Distributed Systems

LIKE IT OR NOT

http://l42.org/JAE

Theo Schlossnagle

CEO @Circonus
Twitter: @postwait

CIRCONUS

Beginning of Software Mass Production.

- Ship
- Install
- Run Disconnected.

Things got connected.

- We built connected software like disconnected software.
  - Either thick with updates (new versions) available online.
  - Or it was thin with all logic on **the server** side.

Distributed Applications (examples)

The Web MVC

MC on the server

V In the browser

PoS Applications

Inventory on main server

Inventory, sales on client

HPC

Coordinated progress:

Central jobs, disitributed work.

Distributed Backends Changed Everything

- Ratio of

  Affected users
  to
  Node failures

The Mysteries of Distributed Failure

Some things are
best described
through poetry

It's not DNS

There's no way it's DNS

It was DNS

- SSBroski

Progression of Understanding

1990
- Single user
- Single system

1995
- Distributed users
- Single system

2000
- Single user*
- Distributed systems

2005
- Distributed users
- Distributed systems

* Lack of tech and understanding to understand both distributed users and systems.

Distributed Systems Are Unavoidable

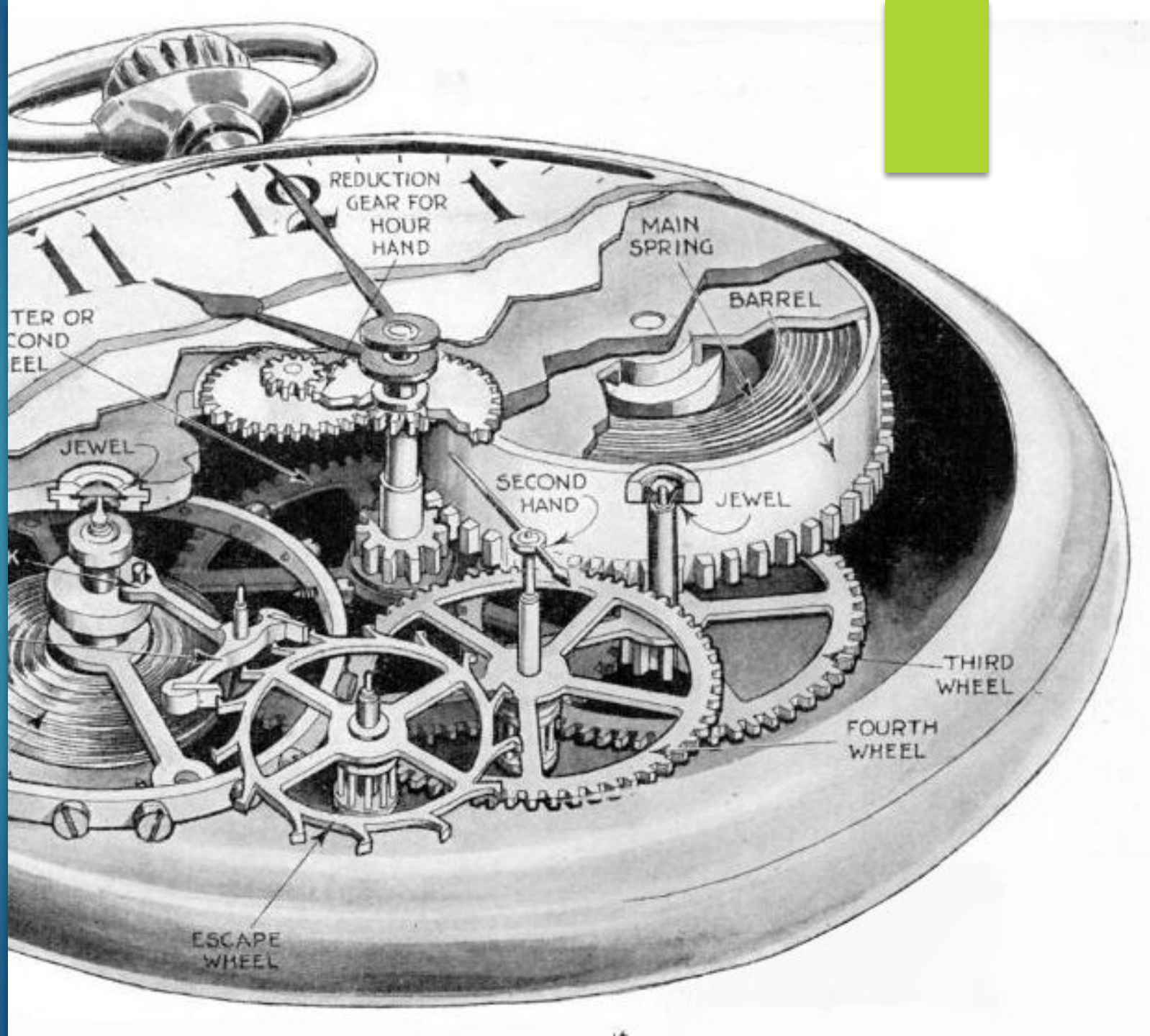
- Almost every system you build today will be distributed.
- Most single systems have multiple processors of different architectures with different clock speeds (think GPU); the distributed nature is mostly hidden.

