Riak in 33 seconds

Distributed datastore

Krzysztof Goj Erlang Solutions Ltd
Michał Zajda Erlang Solutions Ltd

What is Riak?

- non-relational database
- distributed
- fault-tolerant
- scalable

Why?

all nodes equal

ec2
linear scalability
chef - automation

Why?

easily accessible

Why?

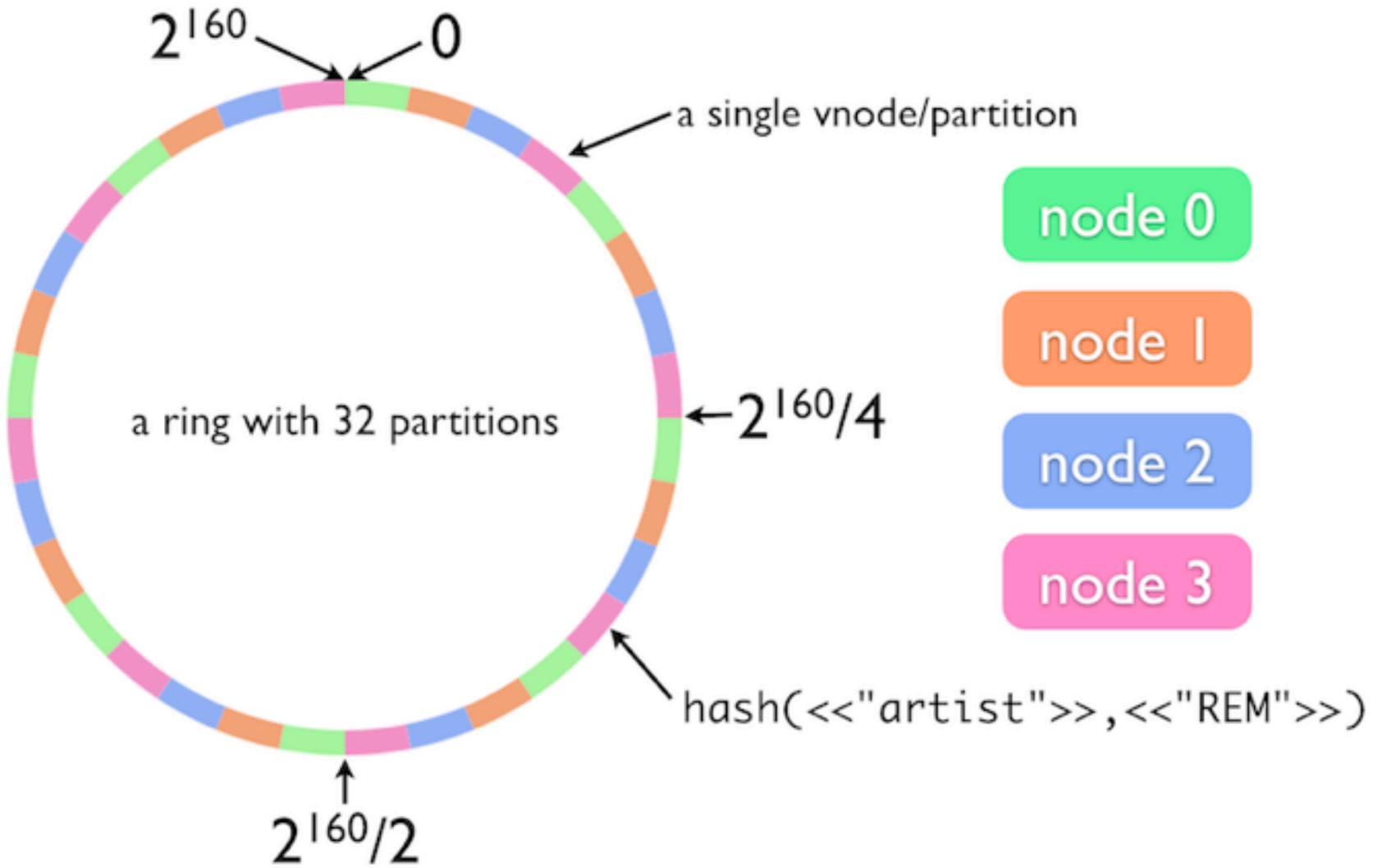
self healing

handful of words

- document / object
- {bucket, key} → value
- link
- vclock

