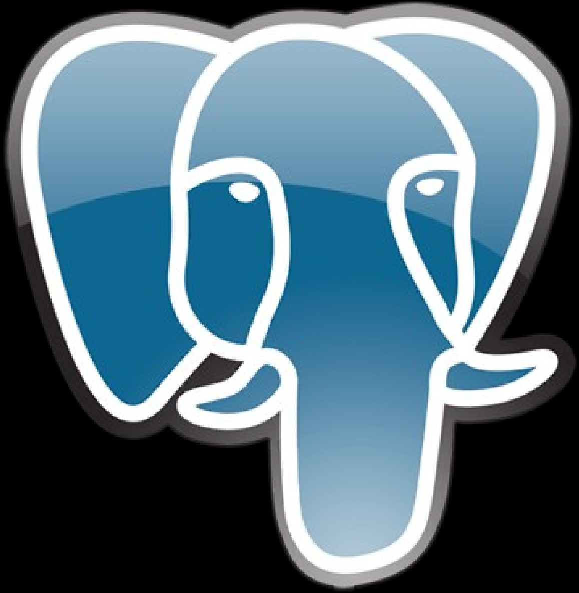
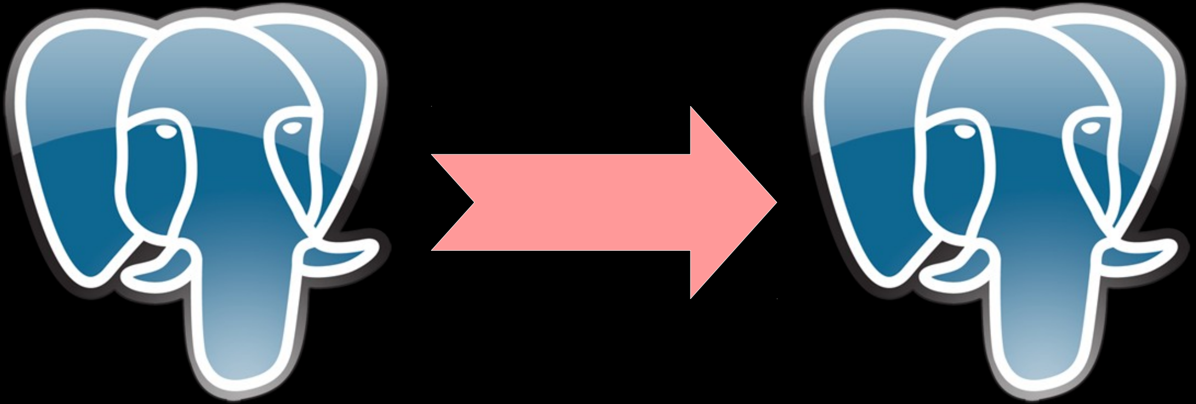
Bots Not Cattle

Josh Berkus
Red Hat
Automacon PDX
September 2016





HA Postgres

third wave of automation?

1. software packages & BASH
2. configuration management
3. ???

third wave of automation?

1. software packages & BASH
2. configuration management
3. container orchestration?

third wave of automation?

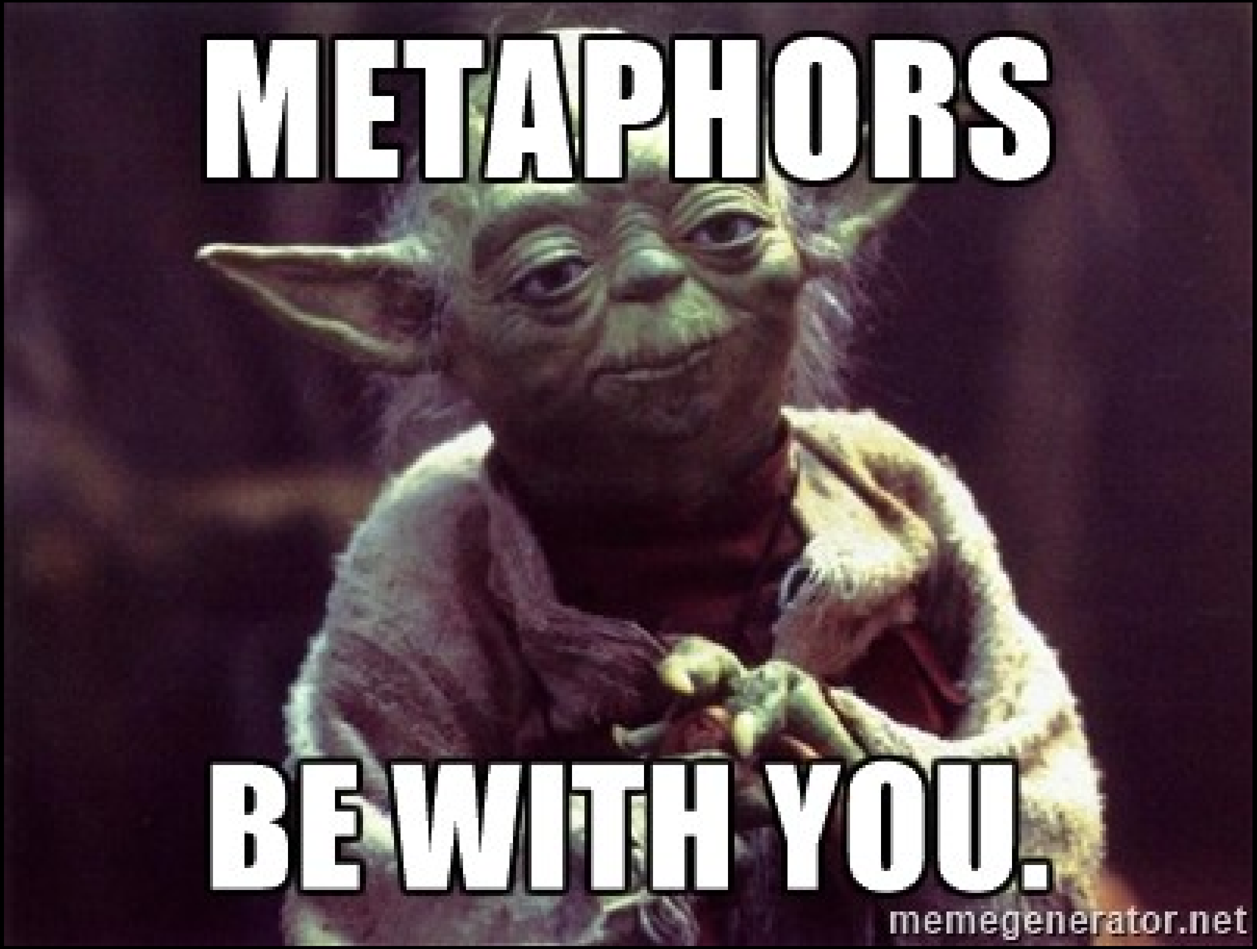
1. software packages
2. a. configuration management

