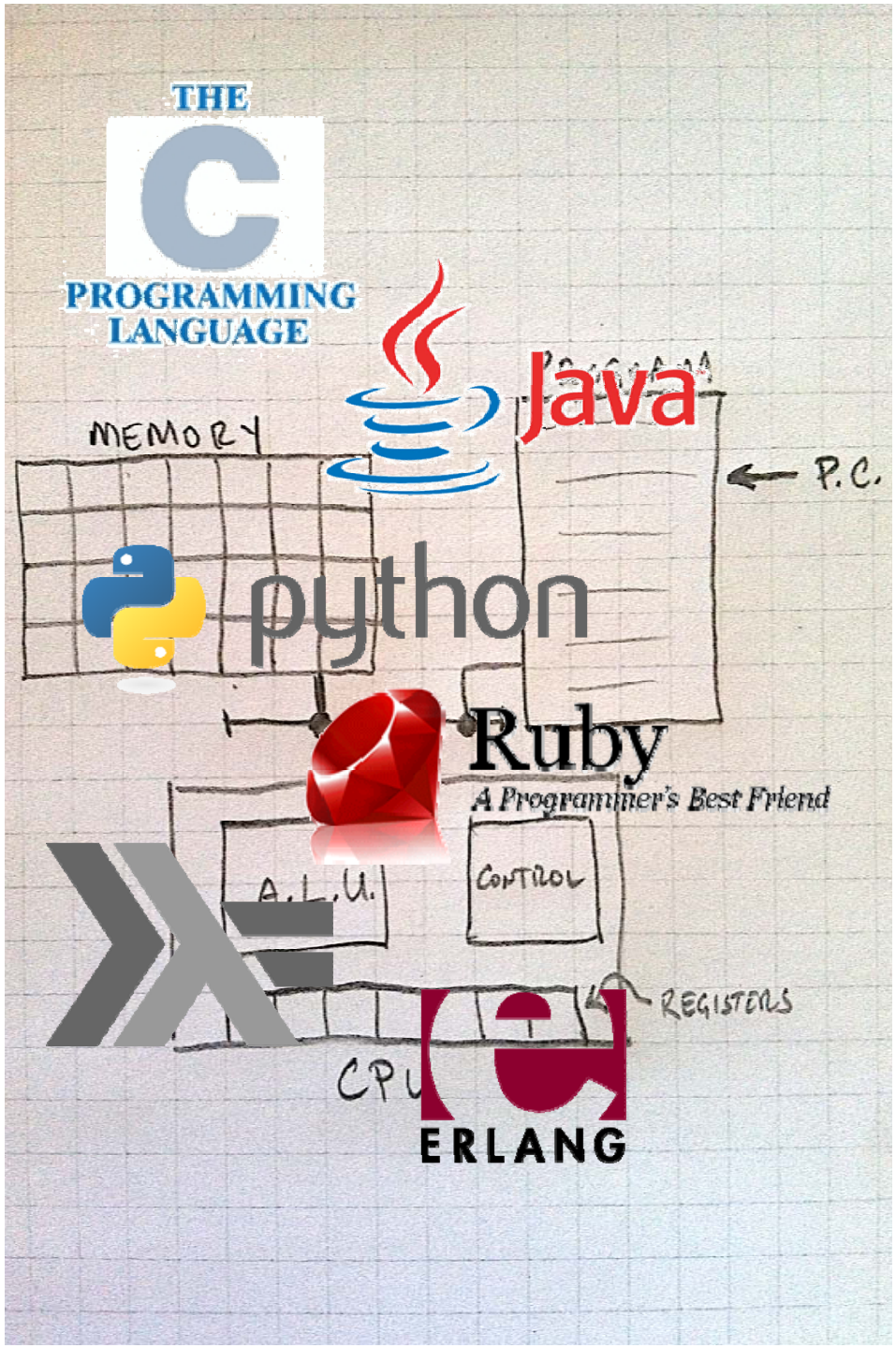
BOOM

In an era of cheap compute and ubiquitous data…

… **Productivity** is a key grand challenge in computing.

