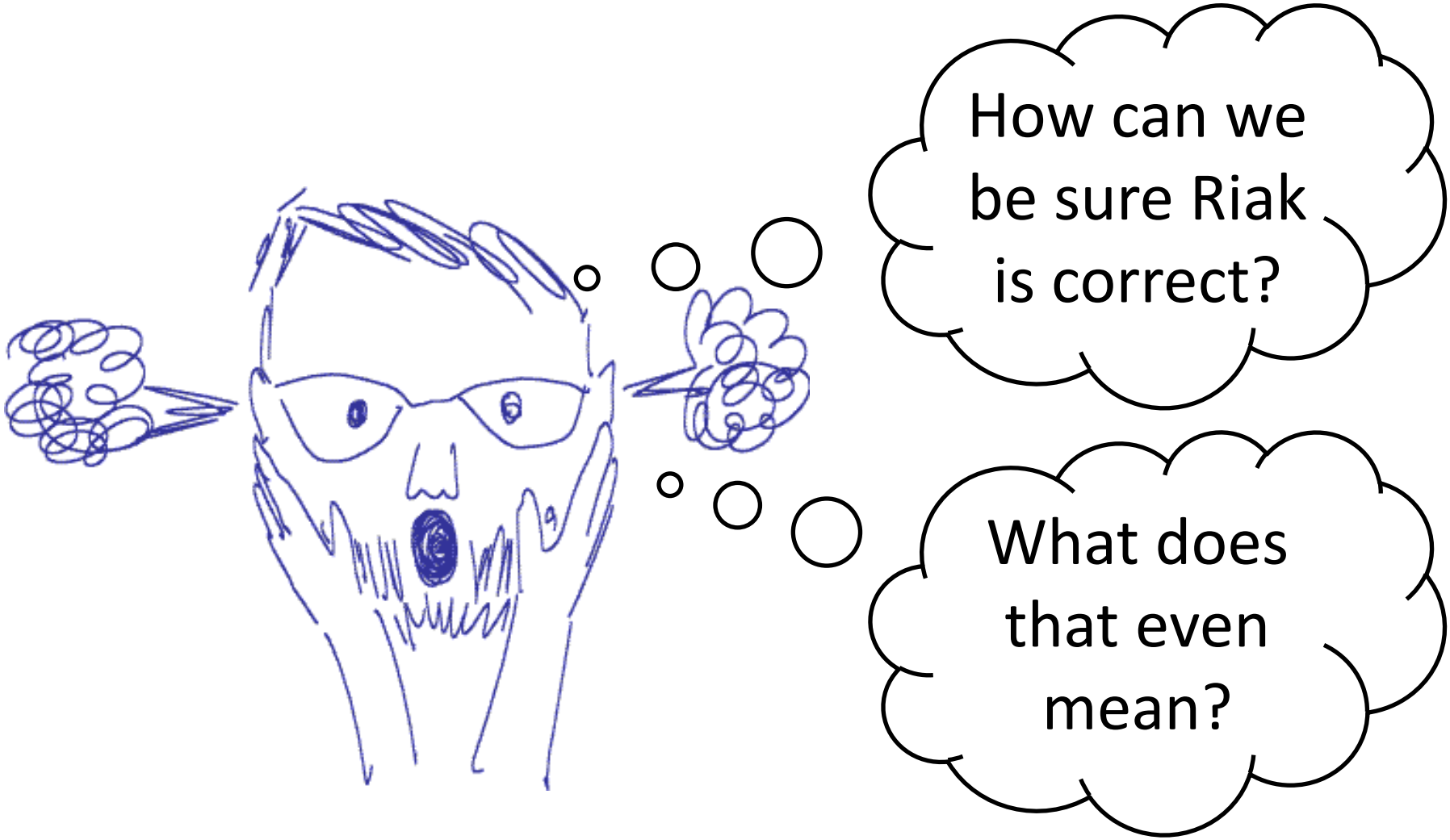
The Properties of 

John Hughes
Chalmers University/Quviq AB, Gothenburg
(with Scott Lystig Fritchie, Jon Meredith, Dave Smith)

- Distributed
- Scalable
- Replicated
- Fault-tolerant
- High availability
- Low latency
- No SQL—just keys and values