   b. container orchestration

third wave of automation?

1. software packages
2. a. configuration management

   b. container orchestration
3. ???


METAPHORS
BE WITH YOU.
memegenerator.net

Treat Servers Like Cattle, Not Pets

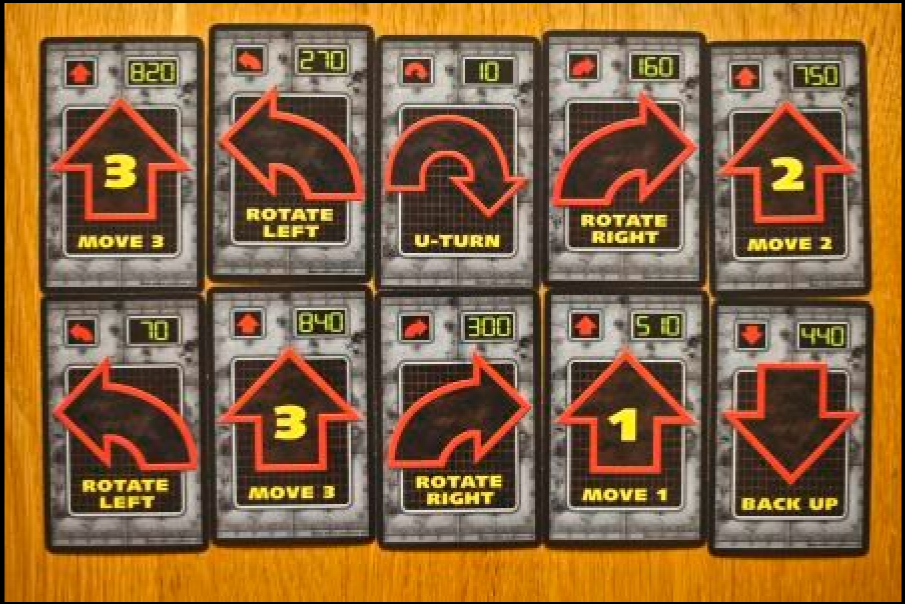
the problem with cattle

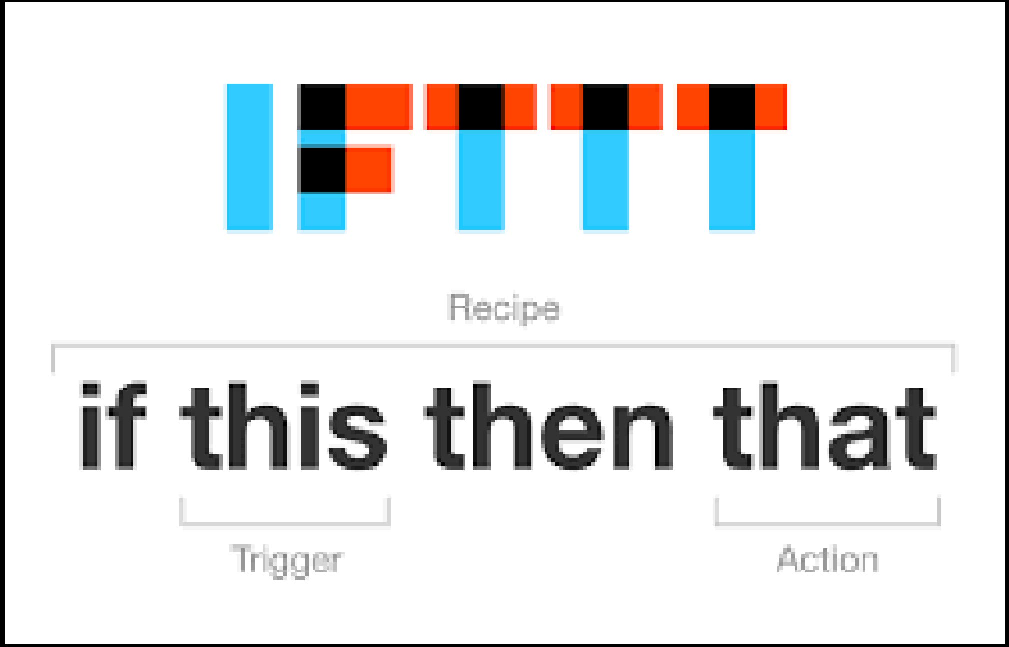
- *dumb*
- move only in one direction
- require (smart) central management