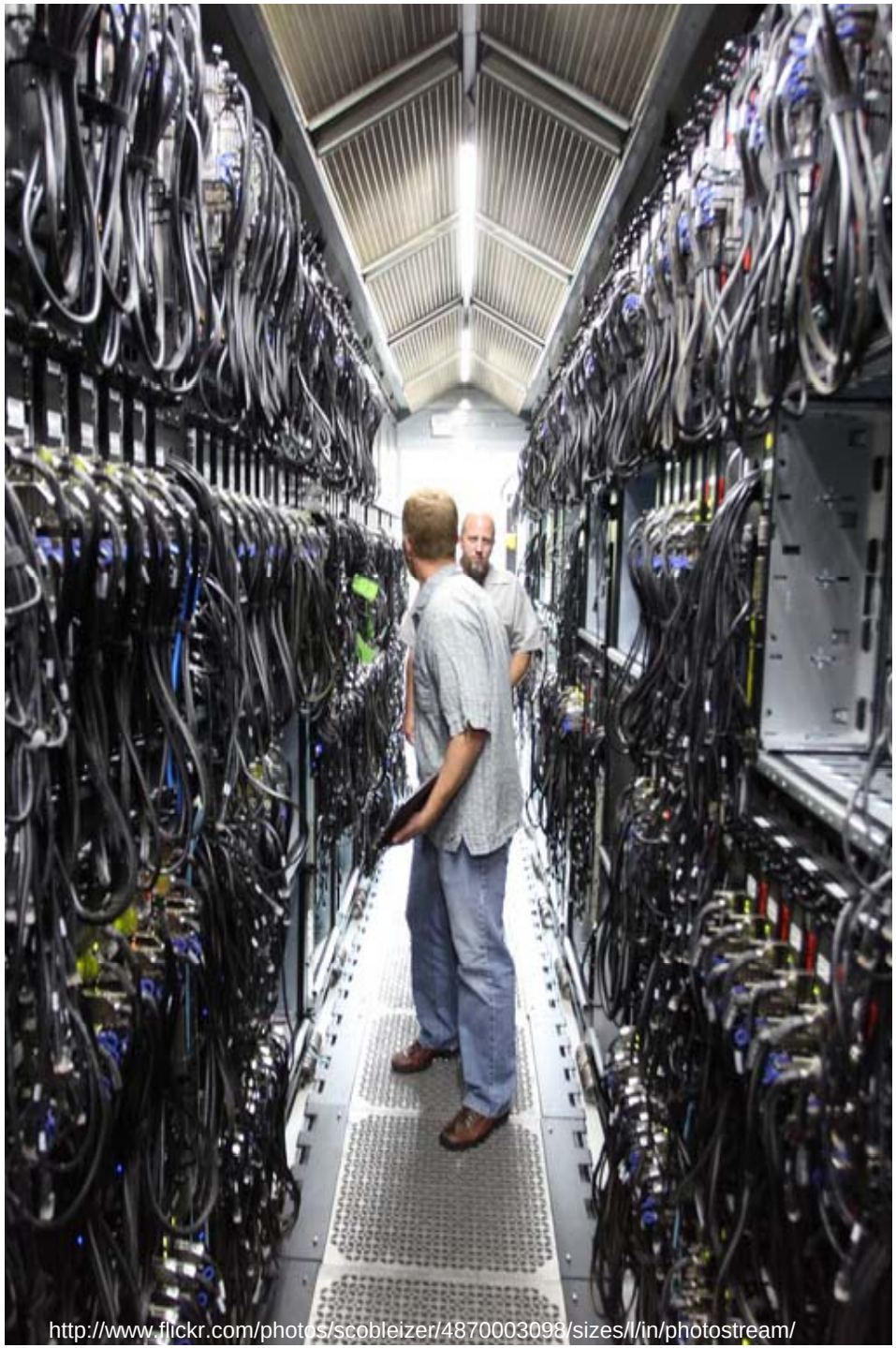
Berkeley **O**rders **O**f **M**agnitude project
OOM bigger systems, OOM less code.

Significantly improve productivity for developers of distributed systems.