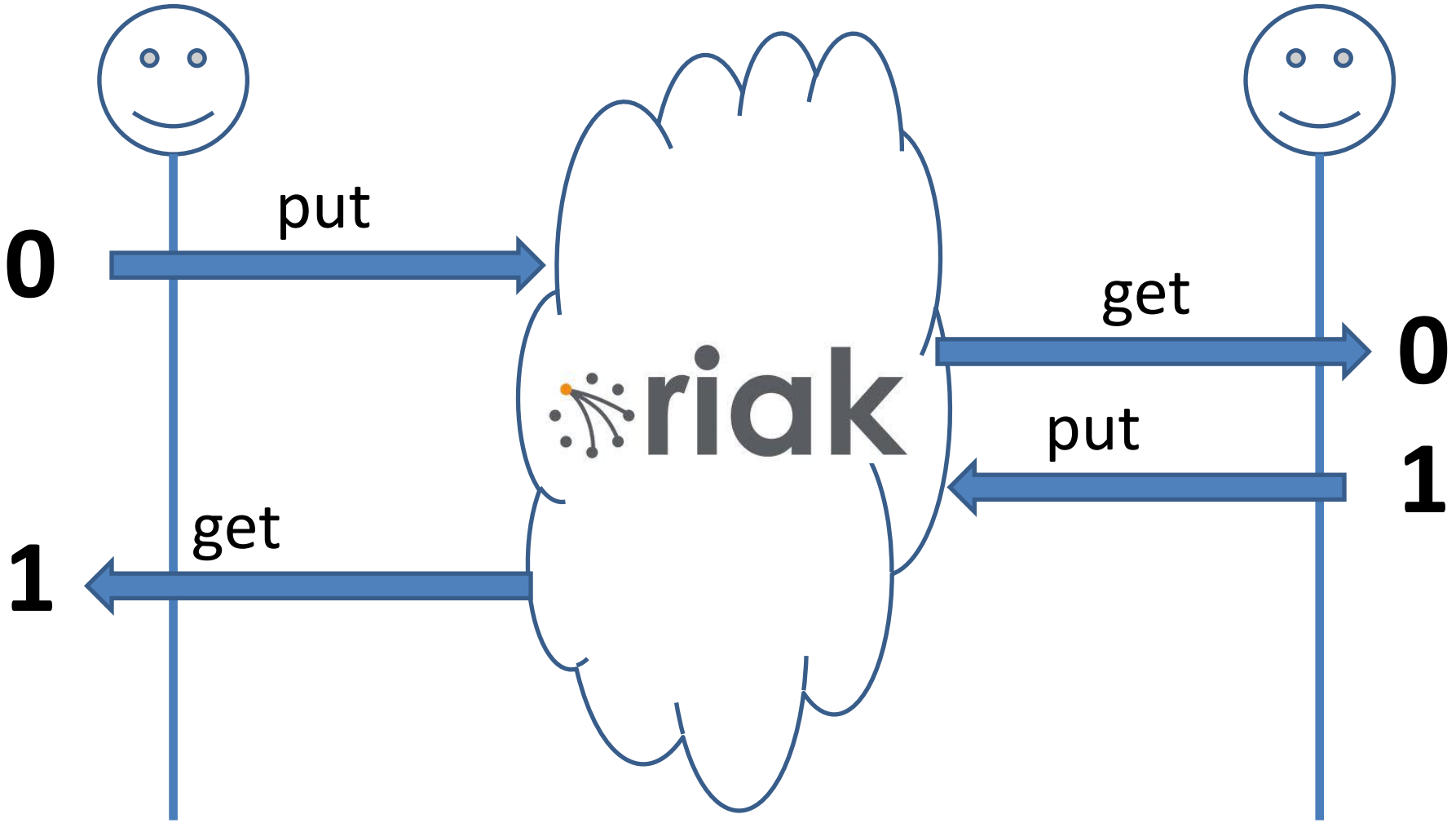


QuickCheck!