http://isuckatanimation.com/blog/wp-content/uploads/2009/08/amazons3_bucket.jpg

Dynamo model



ring

vnode

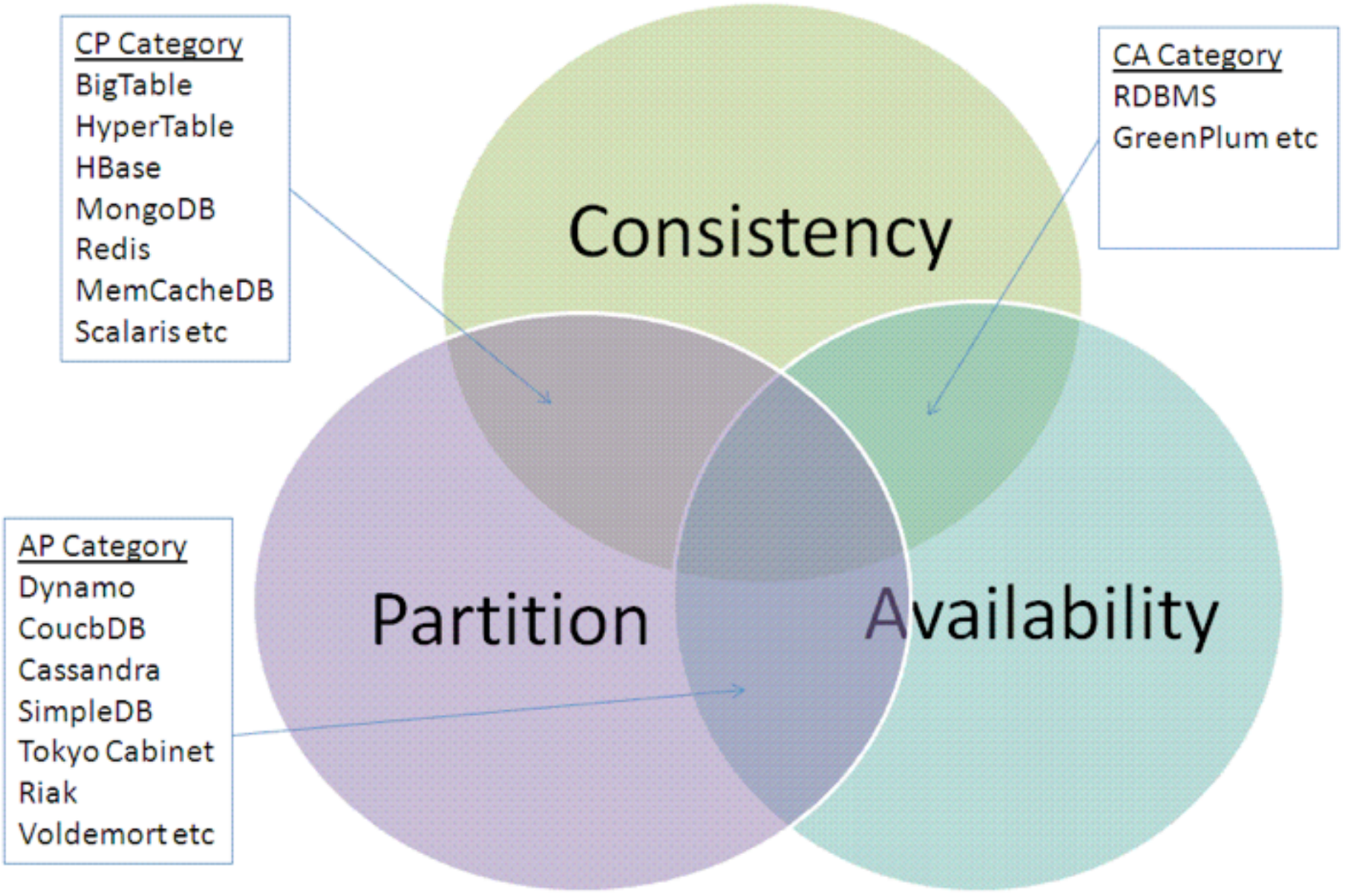
node

consistent hashing

gossip protocol

Diagram from Basho Wiki: https://wiki.basho.com/display/RIAK/An+Introduction+to+Riak

CAP Theorem



Eric Brewer

Every distributed system

Pick Two

http://2.bp.blogspot.com/_S1OQqsgO8Vs/S-NwsyqJwPI/AAAAAAAACfU/Lo44suCk_uw/s1600/UPick2-NoSQL.GIF

even more letters...