THE
von NEUMANN
MACHINE

- ORDER
  - LIST of Instructions
  - ARRAY of Memory
- THE STATE
  - Mutation in time



DISTRIBUTED COMPUTING IS THE NEW NORMAL

- ORDER IS TOO COSTLY
  - Coordination
- THE STATE IS HEARSAY
  - Delay
  - Failure

DISORDERLY PROGRAMMING

STATE

- Order-insensitive objects

LOGIC

- Order-insensitive merge rules

IMPLICATION: KEEP CALM

- Asynchrony is irrelevant
- Replication is easy
- Coordination is unnecessary

Not always possible! But often.

- *Disorder by default*
- *Order is the exception*.

The CALM Theorem says when.

<~ bloom

- A disorderly distributed language as above
  - [Hellerstein, et al. CIDR11]
  - http://bloom-lang.org
  - Ruby prototype: Bud

```
% gem install bud
```

- Theoretical grounding: Dedalus
  - A logic for data, space and time
  - Model-theoretic (fully declarative) semantics
  - [Alvaro, et al. Datalog2.0-11, Datalog2.0-12]

- A disorderly distributed language as above
  - [Hellerstein, et al. CIDR11]
  - http://bloom-lang.org
  - Ruby prototype: Bud

```
% gem install bud
```

- Theoretical grounding: Dedalus
  - A logic for data, space and time
  - Model-theoretic (fully declarative) semantics
  - [Alvaro, et al. Datalog2.0-11, Datalog2.0-12]

MIT Technology Review

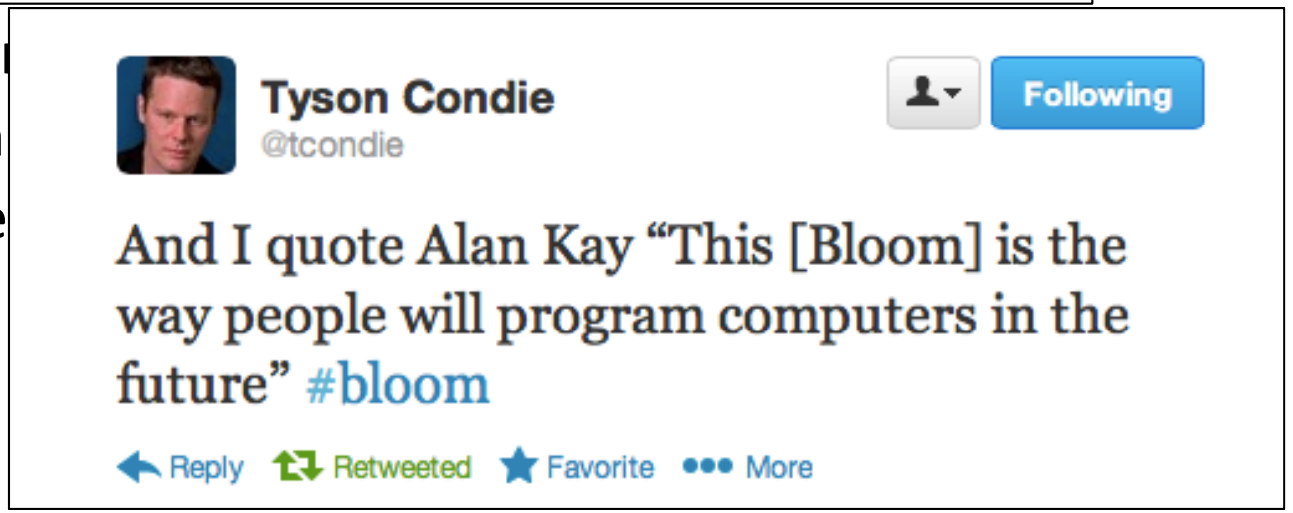
10 BREAKTHROUGH TECHNOLOGIES

- A diso
  - [He
  - htt
  - Rub
- Theor
  - A logic for data, space and time
  - Model-theoretic (fully declarative) semantics
  - [Alvaro, et al. Datalog2.0-11, Datalog2.0-12]

MIT Technology Review

10 BREAKTHROUGH TECHNOLOGIES

- A diso
  - [He
  - htt
  - Rub
- Theor
  - A logic fo
  - Model-th
  - [Alvaro, e