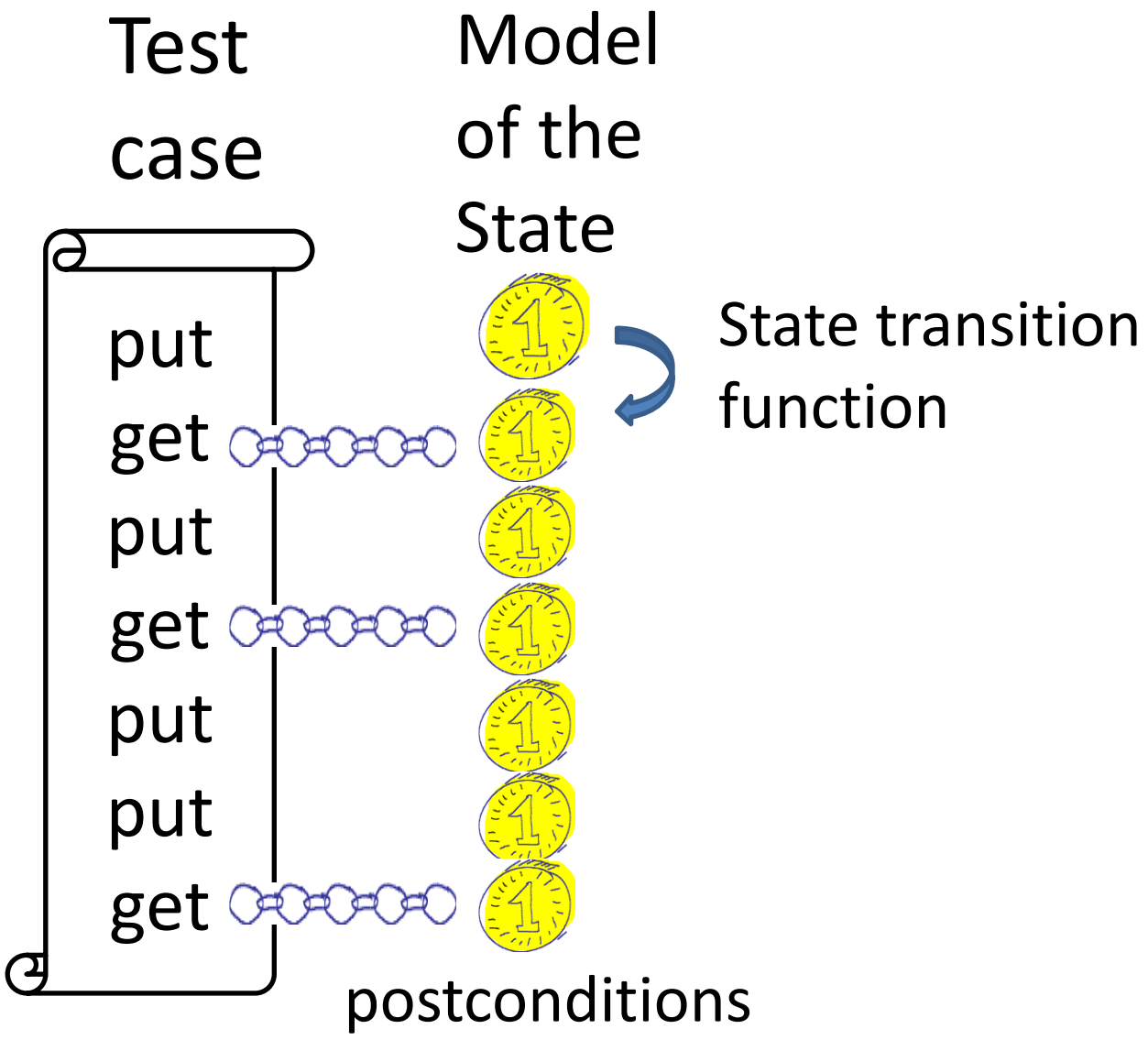
Keep It Simple!