N - replicas
R - read ACK
W - write ACK

Consistency vs Throughput

Consistency vs Availability

Storage space vs Availability

...

http://www.benchmark.pl/uploads/image/datacenter.gif

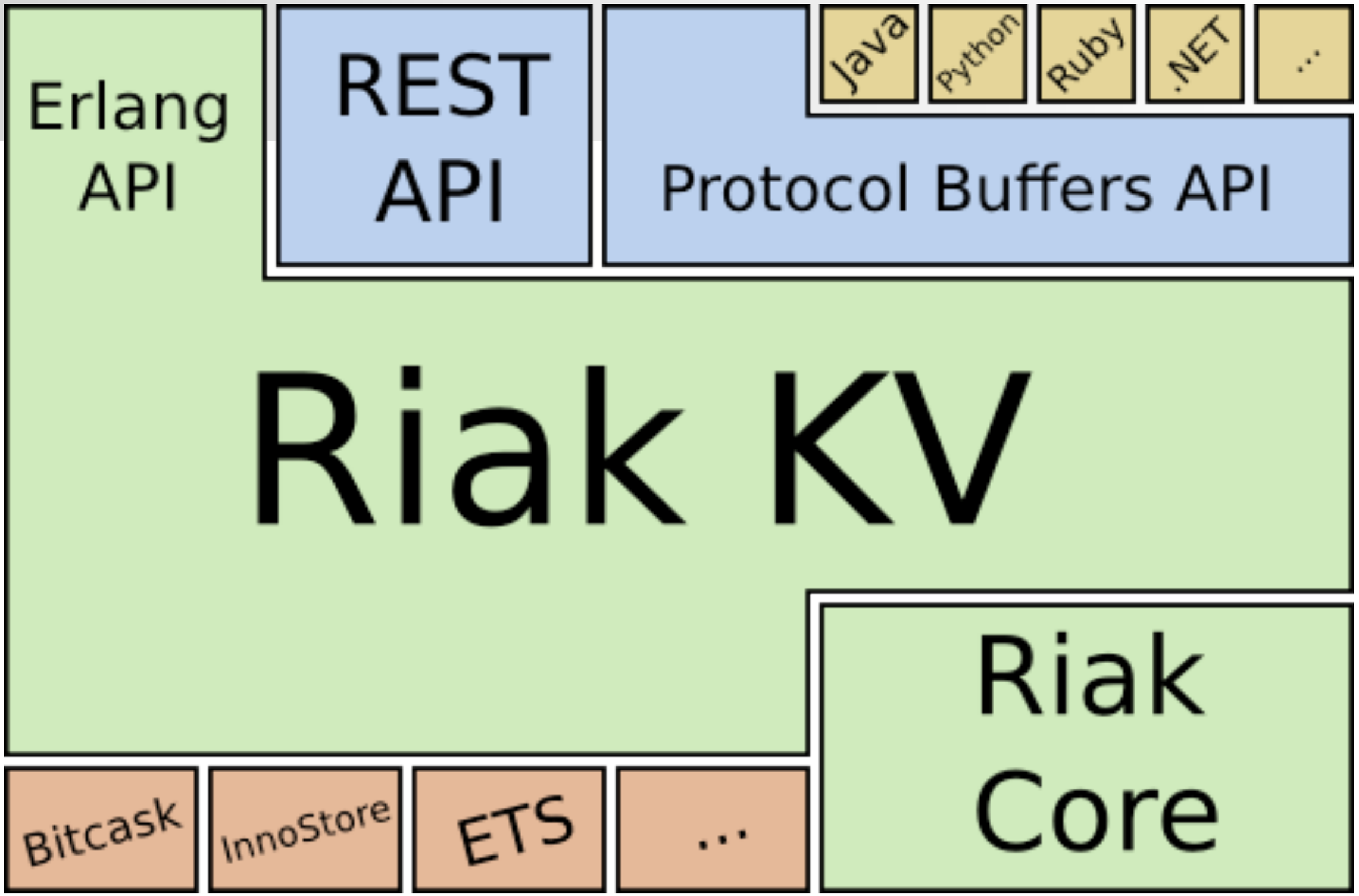
CAP in a real life

- N = 3, R = 3, W = 1
- no node can be down for reads
- two nodes can be down for write
- we can change R and W per request