central management limits

- infrastructure scale
- people scale (dev/ops split)
- communications lag
- application event response

ROBO RALLY
AVALON HILL
A STRATEGY GAME BY RICHARD GARFIELD
A FRENZIED RACE FILLED WITH COMPUTER-DRIVEN CHAOS!
6
AGE: 12+
TIME: 30+ min.
PLAYERS: 2–8


820
3
MOVE 3
270
ROTATE LEFT
10
U-TURN
160
ROTATE RIGHT
750
2
MOVE 2
70
ROTATE LEFT
840
3
MOVE 3
300
ROTATE RIGHT
510
1
MOVE 1
440
BACK UP



IFTTT

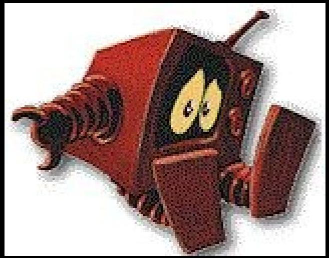
Recipe

if this then that

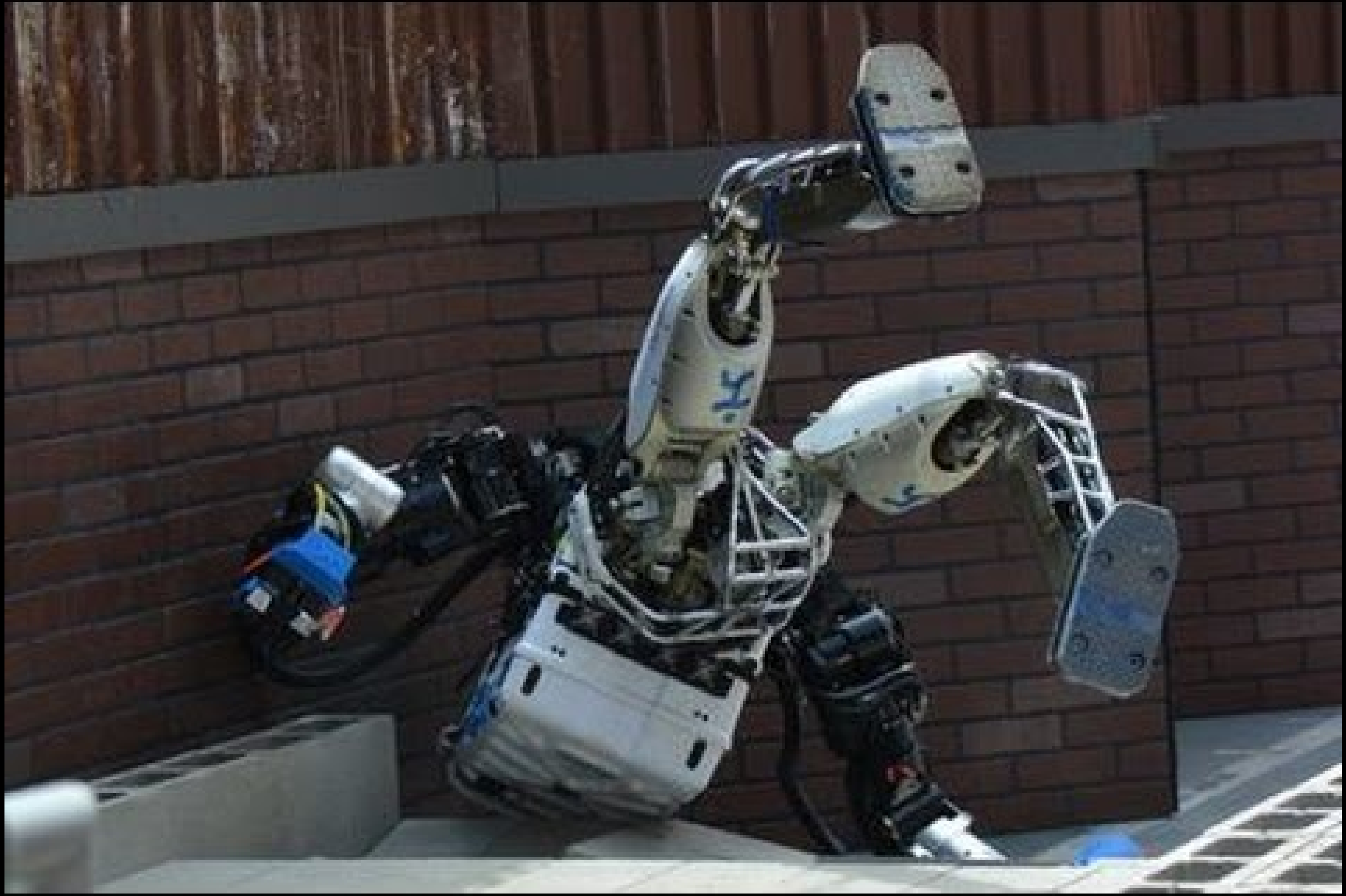
Trigger

Action

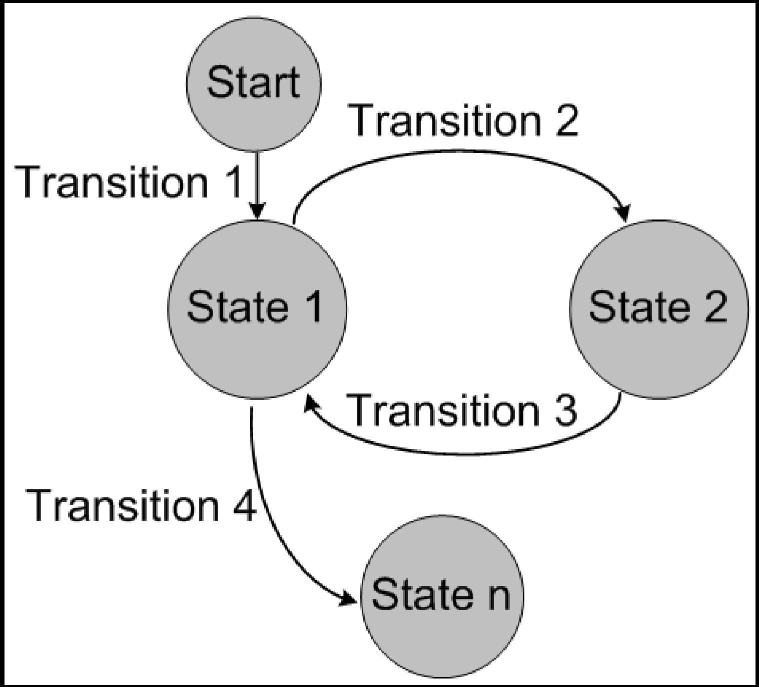
intelligent "bots"



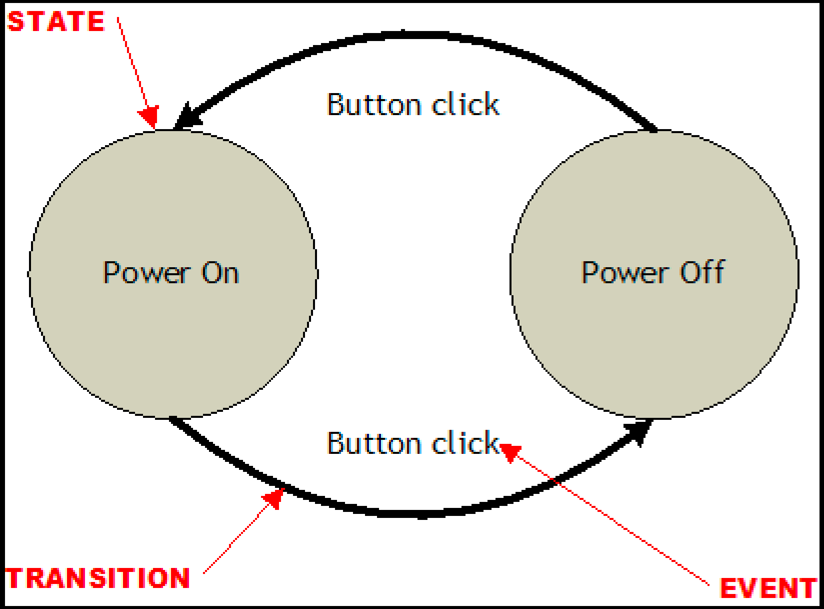
- capable of reacting to deployment, configuration, and availability events
- programmed to "do the right thing" in each case