CS194-17 at Berkeley: Programming the Cloud

- Joe Hellerstein & Peter Alvaro
  - Now in its second offering.
- Tuesdays: Big Picture
  - lectures on distributed systems fundamentals
- Thursdays: Hands On
  - live-coding in Bloom
- We'll do a bit of a blend today…

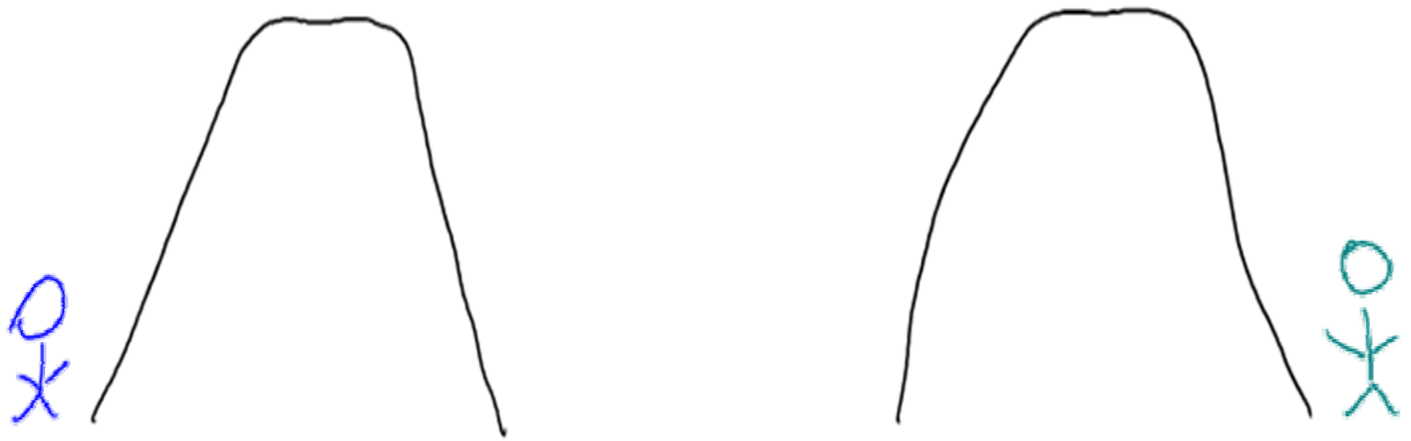
Lessons for Today

1. Communication as Rendezvous in Space & Time
2. The Duality of Communication and Storage
3. Assessing the need for Coordination protocols
   - CALM program analysis