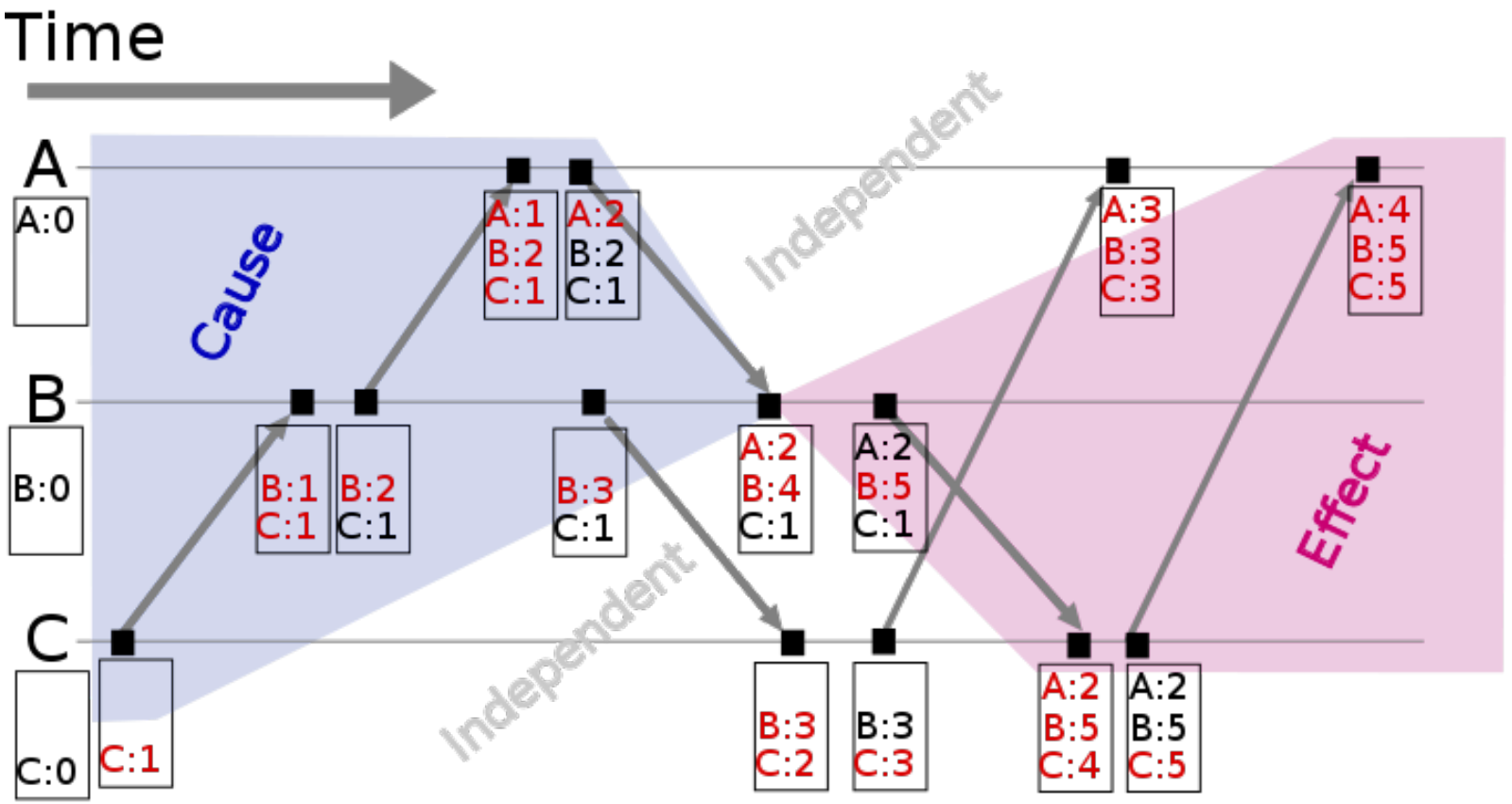
#1
Clocks & Time

- With 1 clock,
  you know the time.
- With 2+ clocks,
  you never know the time.

- You must think