state machines



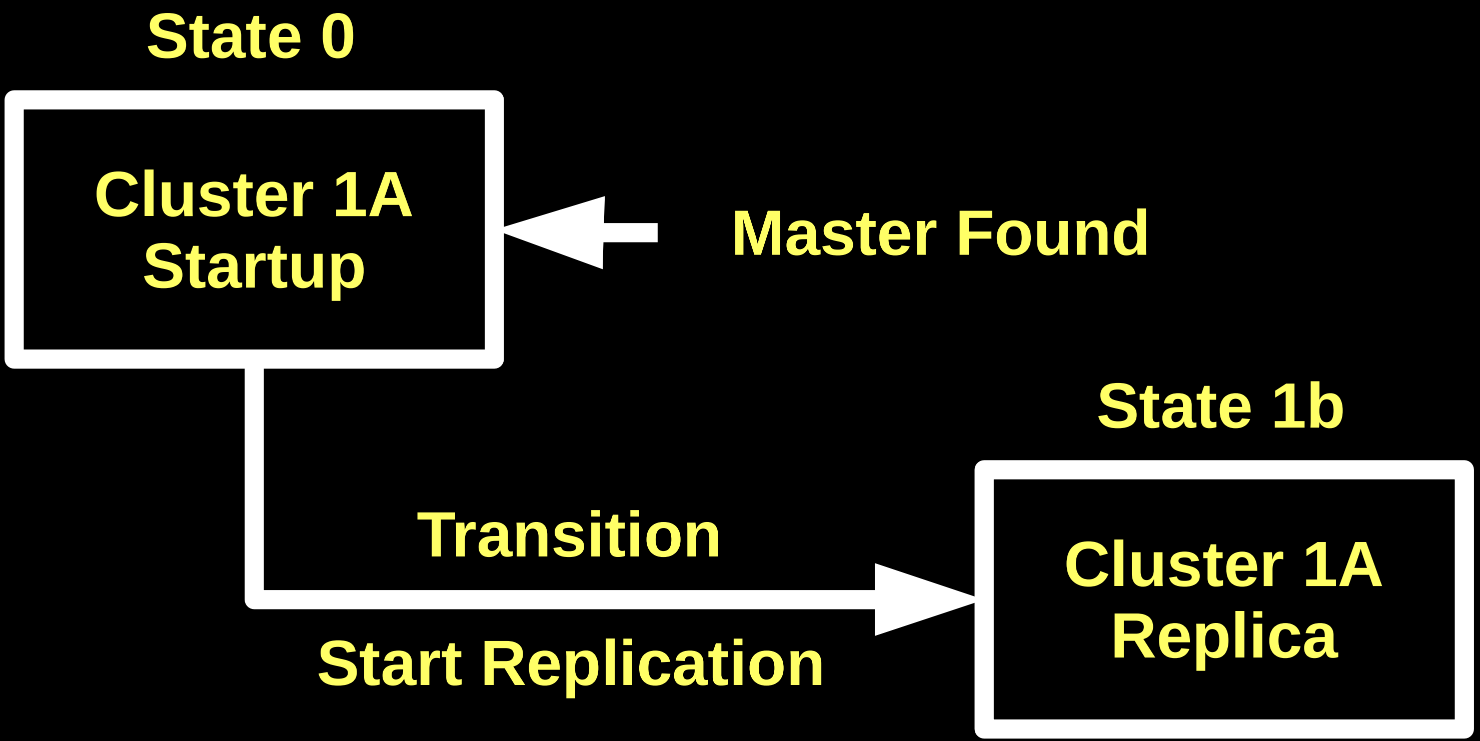
state machines



state machines

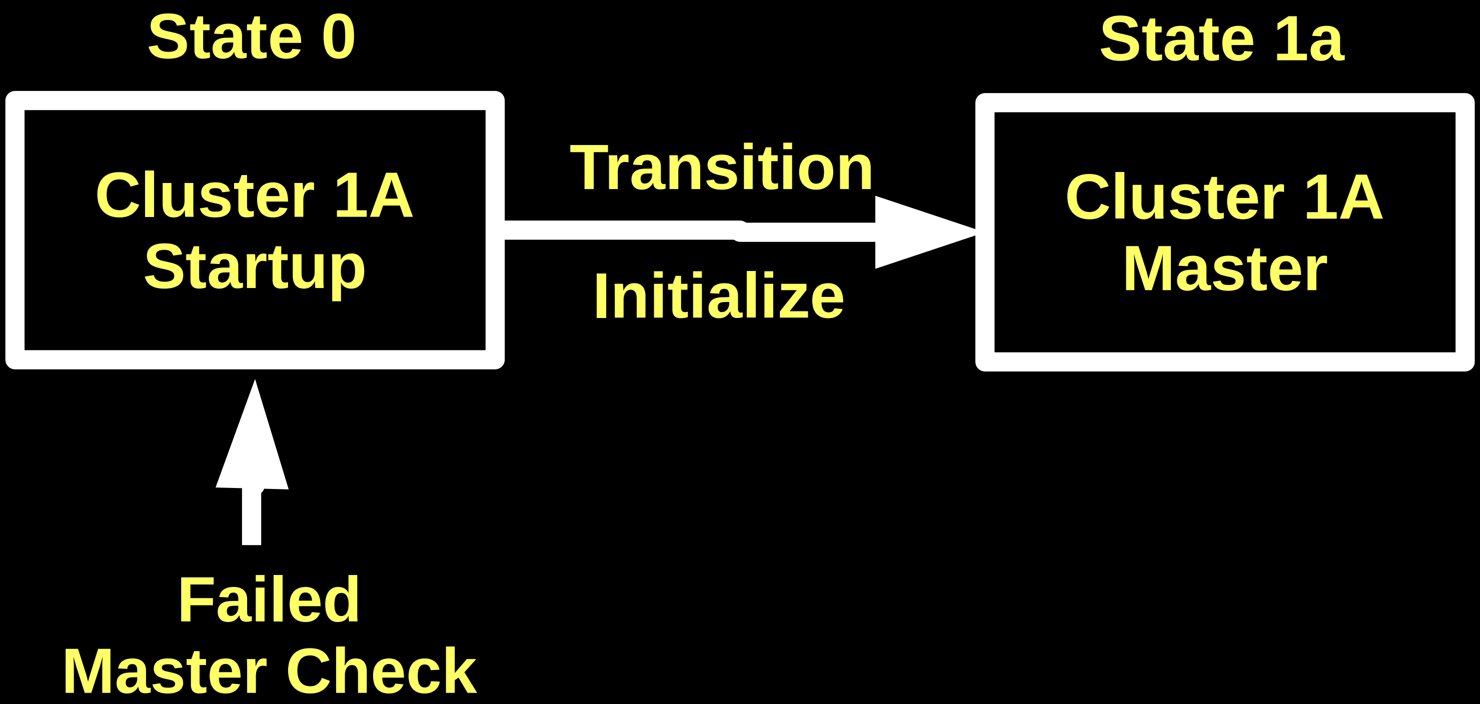
- simpler, testable logic
- avoid unhandled events
- *much* simpler scheduling