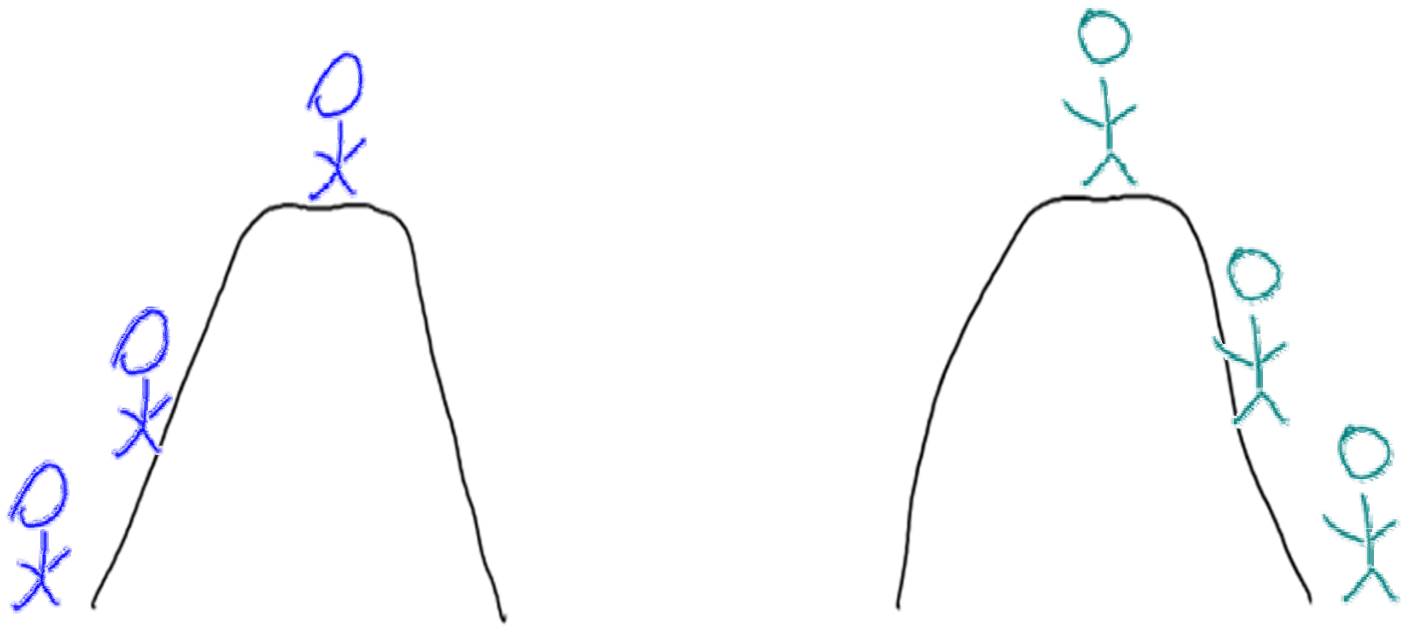
Lessons for Today

1. **Communication as Rendezvous in Space & Time**
2. The Duality of Communication and Storage
3. Assessing the need for Coordination protocols
   - CALM program analysis

