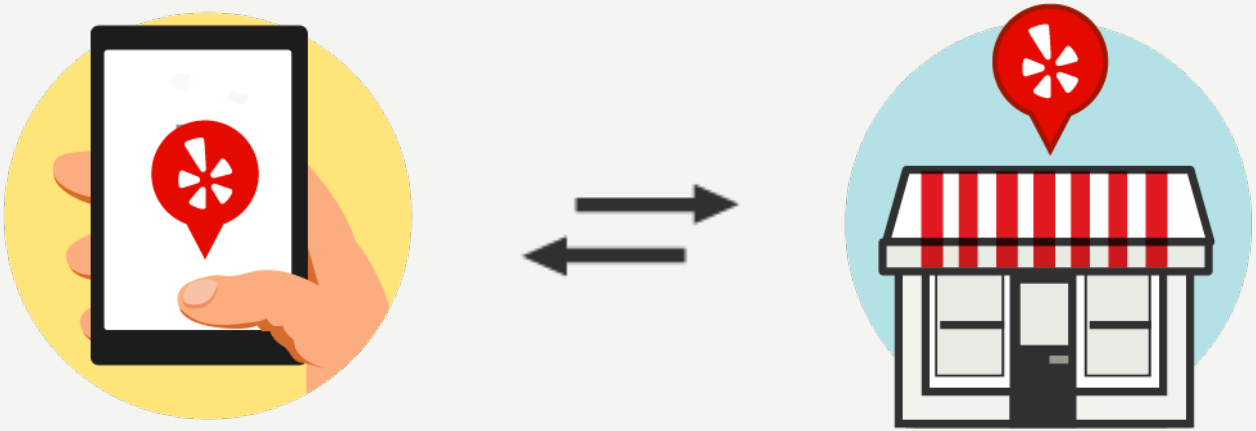
Protect your users with Circuit Breakers

Scott Triglia

Europython 2016

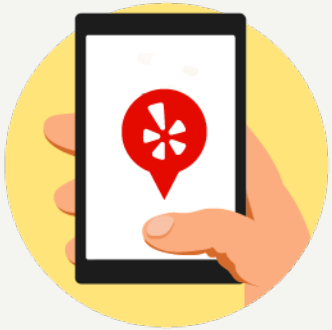
Yelp's Mission:

Connecting people with great local businesses.