patroni: postgres automation



State 0

Cluster 1A
Startup



State 0

Cluster 1A Startup

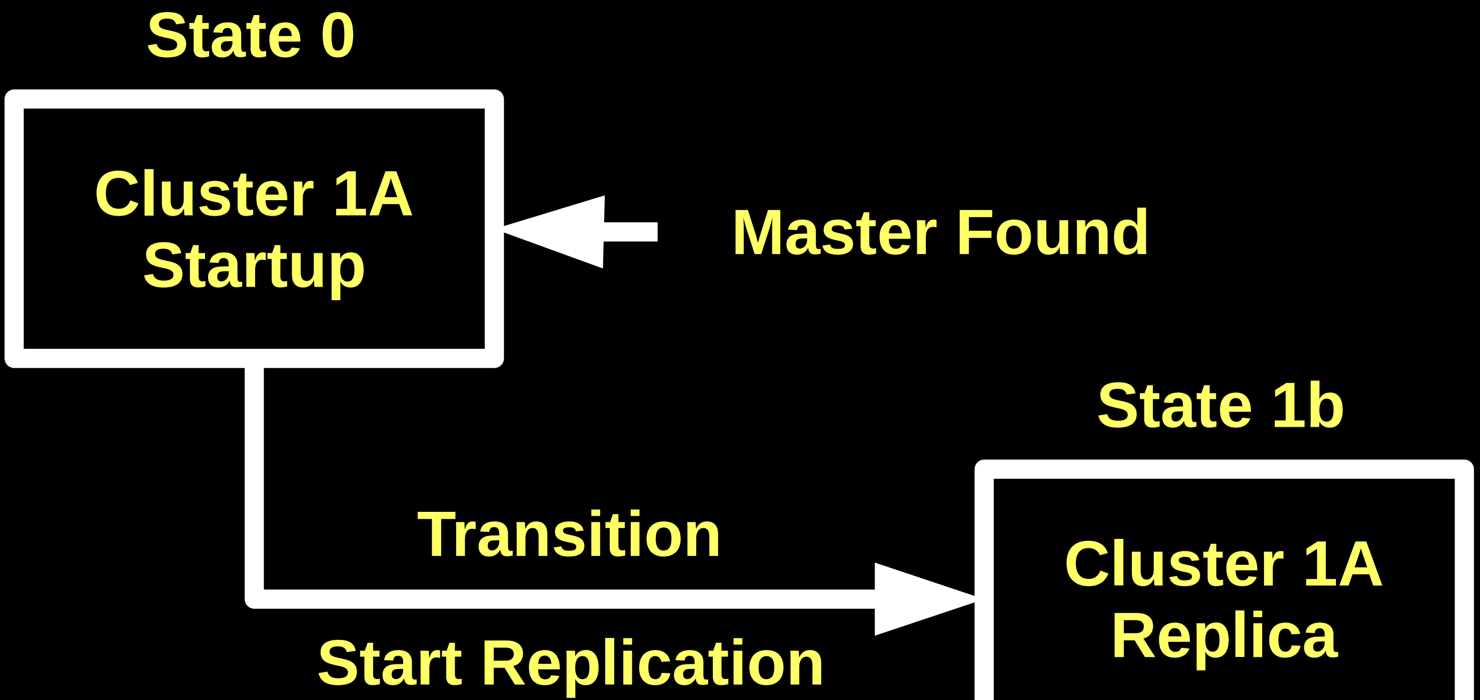
Transition

Initialize

State 1a

Cluster 1A Master

Failed Master Check



State 0

Cluster 1A
Startup

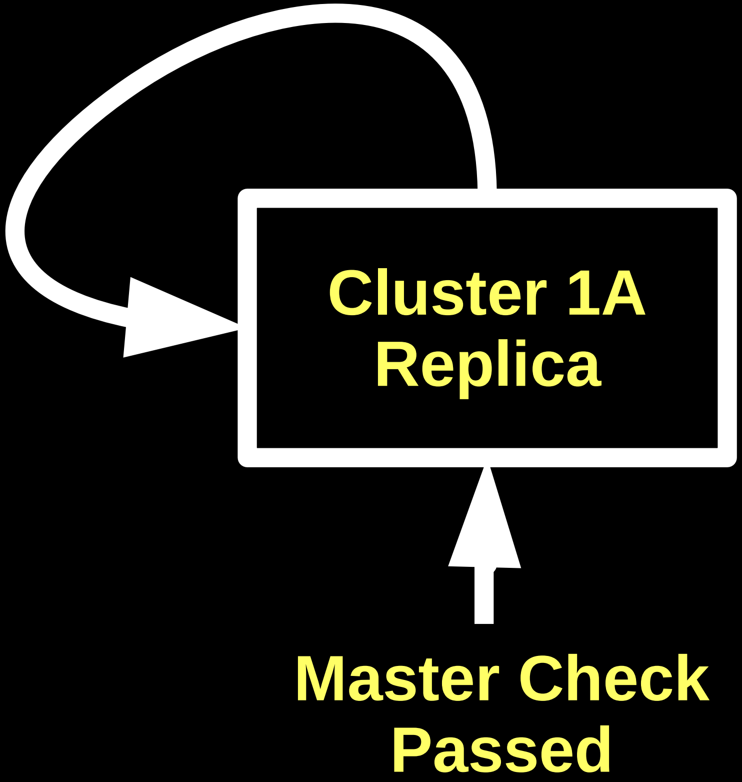
Master Found

State 1b

Transition

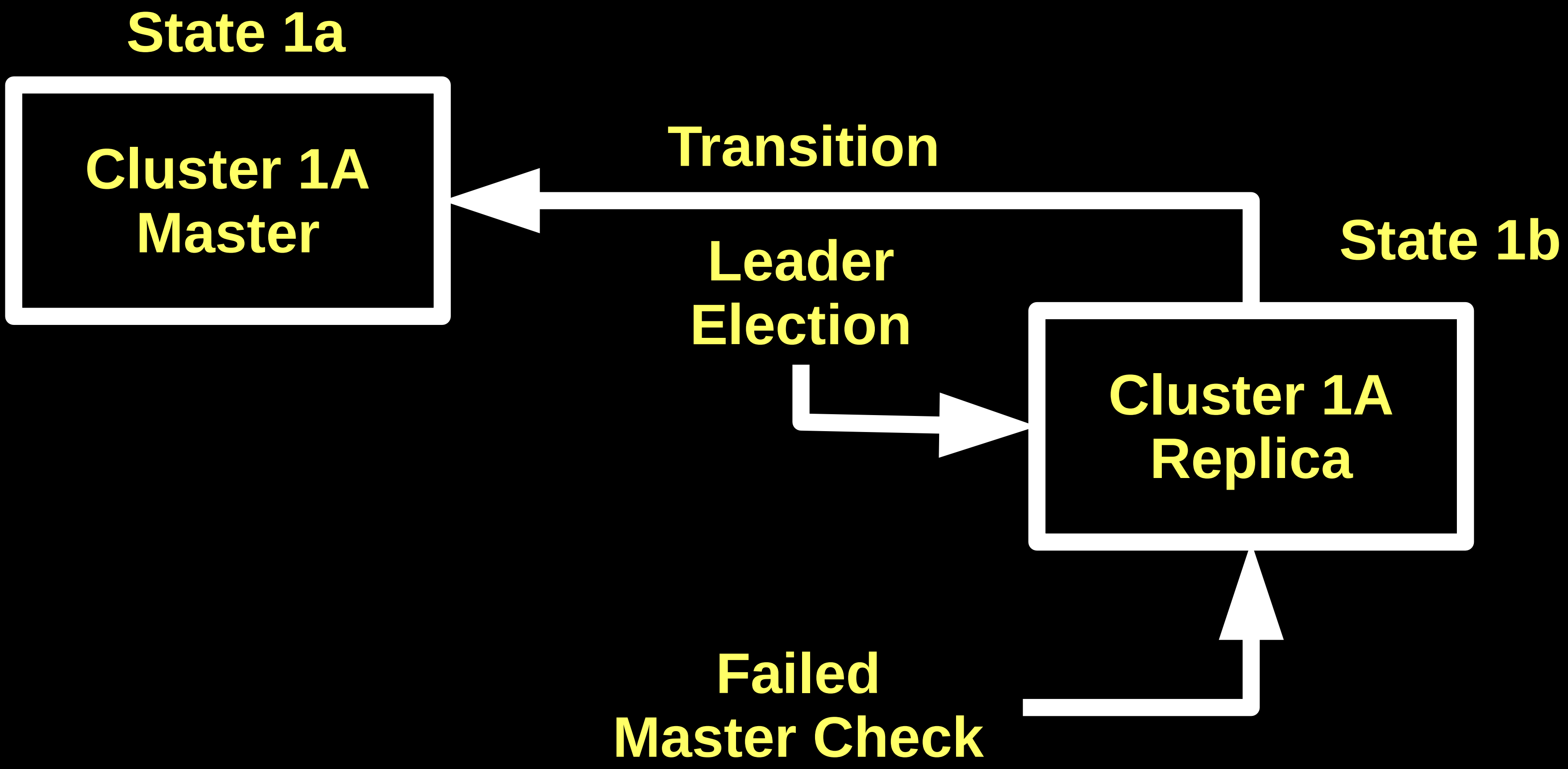
Cluster 1A
Replica

Start Replication



Cluster 1A Replica

Master Check Passed



State 1a

Cluster 1A
Master

Transition

Leader
Election

State 1b

Cluster 1A
Replica

Failed
Master Check

bots

automate the application,
not the system

new bot rules

1. bots initialize themselves
2. bots know their own state
3. bots share their state
4. bots change their state in response to events