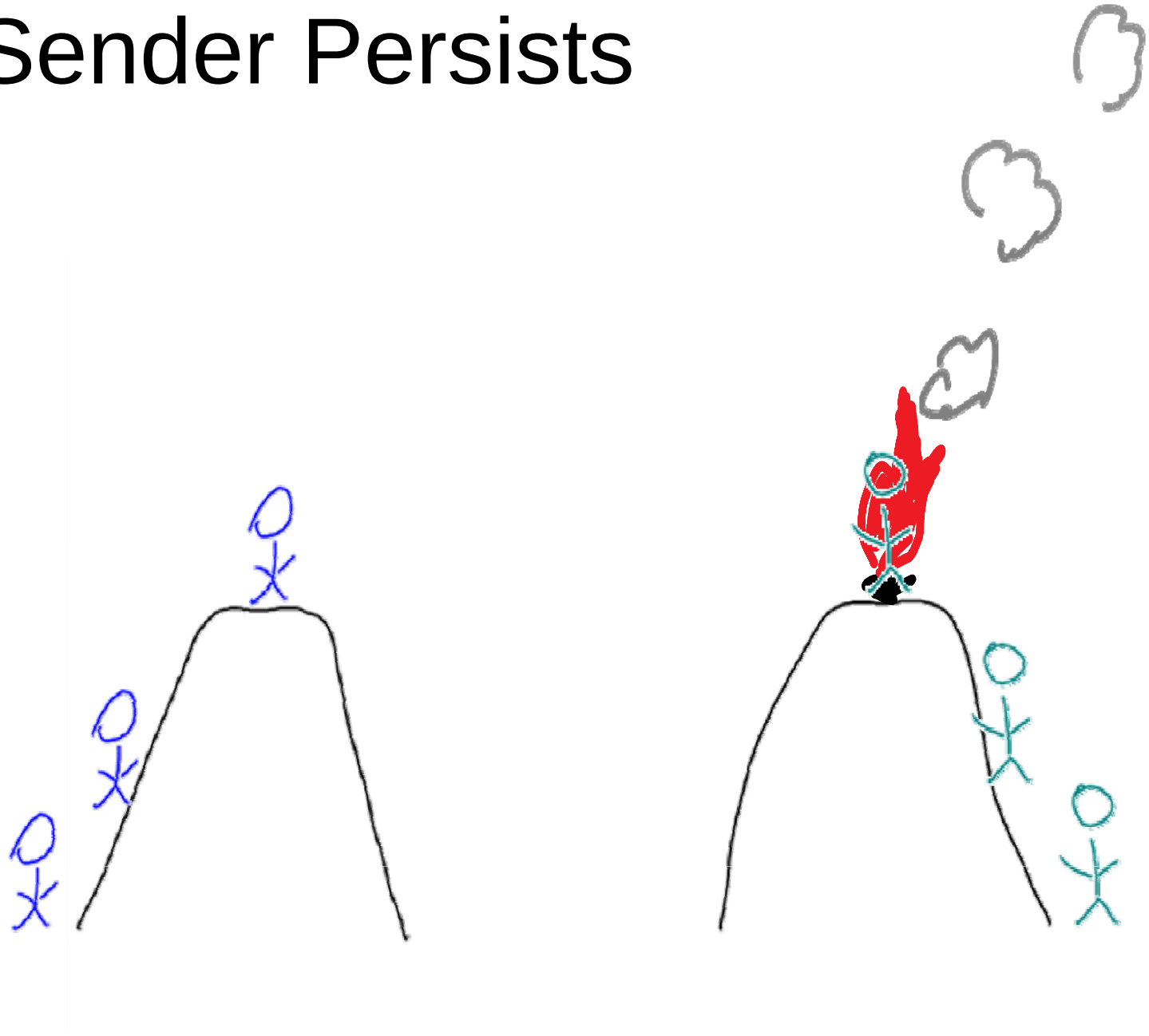
The Land of Two Mountains