Yelp Stats

As of Q1 2016



90M

102M

70%

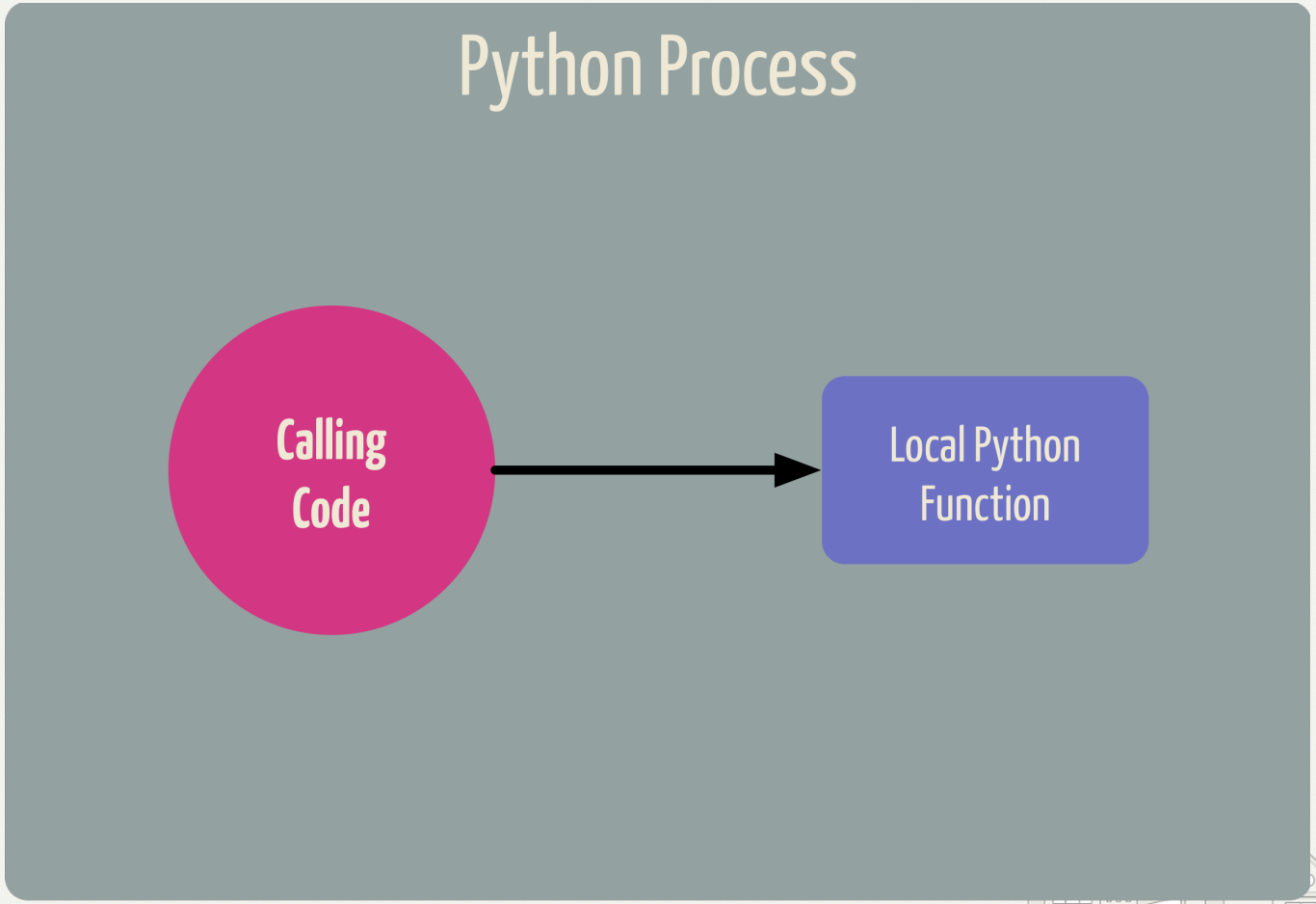
32

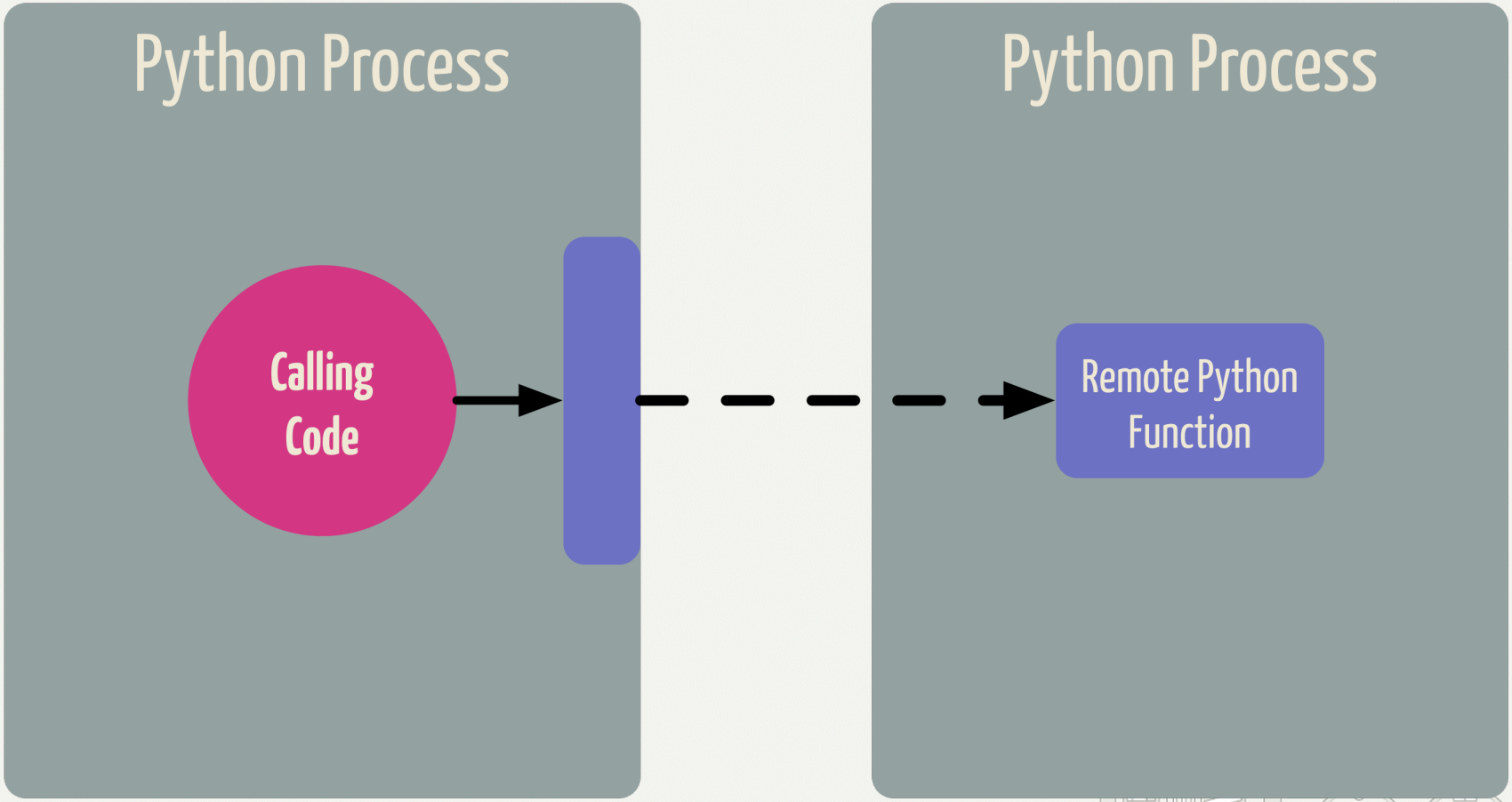
Scott Triglia

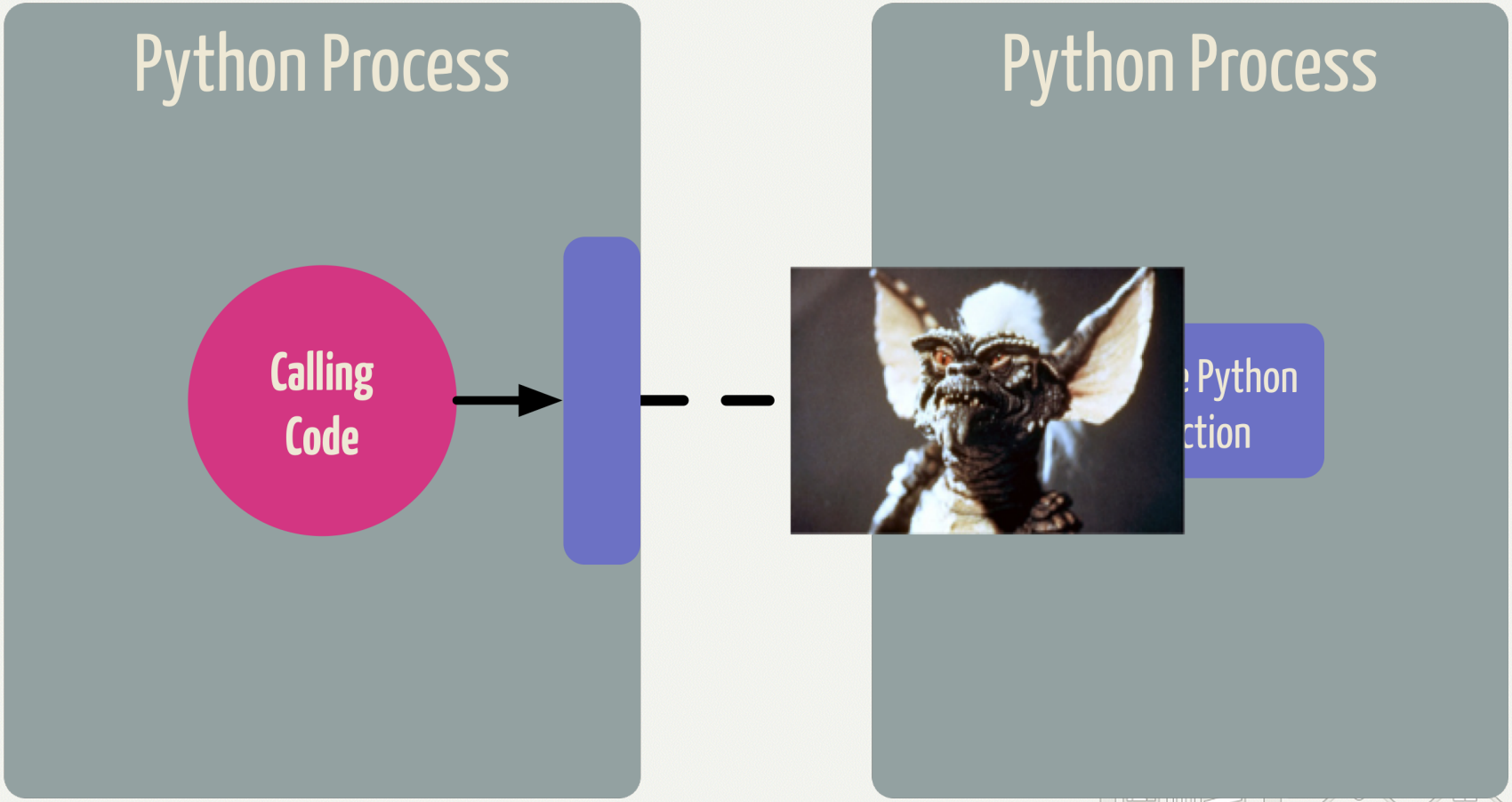
@scott_triglia

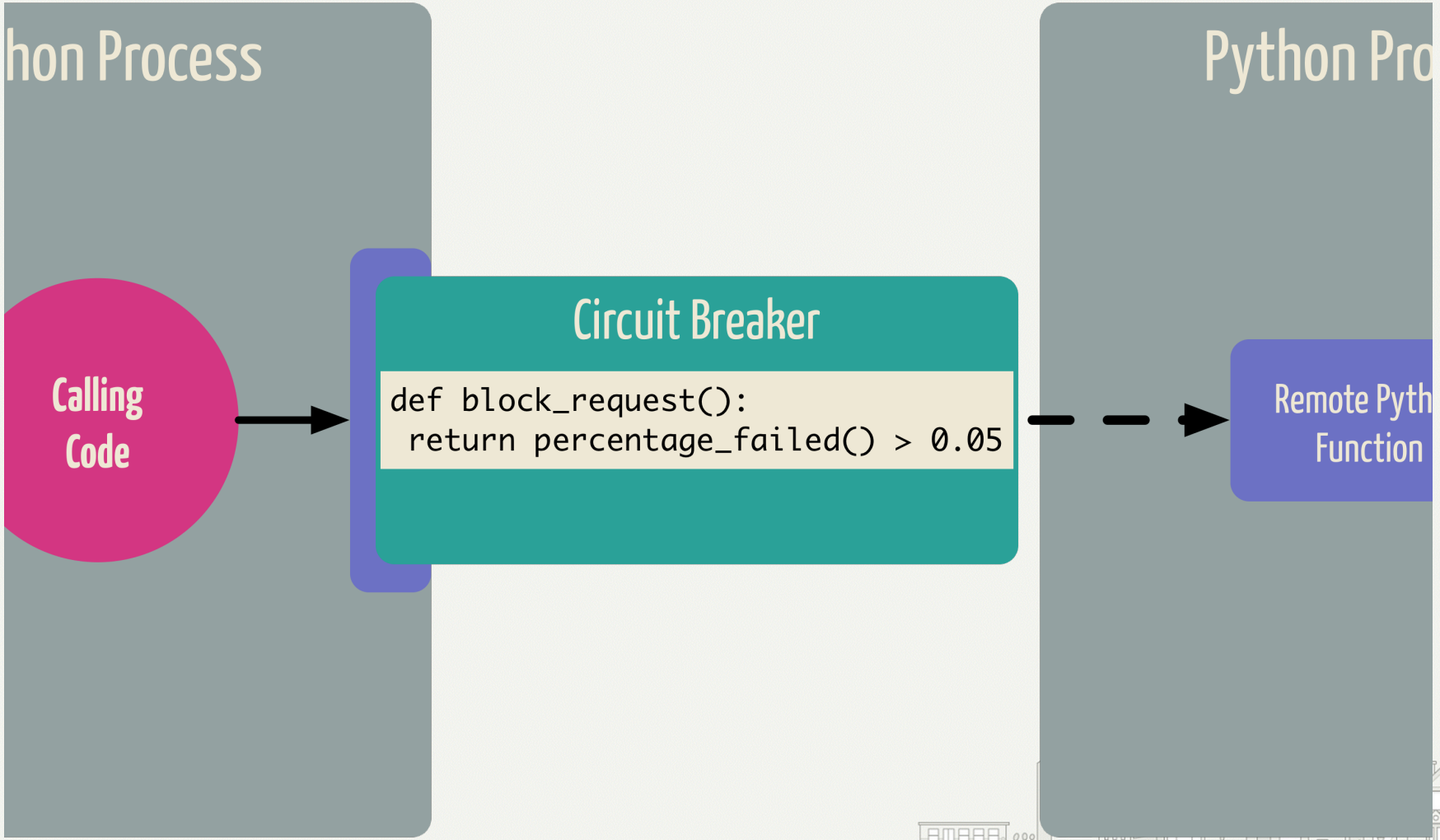
Work with the $$$

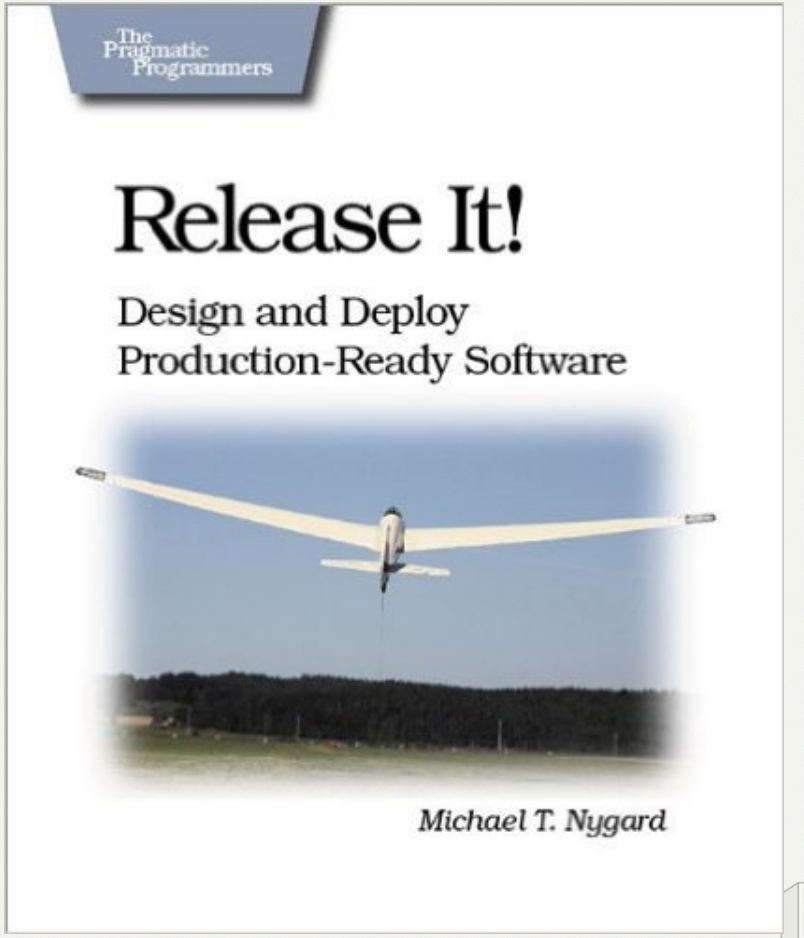
Let’s talk Circuit Breakers


Python Process
Calling Code
Local Python Function


Python Process
Calling Code
Python Process
Remote Python Function


Python Process
Calling Code
Python Process
Python
ction


hon Process
Python Pro
Calling Code
Circuit Breaker
def block_request():
return percentage_failed() > 0.05
Remote Pyth