one  one 

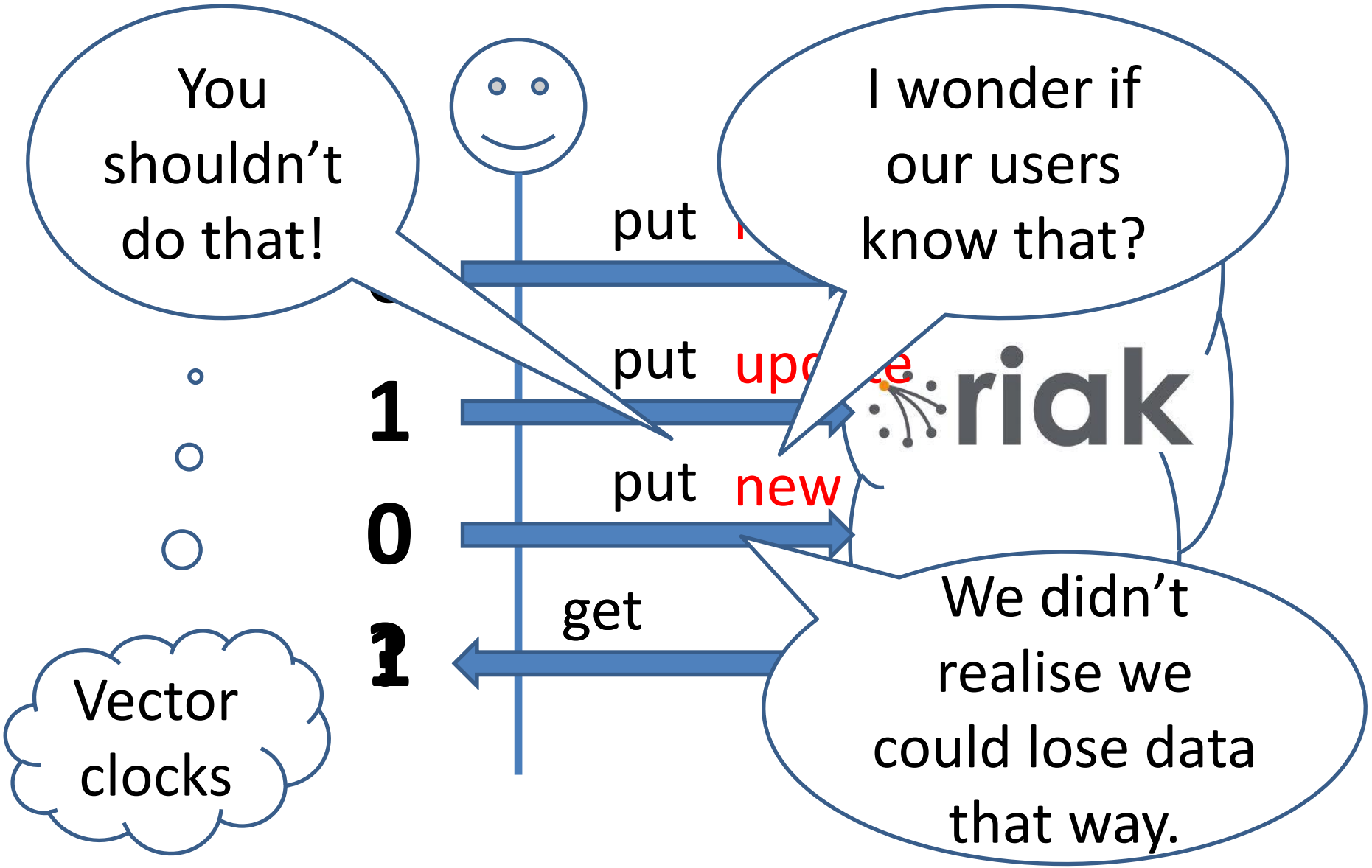
Put and Get



Quick Check


Test case
Model of the State
put
get
put
get
put
put
get
State transition function
postconditions

Example


You shouldn't do that!
I wonder if our users know that?
put
put
1
put new
0
get
1
riak
Vector clocks
We didn't realise we could lose data that way.

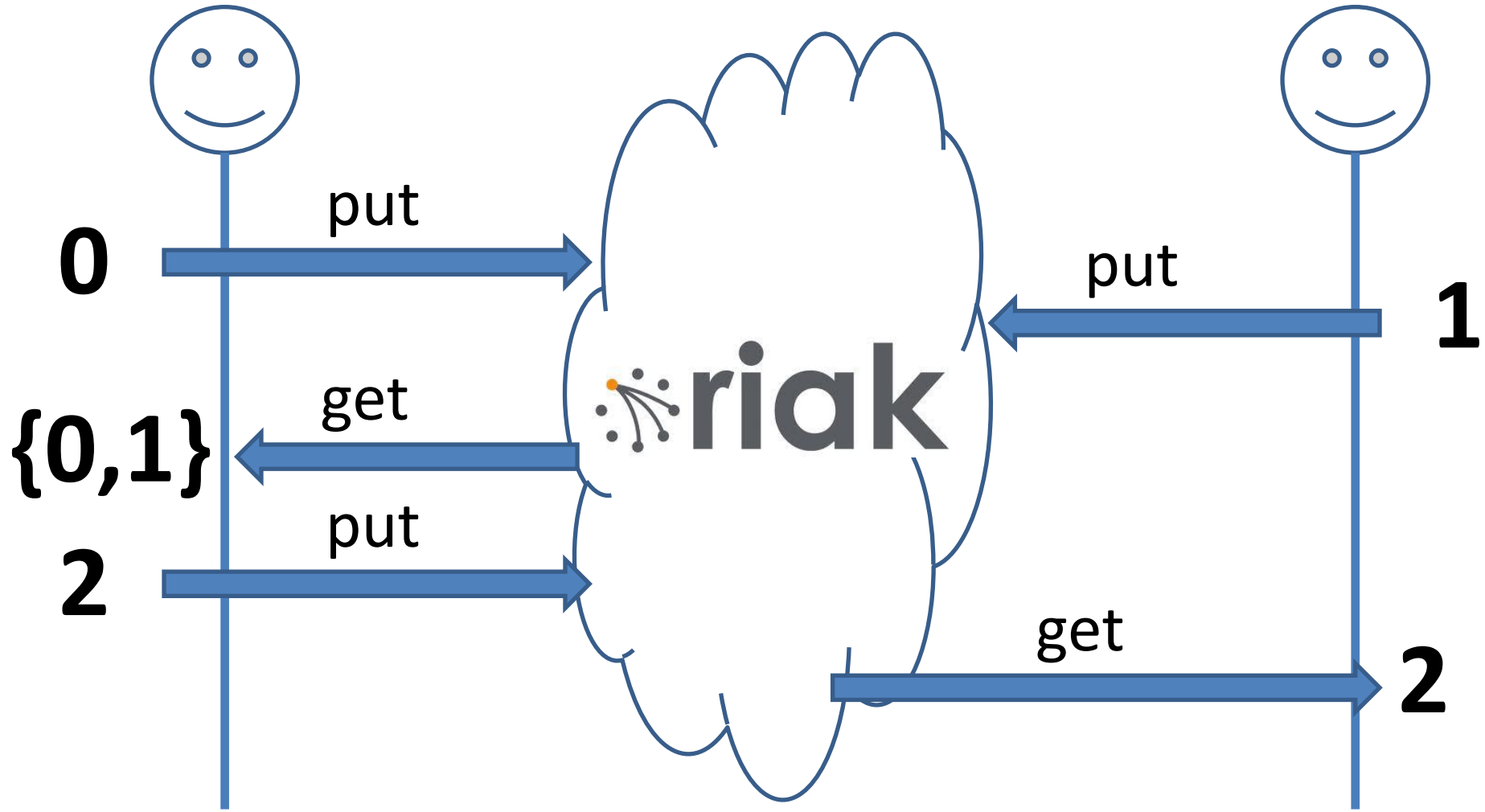
Should we put vector clocks into the model?