  distributed.

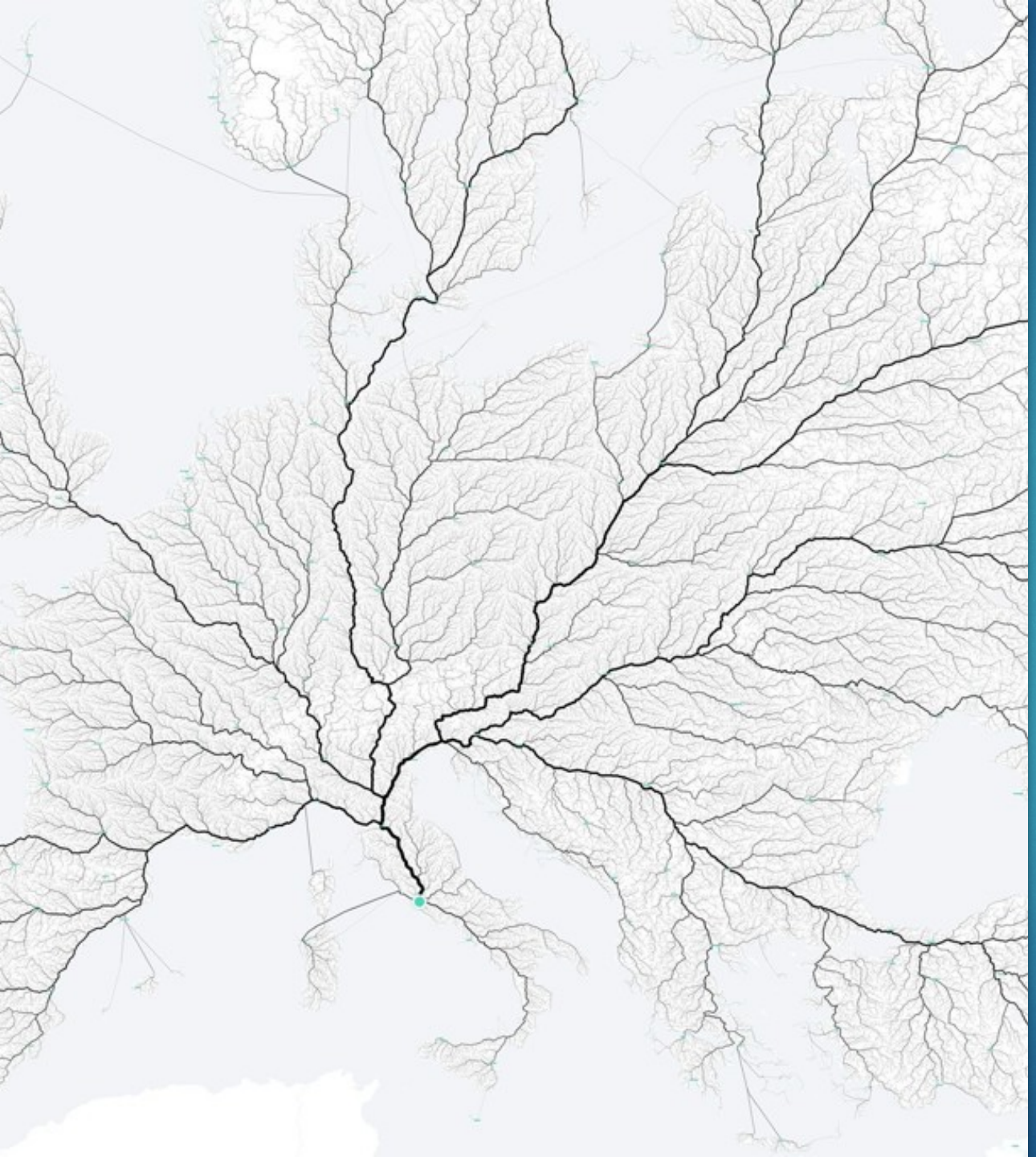
Leslie Lamport, Turing Award Winner:
http://l42.org/HgE
Colin Fidge:
http://l42.org/IgE