Function


The Pragmatic Programmers
Release It!
Design and Deploy
Production-Ready Software
Michael T. Nygard

Our goals today:

introduce a basic circuit breaker

Our goals today:

a modular circuit breaker

Our goals today:

test it out on several scenarios


JIM'S
Restaurant
CATERING
OPEN


THRIFT STORE
LOOK-N-BUY TRADING CO.

1


Jim's
RESTAURANT
2420 Mission Street
San Francisco, CA
415-285-6020
Side Orders
EXPRESSO DRINKS
Regular
Grande
Beverages

2

3

4

the fundamental rule:

your systems will fail

what's your response?

1


Jim's
RESTAURANT
2420 Mission Street
San Francisco, CA
415-285-6020
Side Orders
Vegetables 2.05
Avocado 1.65
Onion Ring 3.05
Sweet Potato Fries 3.05
Fresh Fruit 3.55
Sour Cream 0.75
Cottage Cheese 2.55
EXPRESSO DRINKS
Regular
Grande
1.95
2.25
2.75
3.85
Beverages
Hot Chocolate
Coffee

2

1


Jim's
RESTAURANT
2420 Mission Street
San Francisco, CA
415-285-6020
Side Orders
Vegetables 2.05
Avocado 1.65
Onion Ring 3.05
Sweet Potato Fries 3.05
Fresh Fruit 3.55
Sour Cream 0.75
Cottage Cheese 2.55
EXPRESSO DRINKS
Regular Grande
1.95 2.25
2.75 3.85
Beverages

2

3

1

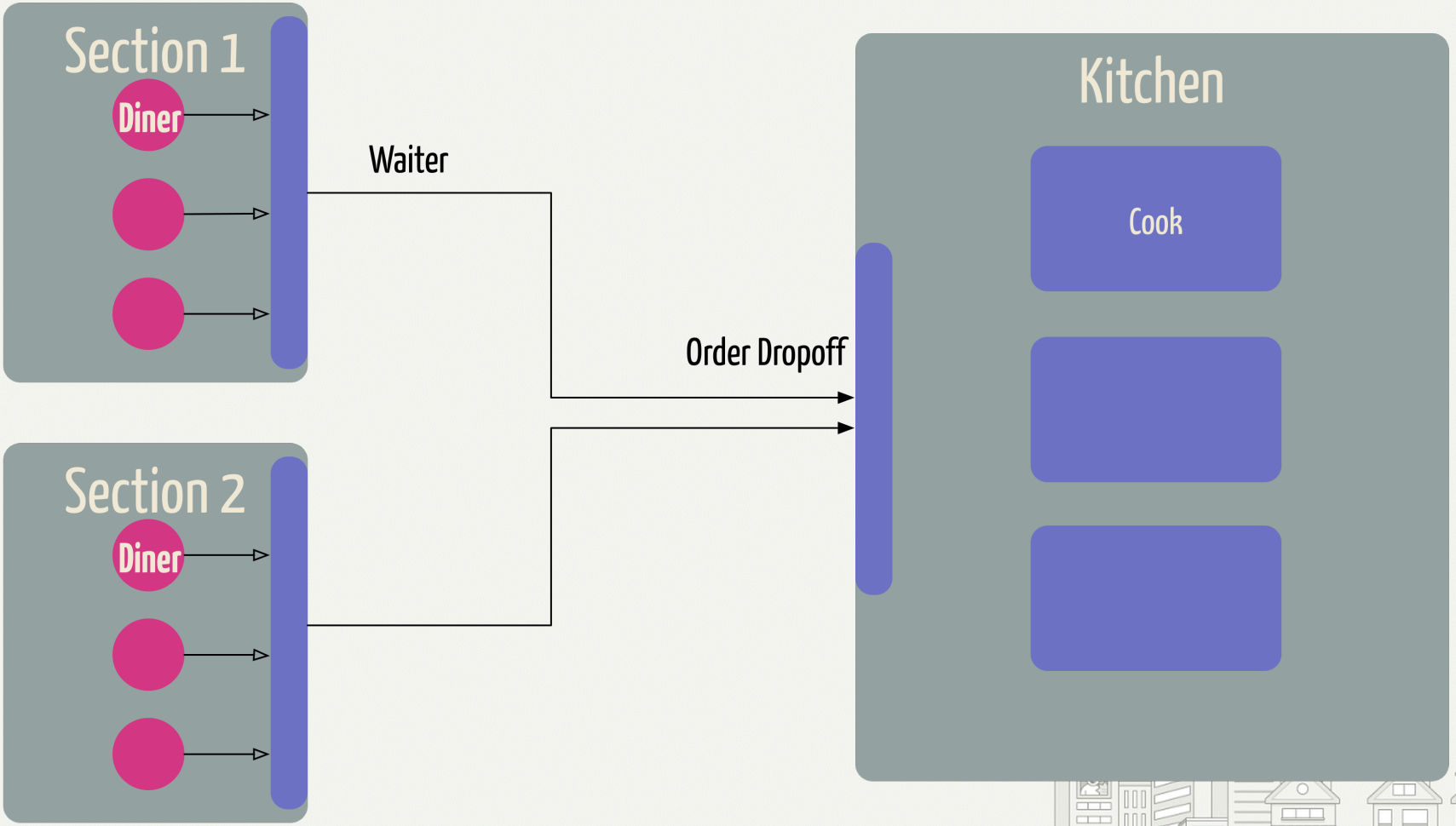

Jim's
RESTAURANT
2420 Mission Street
San Francisco, CA
415-285-6020
Side Orders
EXPRESSO DRINKS
Regular
Grande
Beverages