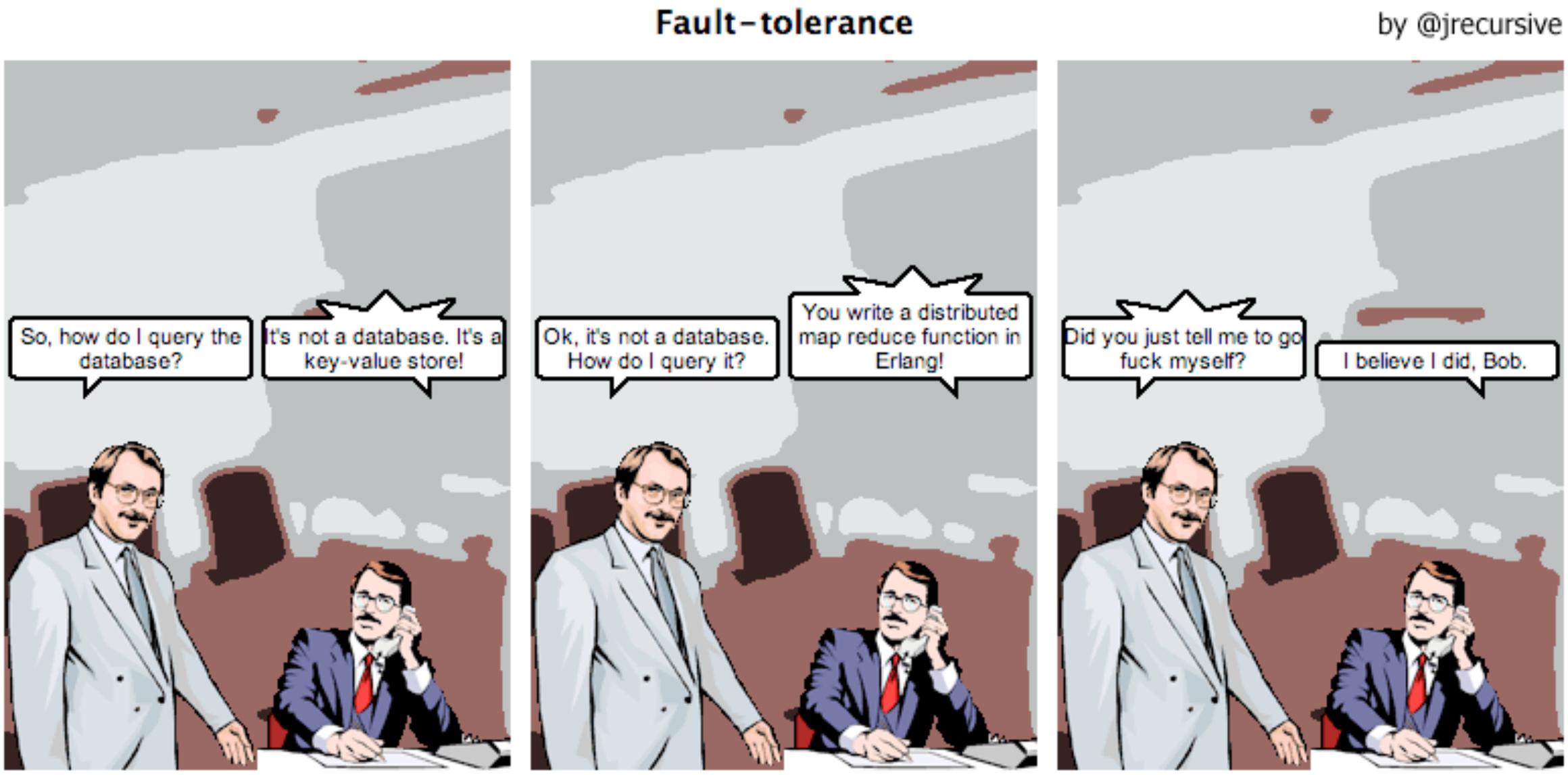
http://stylio.pl/zdjecia/499/499089_l/zjem_ci_szejka/full+cap+domo+..jpg

Erlang

Architecture



Obligatory NoSQL Slide



map/reduce

- in Erlang & JavaScript
- ‘Jobs’ build of Map, Reduce, Link ‘Phases’
- Parallel N Maps vs. 2 Reduces (WiP)

riak search

http://csshook.com/cssresources/wp-content/uploads/2009/03/old-world-map.jpg

http://www.iosss.com/shopping/images/t5033.jpg

Pros

- scalable
- fault tolerant
- easy to manage on the cloud
- in Erlang ;-)

Cons

- You probably don’t need it
- Real-world data happens to be relational
- Implementation still have few bugs

Thank You!