NO!!!

Formalise ”You shouldn’t do that!”

- Add to the model:
  - Client’s last view of the value (result of get)
- Add a *precondition:*
  - Every put must update the client’s view (if present)
- QuickCheck generates tests respecting the precondition

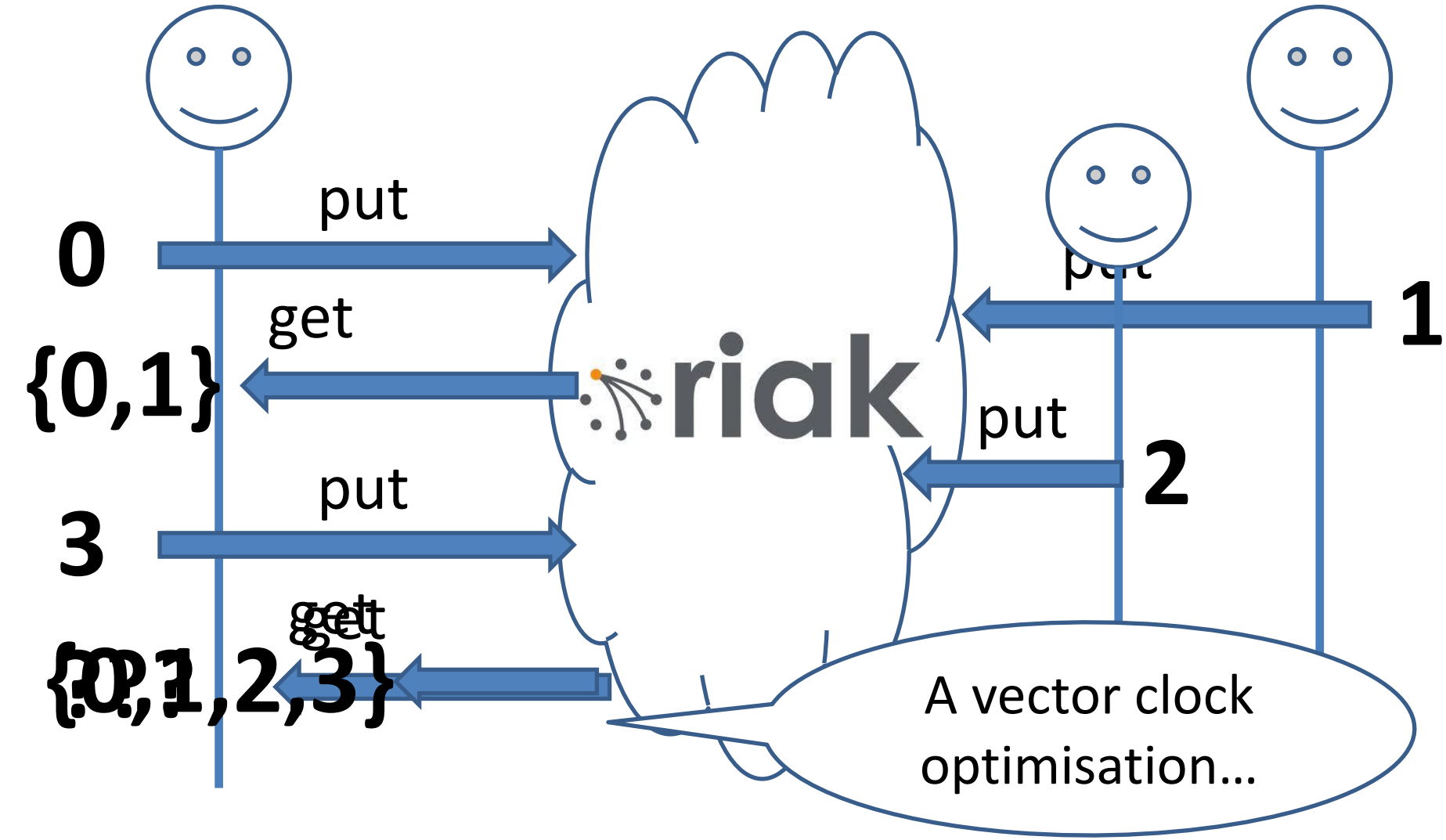
Conflicts



Modelling Conflicts

- The state is a *list* (actually bag) of values
- The client's view is a list of values
- put replaces those values in the state