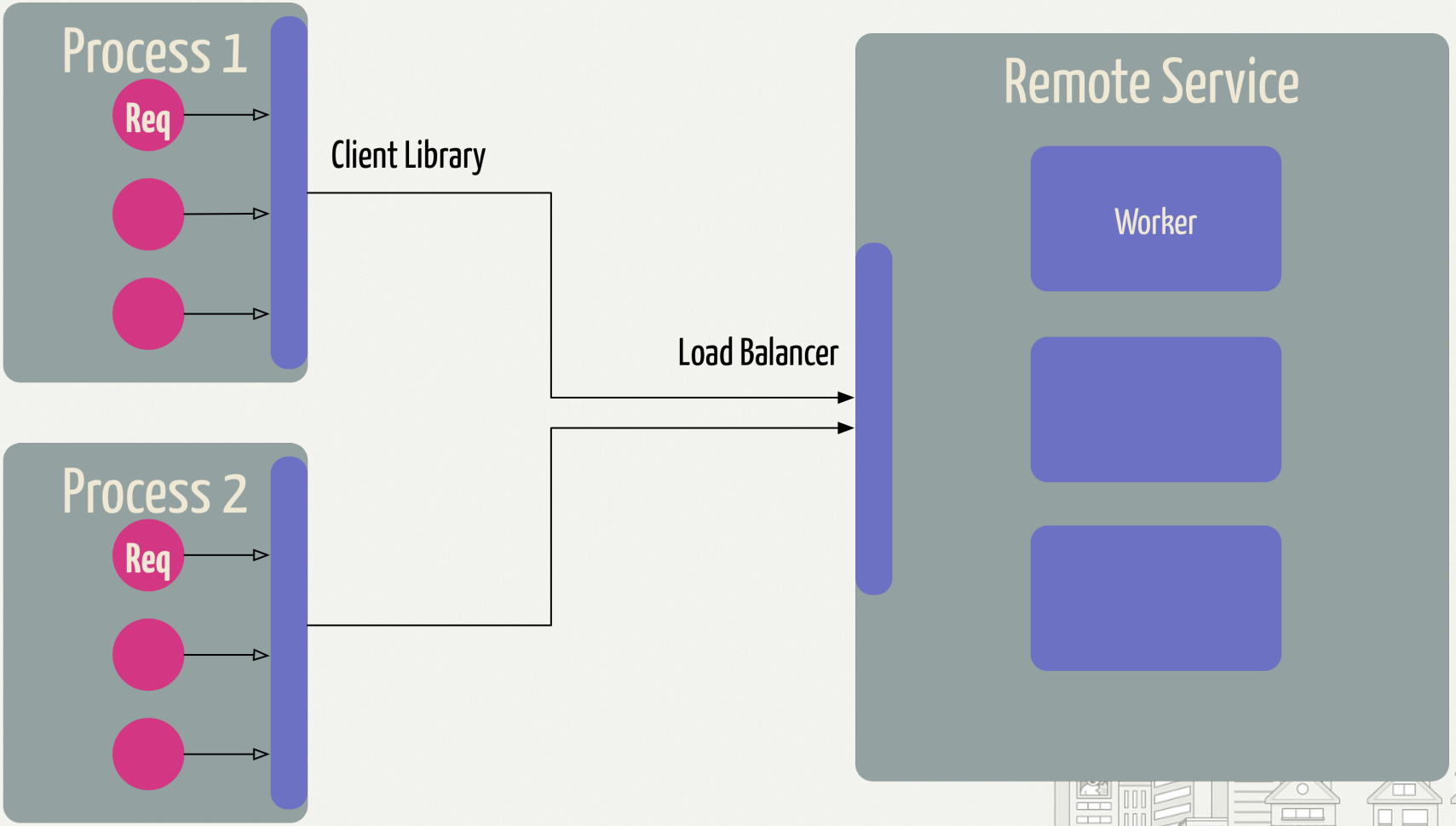
2

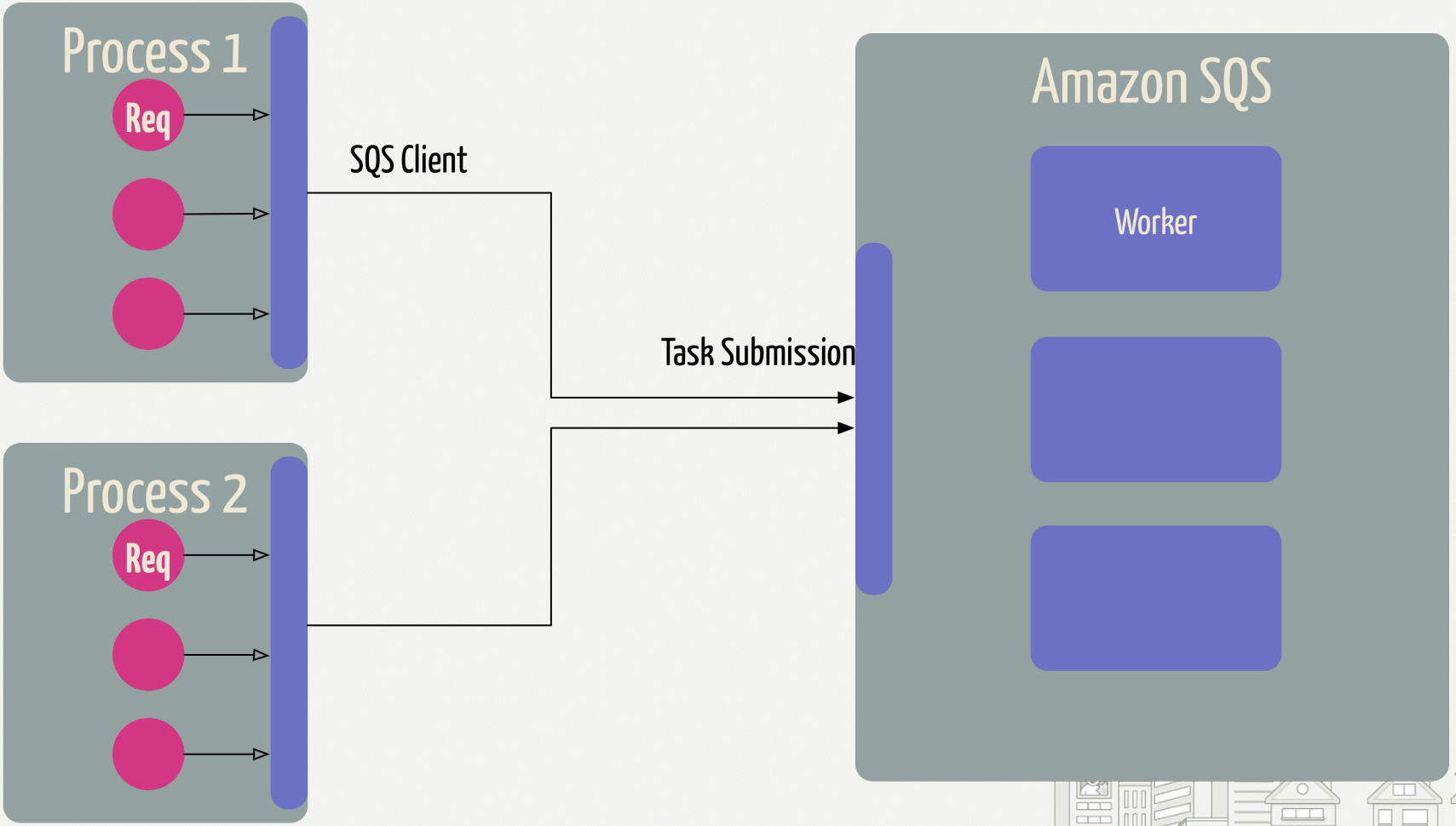
3

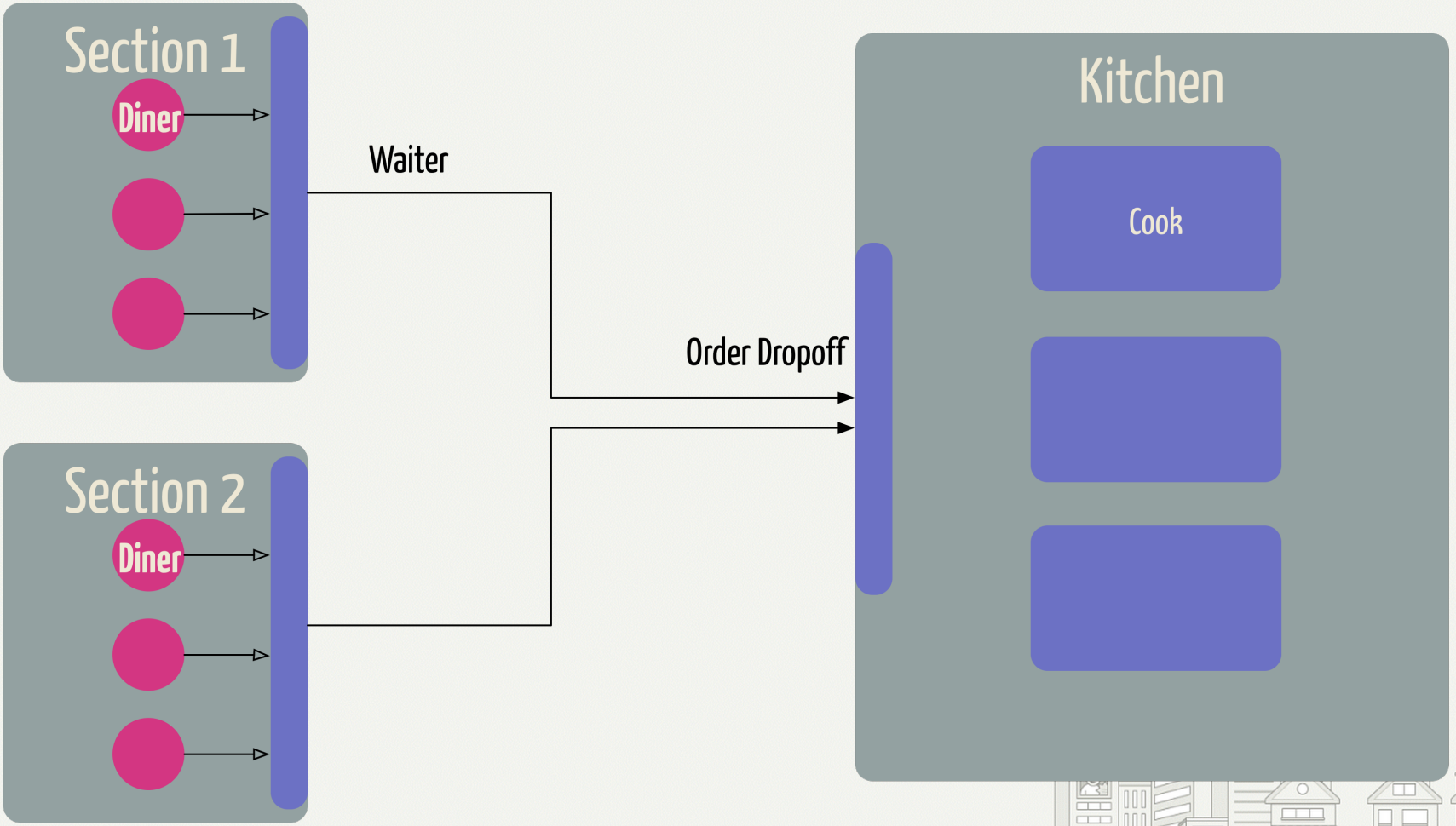
4

Nygard's circuit breaker


Section 1
Diner
Waiter
Section 2
Diner
Order Dropoff
Kitchen
Cook

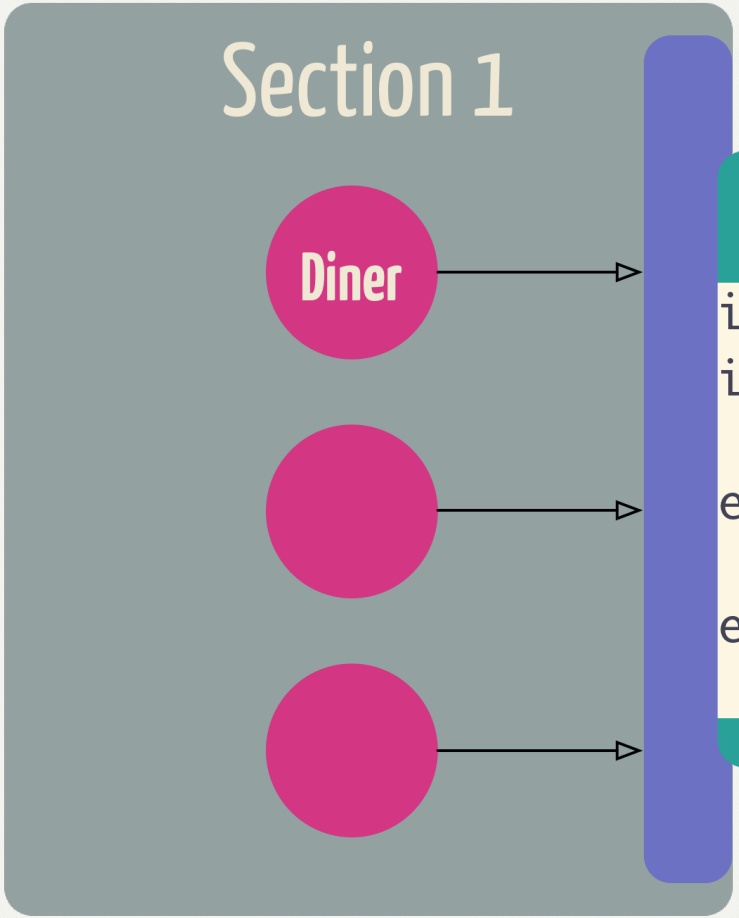

Process 1
Req
Client Library
Process 2
Req
Load Balancer
Remote Service
Worker


Process 1
Req
Process 2
Req
SQS Client
Task Submission
Amazon SQS
Worker


Section 1
Diner
Waiter
Kitchen
Cook
Order Dropoff
Section 2
Diner

Circuit Breaker States:

* Healthy (or “closed”)
* Recovering (or “half-open”)
* Unhealthy (or “open”)



CB Waiter

```
is_healthy = self.assess_if_healthy()
if is_healthy == HEALTHY:
  return self.issue_request()
elif is_healthy == RECOVERING:
  return issue_trial_request()
else:
  raise RequestBlockedError()
```

Recovery:

* Wait for `recovery_timeout` seconds
* Send a trial request, trust its results

Before a circuit breaker:

Before a circuit breaker:

* Diners wait forever to get food

Before a circuit breaker:

* Diners wait forever to get food
* Kitchen has a growing backlog

Before a circuit breaker:

* Diners wait forever to get food
* Kitchen has a growing backlog
* New diners making things worse

With a circuit breaker:

With a circuit breaker:

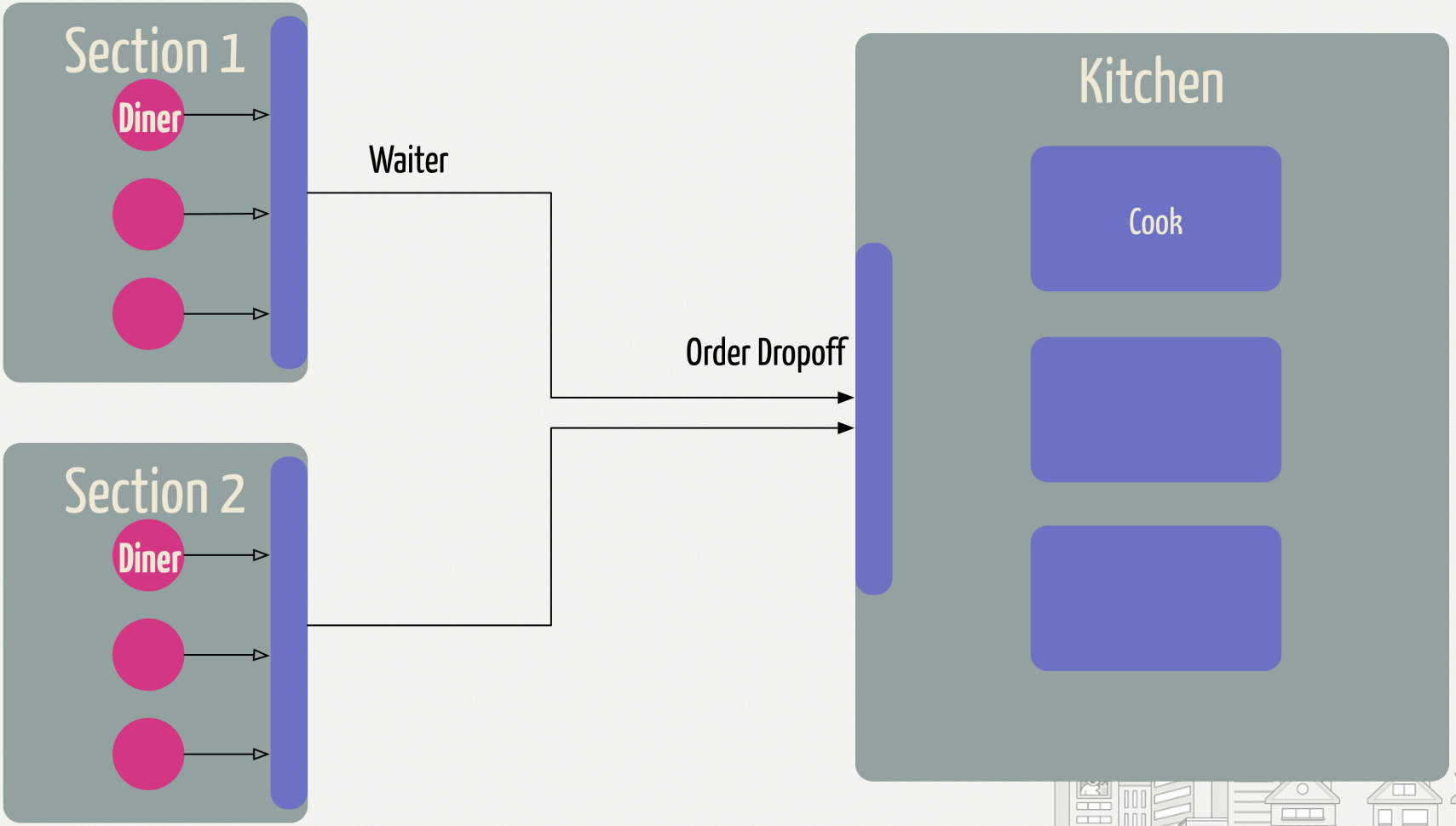
* Fewer frustrated users

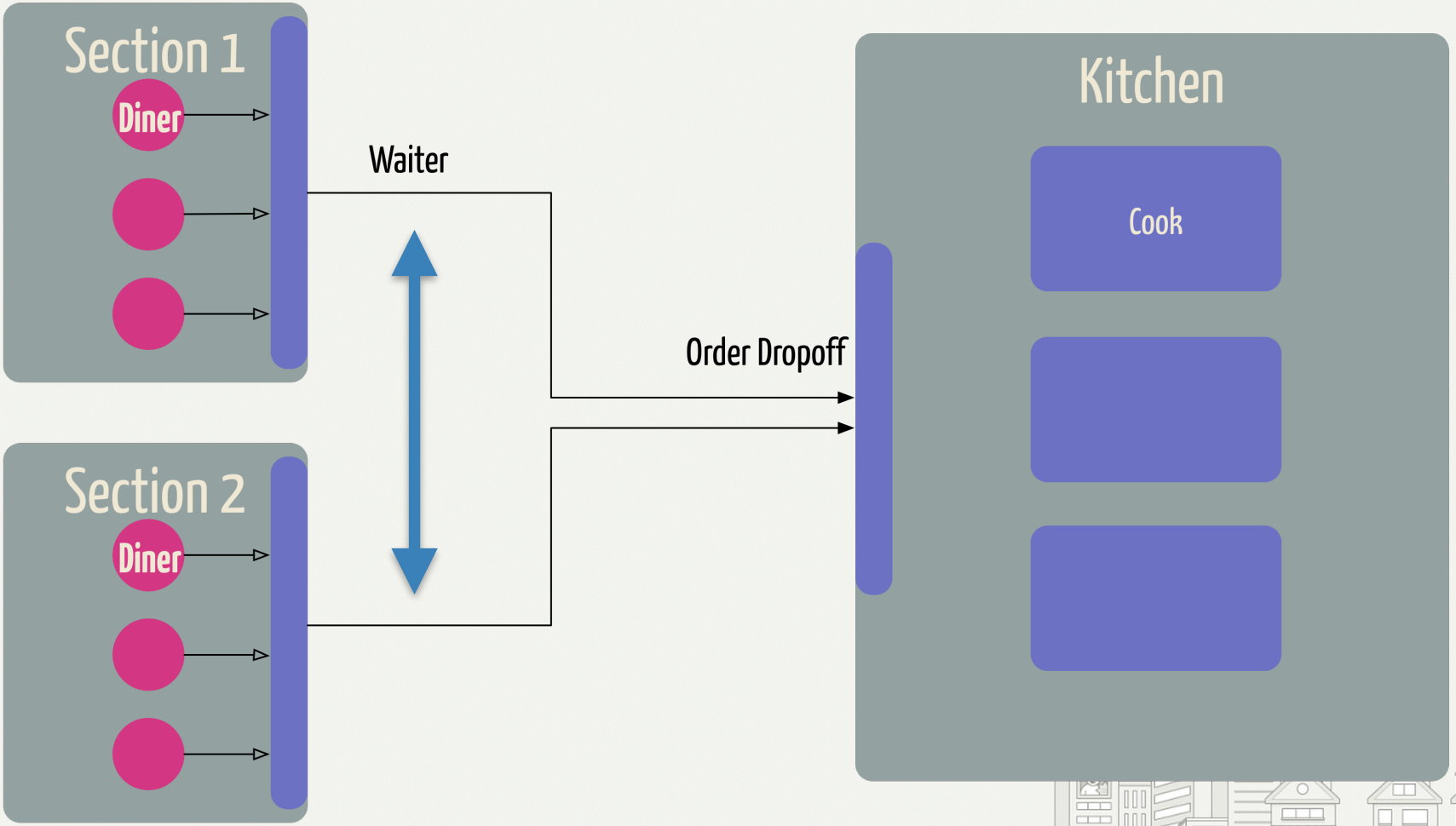
With a circuit breaker:

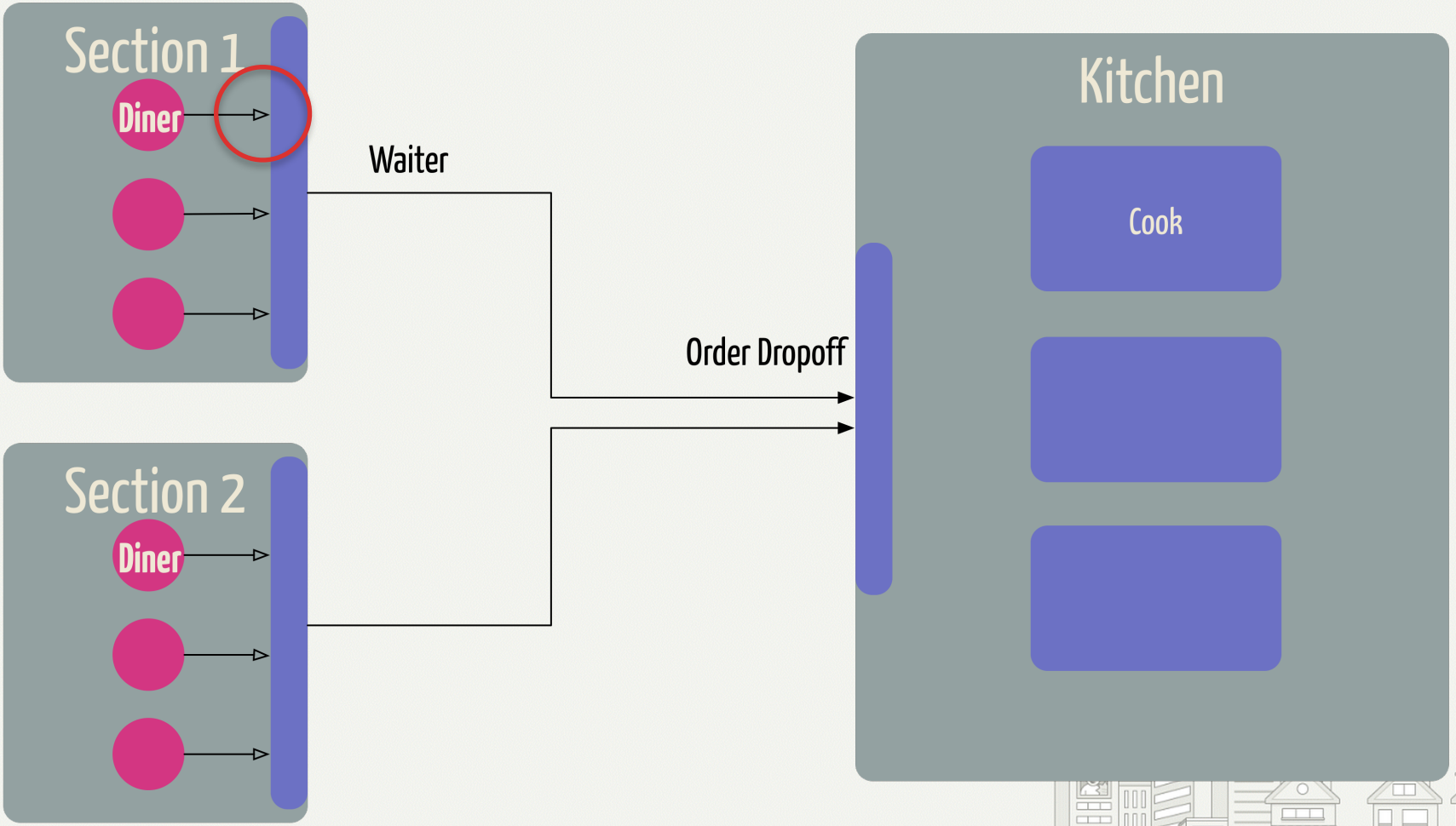
* Fewer frustrated users
* Reduced load on the backend