Thinking about causality

http://l42.org/lwE



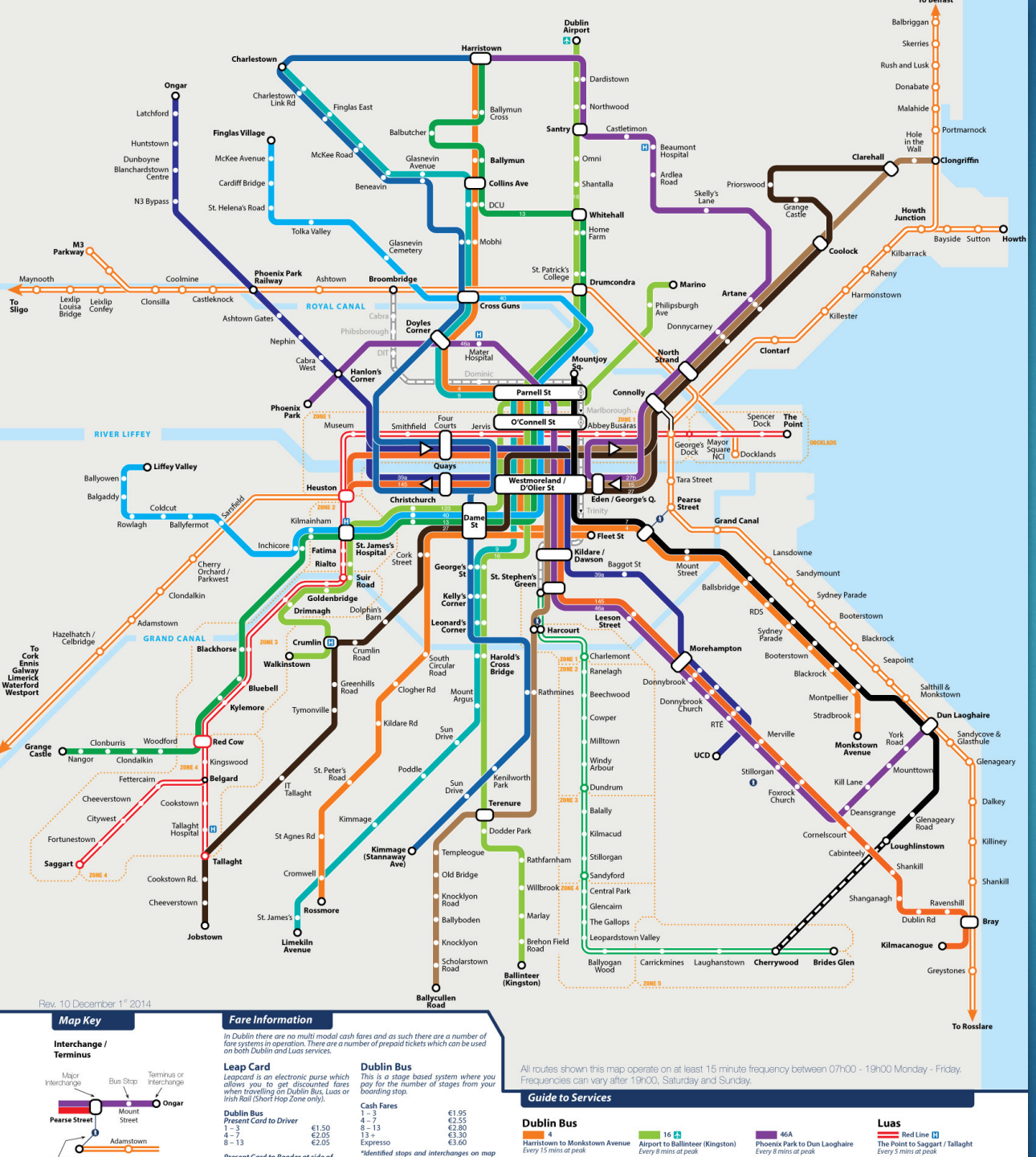
#2
Causal thinking, but with all possible initial states.