Example


put
0
get
{0,1}
riak
put
1
put
2
put
3
get
{0,1,2,3}
A vector clock
optimisation…

Modelling Conflicts

- The state is a *list* (actually bag) of values

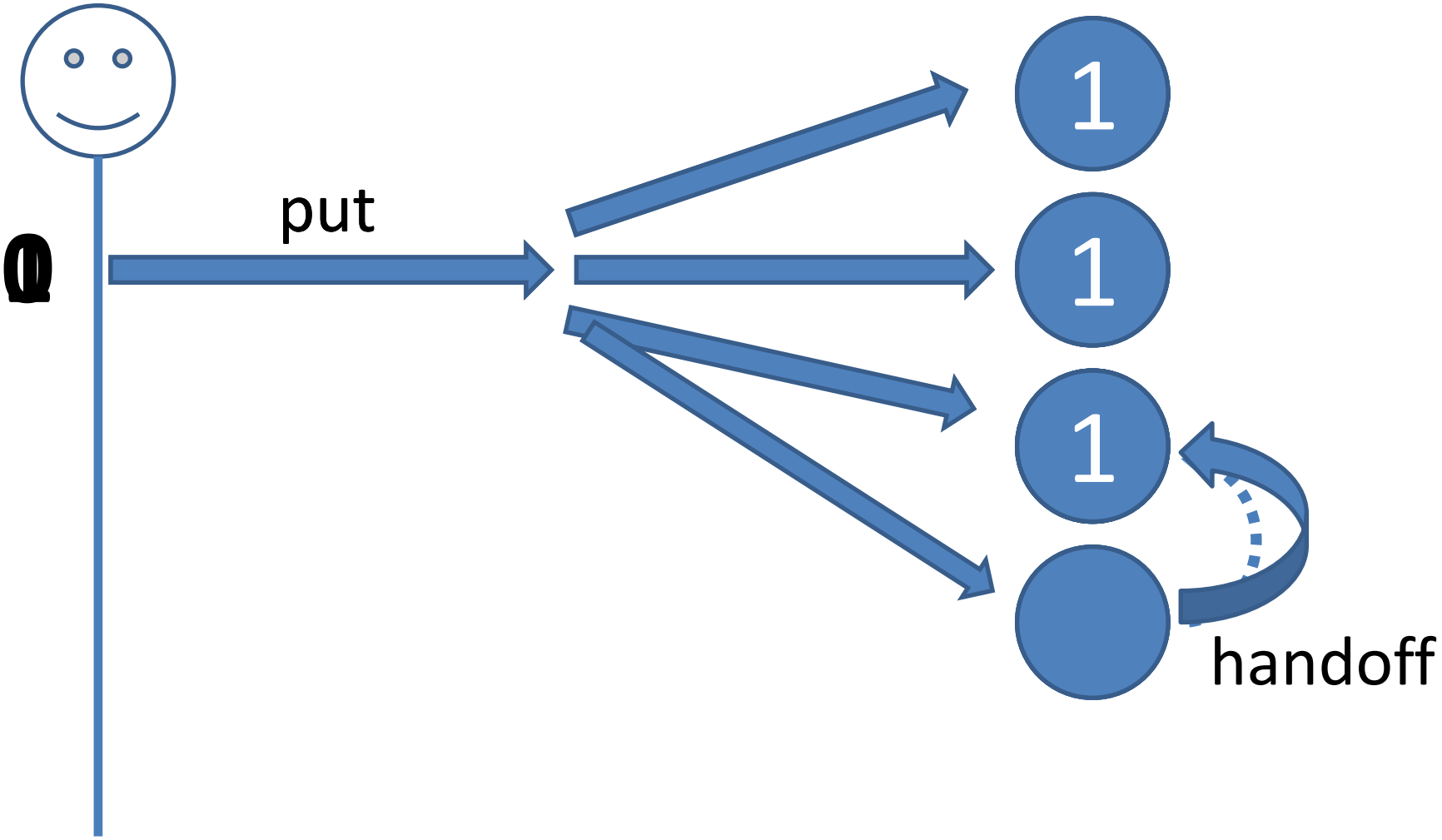
  ”fresh” or ”stale”
- The client’s view is a list of values