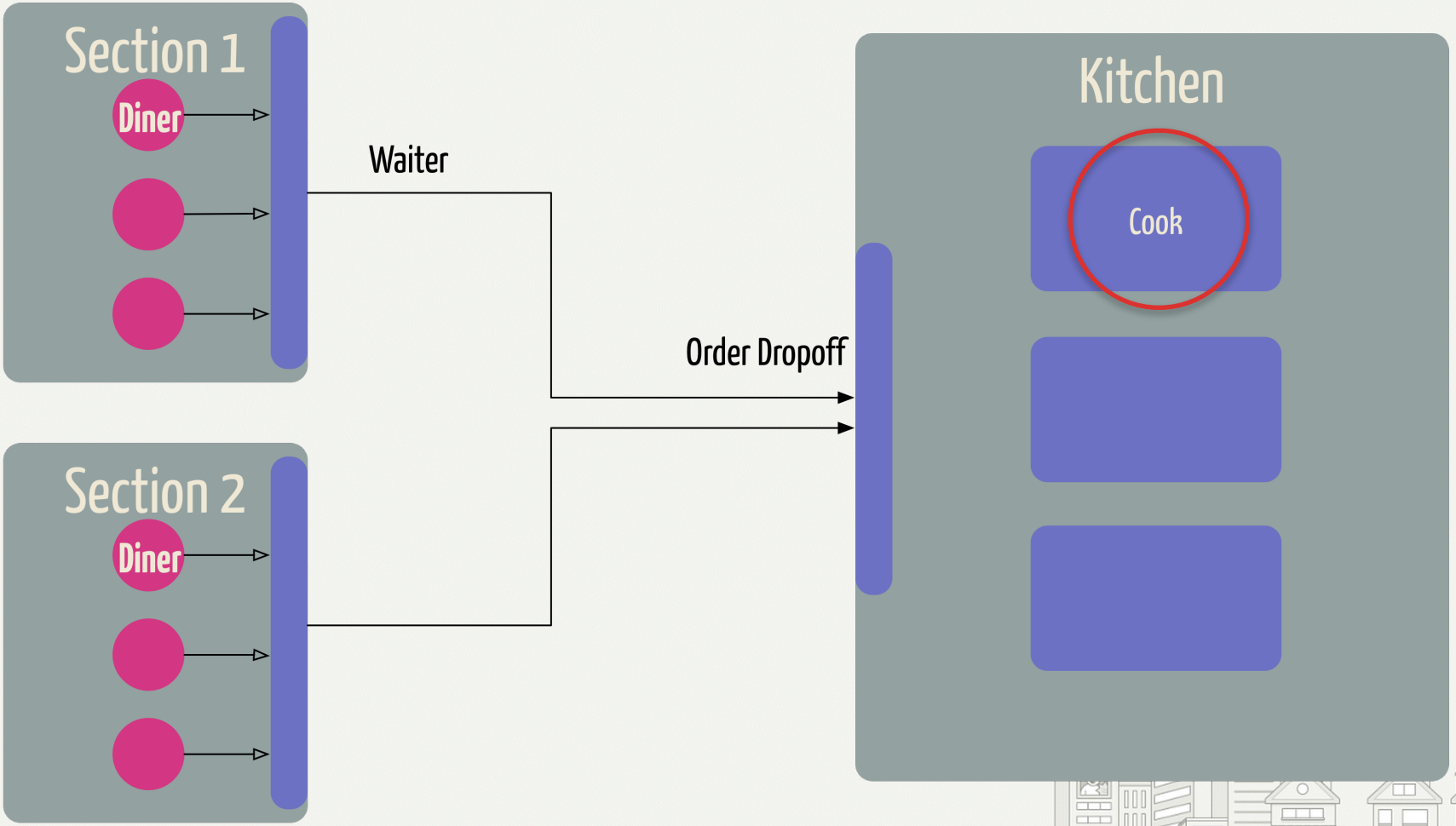
With a circuit breaker:

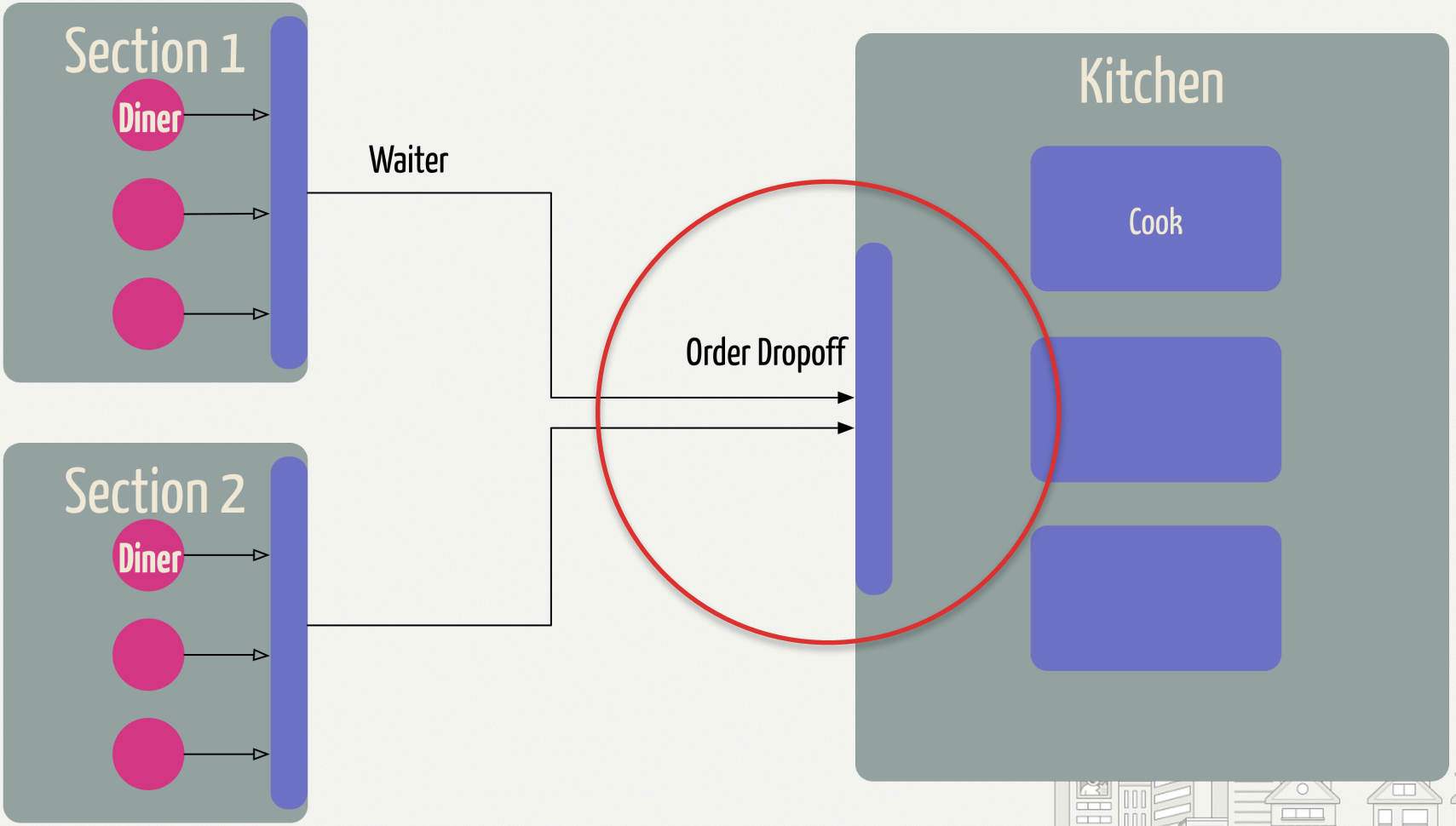
* Fewer frustrated users
* Reduced load on the backend
* A well defined failure mode