new bot rules

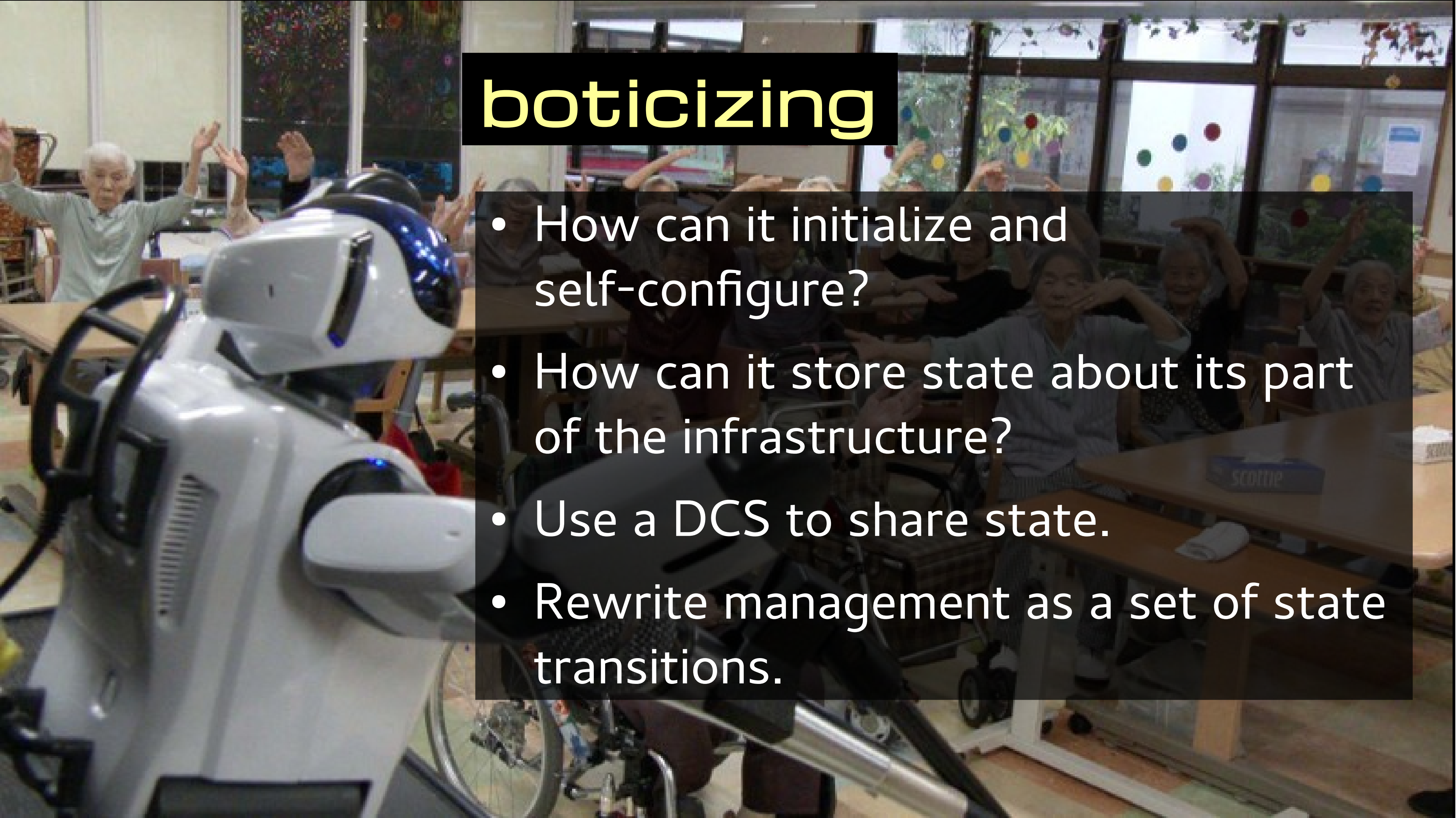
1. self-configuration
2. local state store
3. distributed configuration
4. event-driven response loop

new bot rules

1. sophisticated init code
2. local daemon which maintains state
3. write state to etcd
4. watch etcd for changes

bot-ish platforms

- Postgres: patroni, manatee, flynn.io
- Chef Habitat
- Kubernetes custom schedulers
- Mesos Frameworks



boticizing

- How can it initialize and self-configure?
- How can it store state about its part of the infrastructure?
- Use a DCS to share state.
- Rewrite management as a set of state transitions.

boticizing a web app: init

1. initialize with DCS connection & node name
2. query DCS for DB connection & CDN
3. pull new code from git
4. launch
5. update DCS with state

bot problems

- lagginess:
  event watch, gossip
- race conditions: DCS locking
- resource usage:
  resource events?
- frameworks for writing state machines?

devs need to automate

devs need to automate

¯_(ツ)_/¯

we're all DevOps now

let's build some bots!

Josh Berkus
Red Hat
Automacon PDX
Sept. 2016

let's build some bots!

Contact

- jberkus@redhat.com
- @fuzzychef
- www.projectatomic.io

Events

- Cloud Native PDX Meetup
- KubeCon Nov. 7, Seattle

the fine print