- That debugging step of understanding prior state still exists:
  - more possible states,
  - harder to constrain, and
  - no clocks (#1)

http://l42.org/HAE

#3
One person's failure, is another's Byzantine tragedy.

- Arguably reasonable isolated behavior can have pathologically bad system behavior.



Pathological Timing Failures

- The most common failure conditions are timeout related (in my experience)
- Mismanaged and awkwardly aligned timeouts can cause pathological and unstable situations.

#4 Understand Consensus

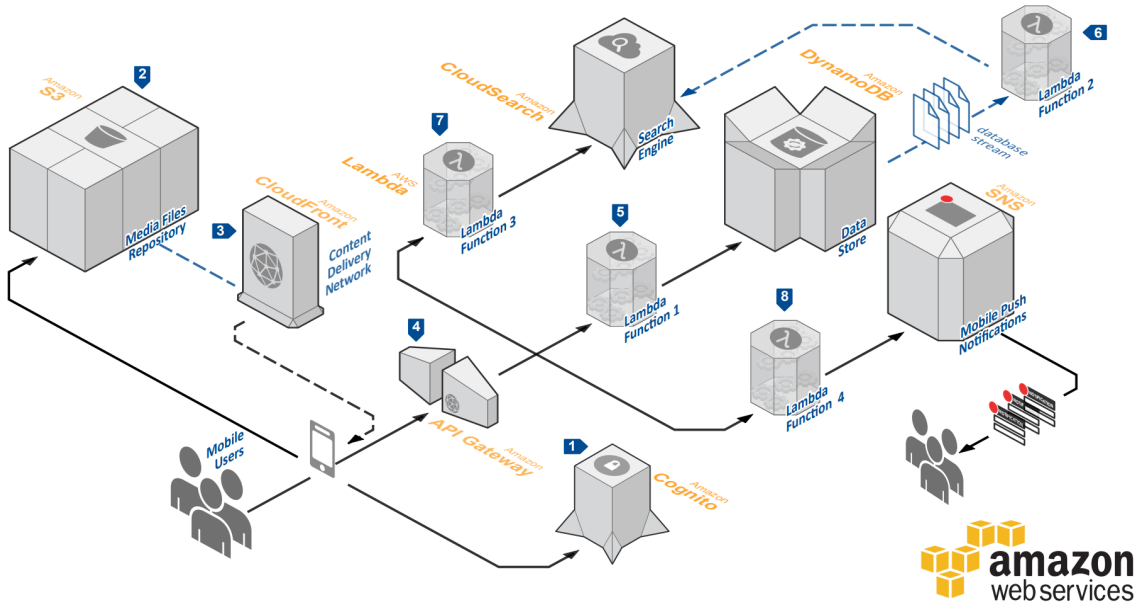
Paxos
- **Lamport**
  http://l42.org/IQE

Raft
- **Ongaro & Ousterhout**
  http://l42.org/IAE

VS
- **Birman**
  http://l42.org/HwE

Microservices
SOA
Cloud

The democratization of useful services has shifted the burden of understanding distributed systems from “distributed systems engineers” to every developer everywhere.



Is it all worth it?

- Modular development
- Language domains

- Security domains
- Availability domains
- Resiliency domains

Many Distributed Situations Make Little Initial Sense

Pathology

- ***Pathology***:
  concerned with diagnosis of issues via post-event analysis
  (coopted from medicine)

- ***Pathological***:
  involving, caused by, or the nature of a physical or mental disease
  or
  obsessive or compulsive

Thank You!

http://l42.org/JAE