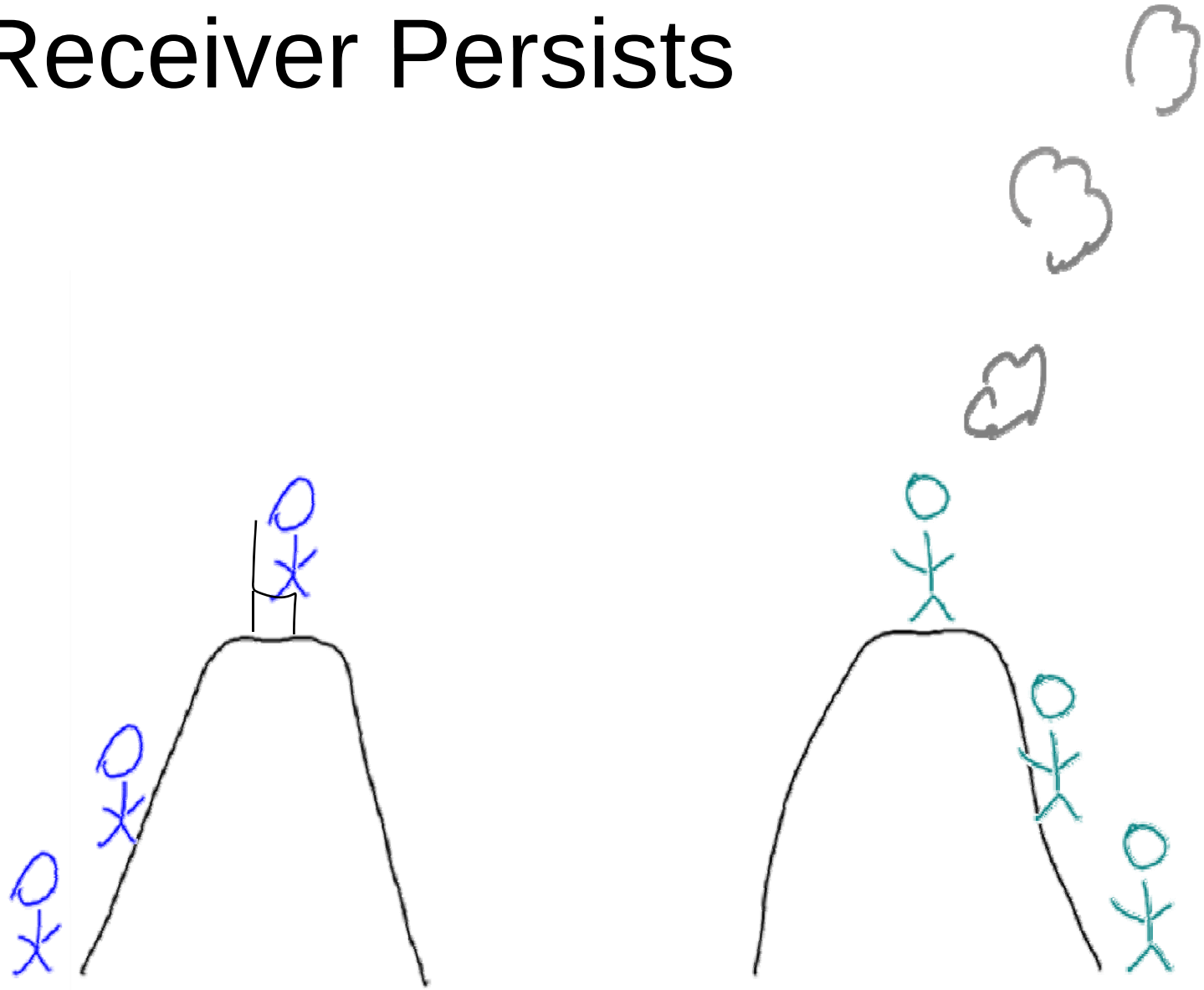
Rendezvous by Luck (Smoke Signals)