- put replaces those values in the state

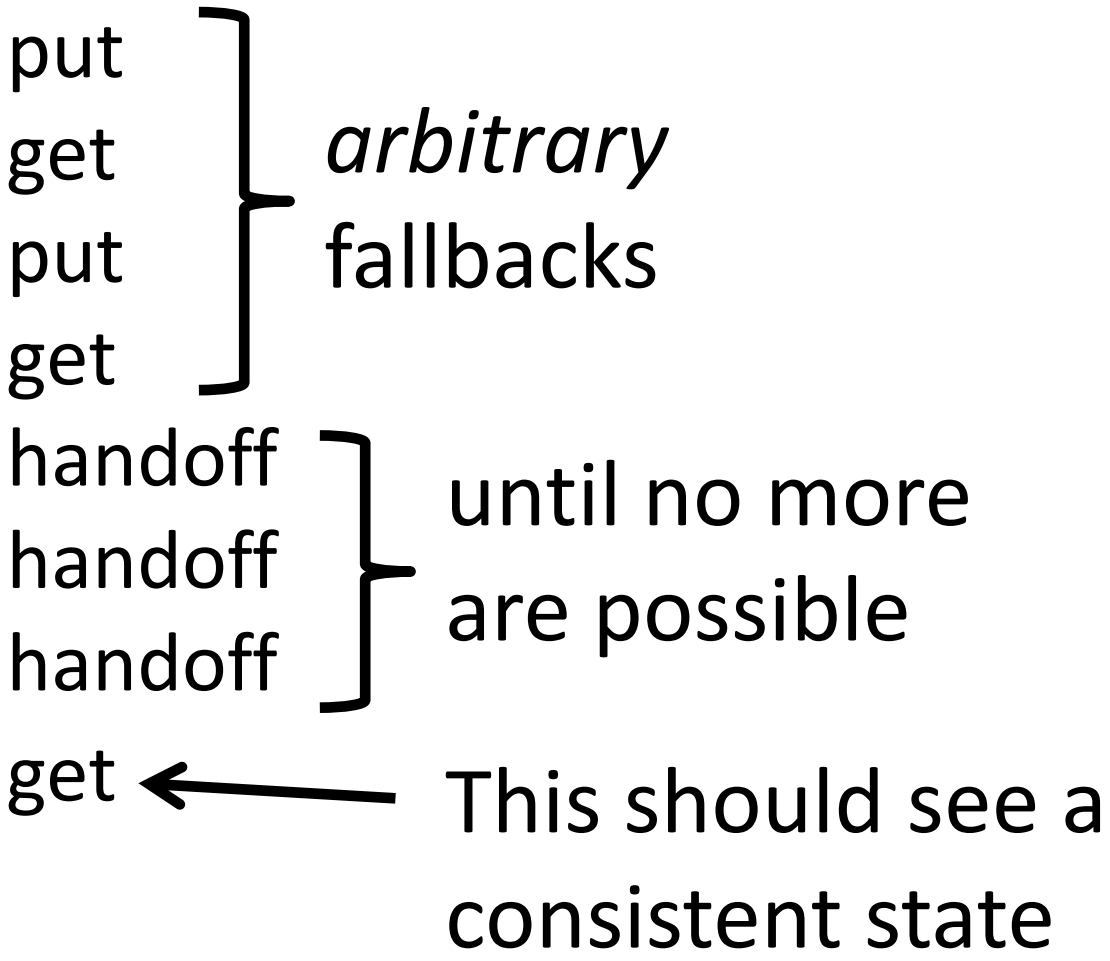
  the state if the client’s view was fresh

  adds a conflict if the client’s view was stale

Redundancy and Fault Tolerance



Testing Eventual Consistency

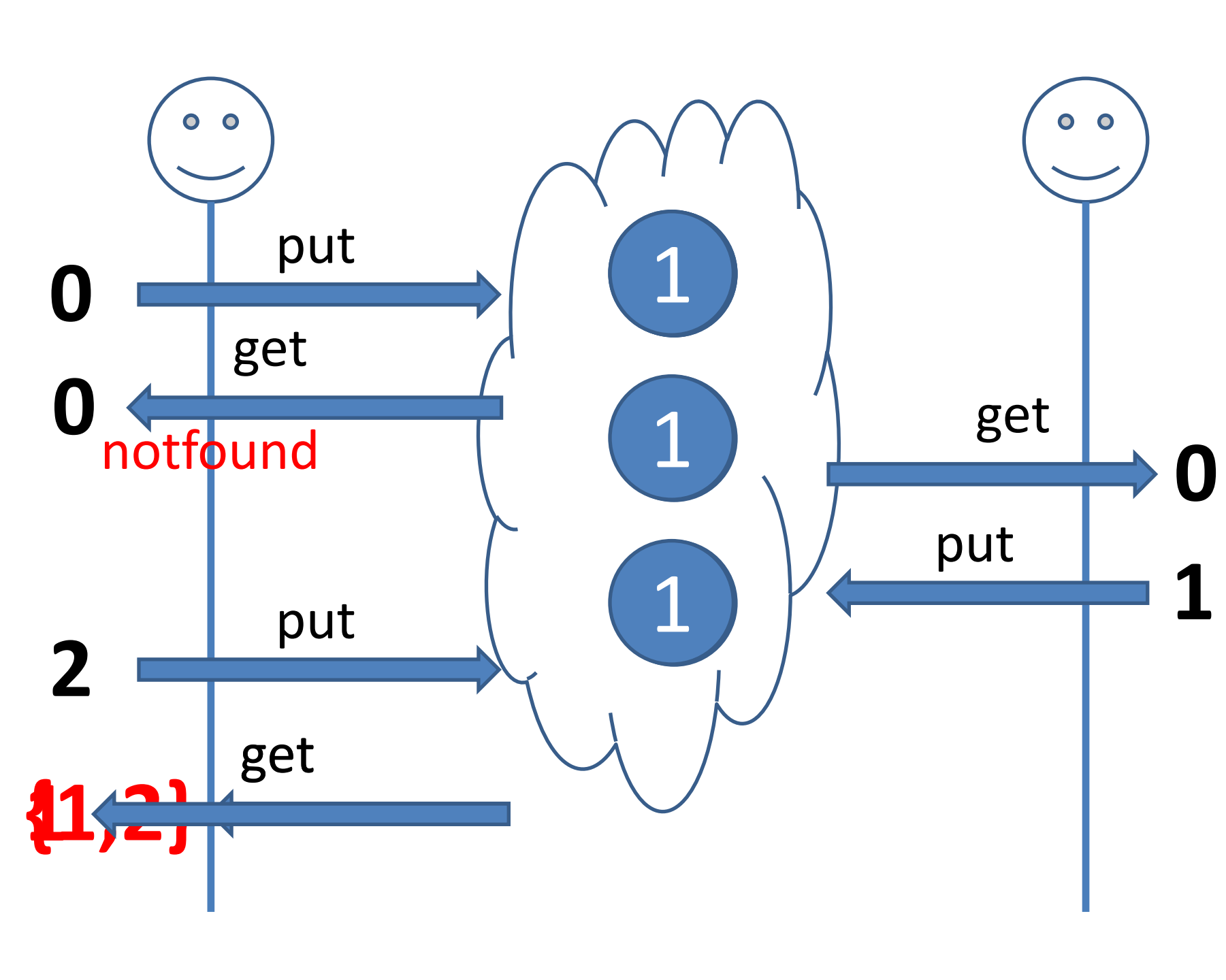


Should we model the *locations* of values?

NO!!!

Modelling Eventual Consistency

- A value ***may*** appear in the final get, if it was ever put
- A value ***must*** appear in the final get, if it was put, and never replaced


put
0
get
0
notfound
1
1
1
get
0
put
1
put
2
get
{1,2}

SHOCK!!

HORROR!

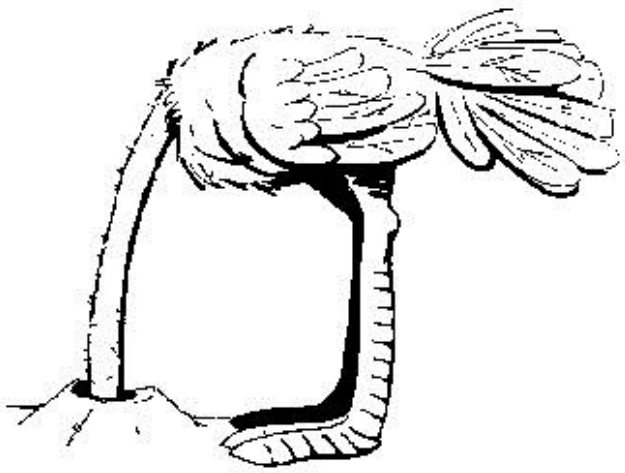
Riak is not eventually consistent!

What does it mean?

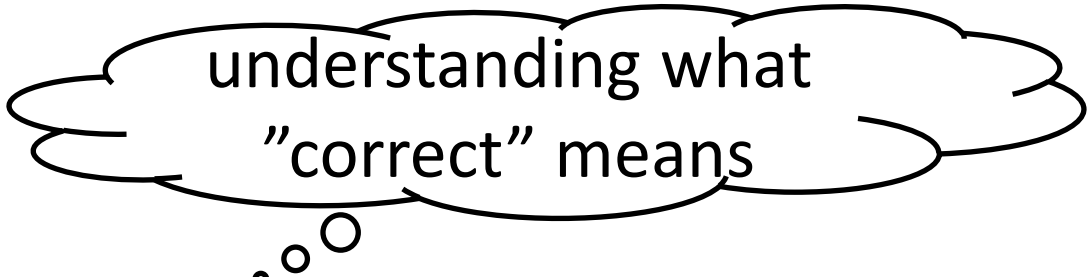
- QuickCheck found…
  - An extreme failure scenario
    - Two quite different failures in a short interval
  - Leading to loss of data
  - A failure of eventual consistency

- So does this mean

  Riak is buggy?



QuickCheck…



- …let us gradually develop a specification, validated against the implementation
- …revealed potentially serious bugs
- …guided the development of alternative solutions that fix the problems

GOOD NEWS!!

Riak *IS* eventually consistent!

— *(at least partially) thanks to QuickCheck* ☺