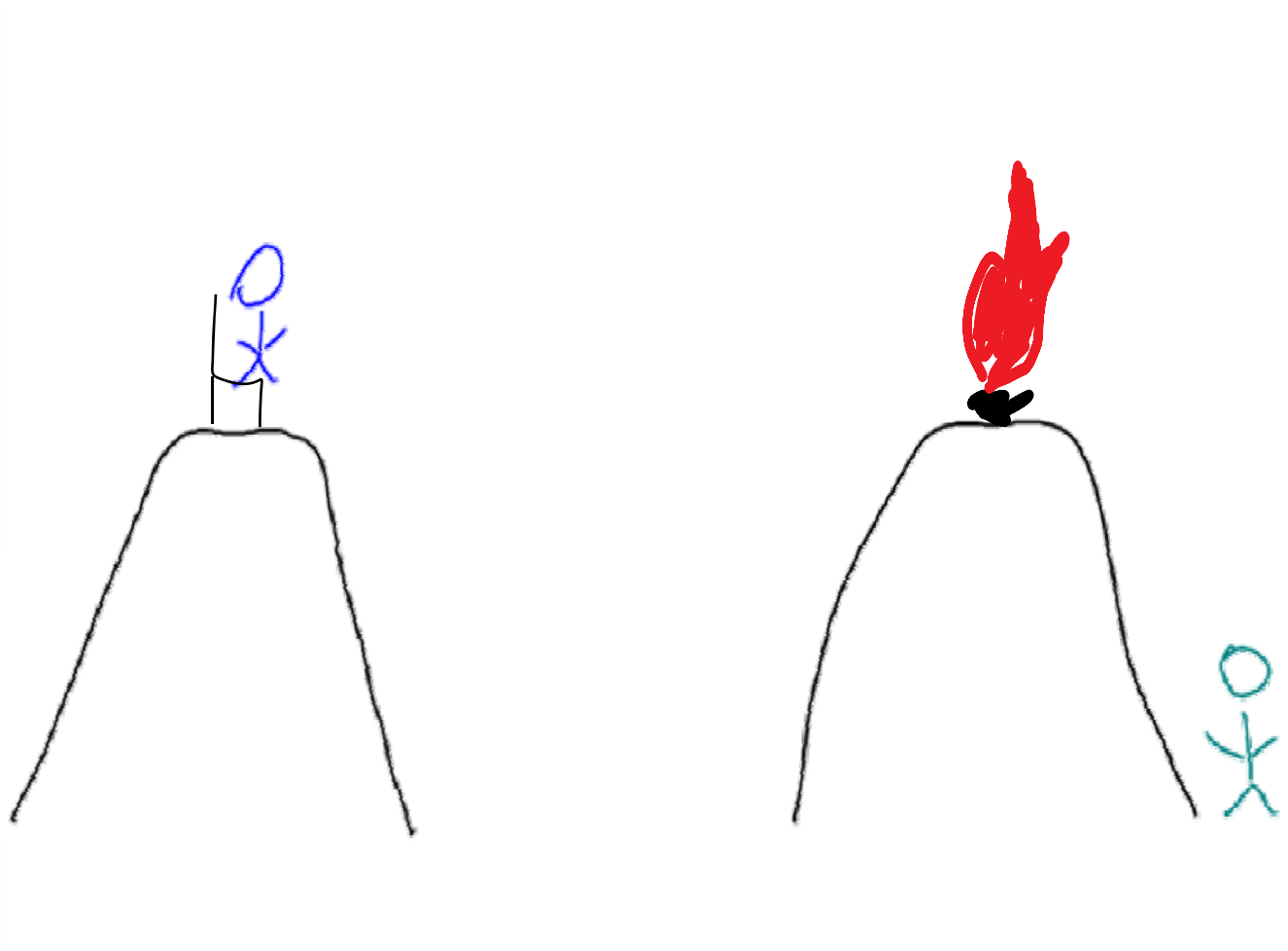
Sender Persists



Receiver Persists



Both Persist



Lessons for Today

1. Communication as Rendezvous in Space & Time
2. **The Duality of Communication and Storage**
3. Assessing the need for Coordination protocols
   - CALM program analysis

Lessons for Today

1. Communication as Rendezvous in Space & Time
2. The Duality of Communication and Storage
3. **Assessing the need for Coordination protocols**
   - **CALM program analysis**

Directions for Thought

- Thm (CALM): Consistency As Logical Monotonicity
  - <= : *Distributed code that's **monotonic** will be "eventually consistent" without coordination.*
    - Corollary: *It is sufficient to use coordination only to "guard" the non-monotonic statements in a program.*
  - => : *Any eventually consistent program is in some fundamental way monotonic.*
- Said differently:
  - "Thank you for all the Paxos, Dr. Lamport. Do I need it?"
  - Or perhaps better: "What is time for? Must I spend it?"
- [Hellerstein, SIGMODRecord 3/10; Ameloot PODS11, ICDT12, Marczak Datalog 2.0-12]
- **Realized in practice via Bloom/Budplot.**

More Results

- http://boom.cs.berkeley.edu
- http://bloom-lang.org

Materials for this talk:

- https://github.com/programthecloud/ptcrepo/tree/gh-pages/demo

BOOM TEAM


joe hellerstein


david maier


ras bodik


alan fekete


peter alvaro


peter bailis


neil conway


bill marczak


haryadi gunawi


sriram srinivasan


Joshua rosen


emily andrews

andy hutchinson

Key Results 1

- **BOOM Analytics** [Alvaro, et al. Eurosys '10]
  - HDFS rebuilt in Overlog, the predecessor to Bloom, with HA and scale-out
  - Hadoop scheduler as well
- **Bloom^L:** Beyond sets/tables [Conway, et al. SoCC '12]
  - Extensions for natural monotone data types like counters, vector clocks, KVS with commutative merges
  - Safe mappings between these types
- **Blazes:** Coordination analysis of streaming services [Alvaro, et al. In process]
  - Grey-box: bring CALM analysis to popular streaming systems like Storm
  - White-box: more fully automated stream analysis in the Bloom context
- **Correct, Composable Concurrent Editing** [Conway, et al. In process]
  - Google-Doc style concurrent editing remains a black art
    - Operational Transforms
  - Lattices underlie a lot of the intuition
  - Bloom^L provides a rich language for composing lattices and traditional data
  - Automated analysis of correctness

Key Results 2

- **Consistency and Causality in the Wild (Bailis et al.)**
  - Probabilistically Bounded Staleness [VLDB ‘12]
  - Dangers of Causal Consistency and a Solution [SoCC ‘12]
  - HAT, not CAP: Towards Highly Available Transactions [HotOS ‘13]
  - Bolt-On Causal Consistency [SIGMOD ‘13]

Summing Up

- Distributed? Disorderly by default.
  - Logic and Lattices in Space and Time
- The Duality of Communication and Storage
  - Unifying the two linguistically makes for nice code
- Assessing the need for Coordination protocols
  - CALM leads to straightforward program checks in Bloom
  - Points to games we can play in other languages/systems
  - Many interesting questions remain