Section 1
Diner
Waiter
Section 2
Diner
Order Dropoff
Kitchen
Cook


Section 1
Diner
Section 2
Diner
Waiter
Order Dropoff
Kitchen
Cook

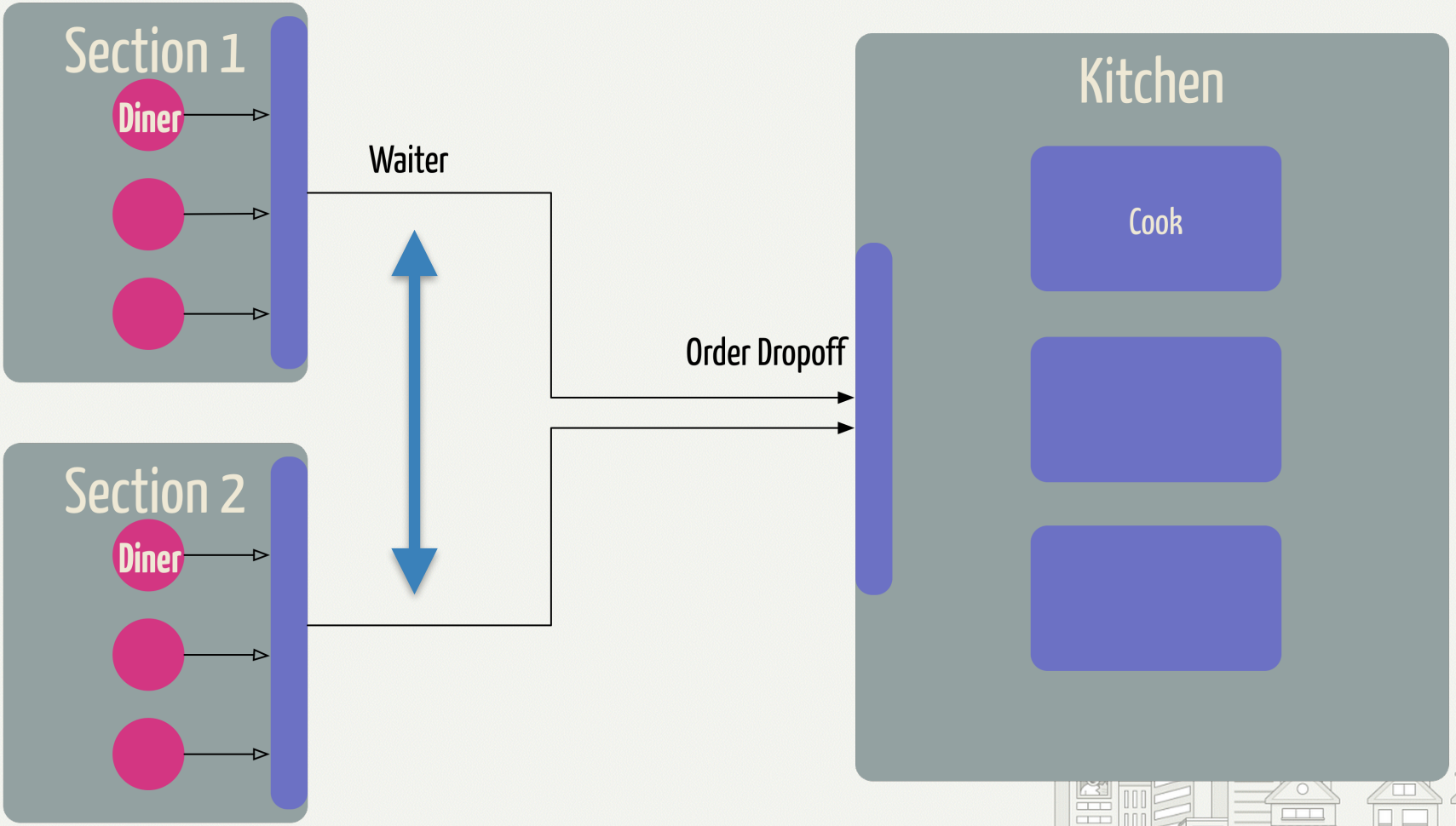

Section 1
Diner
Waiter
Kitchen
Cook
Order Dropoff
Section 2
Diner

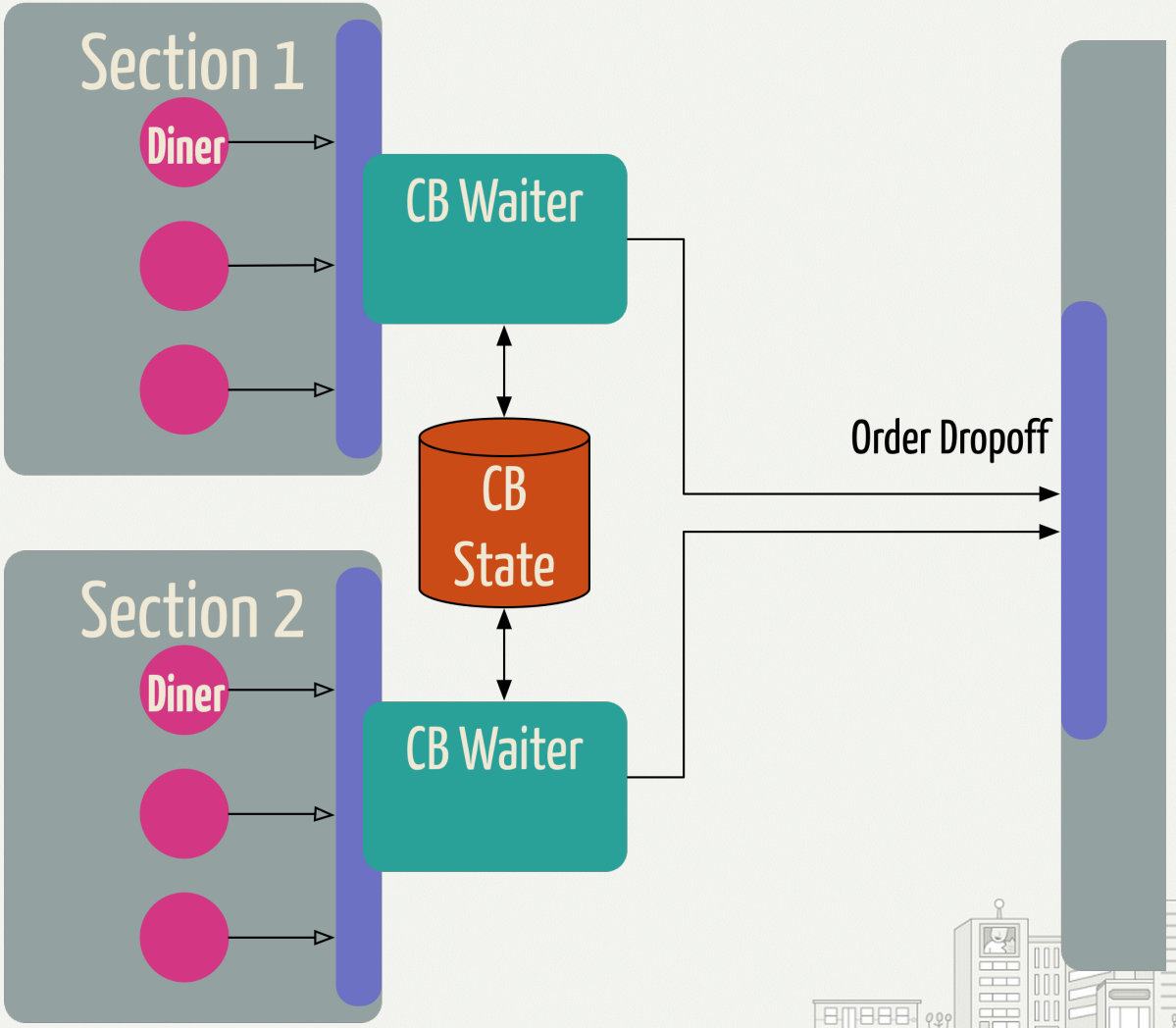

Section 1
Diner
Waiter
Kitchen
Cook
Order Dropoff
Section 2
Diner


Section 1
Diner
Waiter
Kitchen
Cook
Order Dropoff
Section 2
Diner

Module 1:

Should our waiters all agree?


Section 1
Diner
Section 2
Diner
Waiter
Order Dropoff
Kitchen
Cook


Section 1
Diner
CB Waiter
CB State
Order Dropoff
Section 2
Diner
CB Waiter

New Behavior:

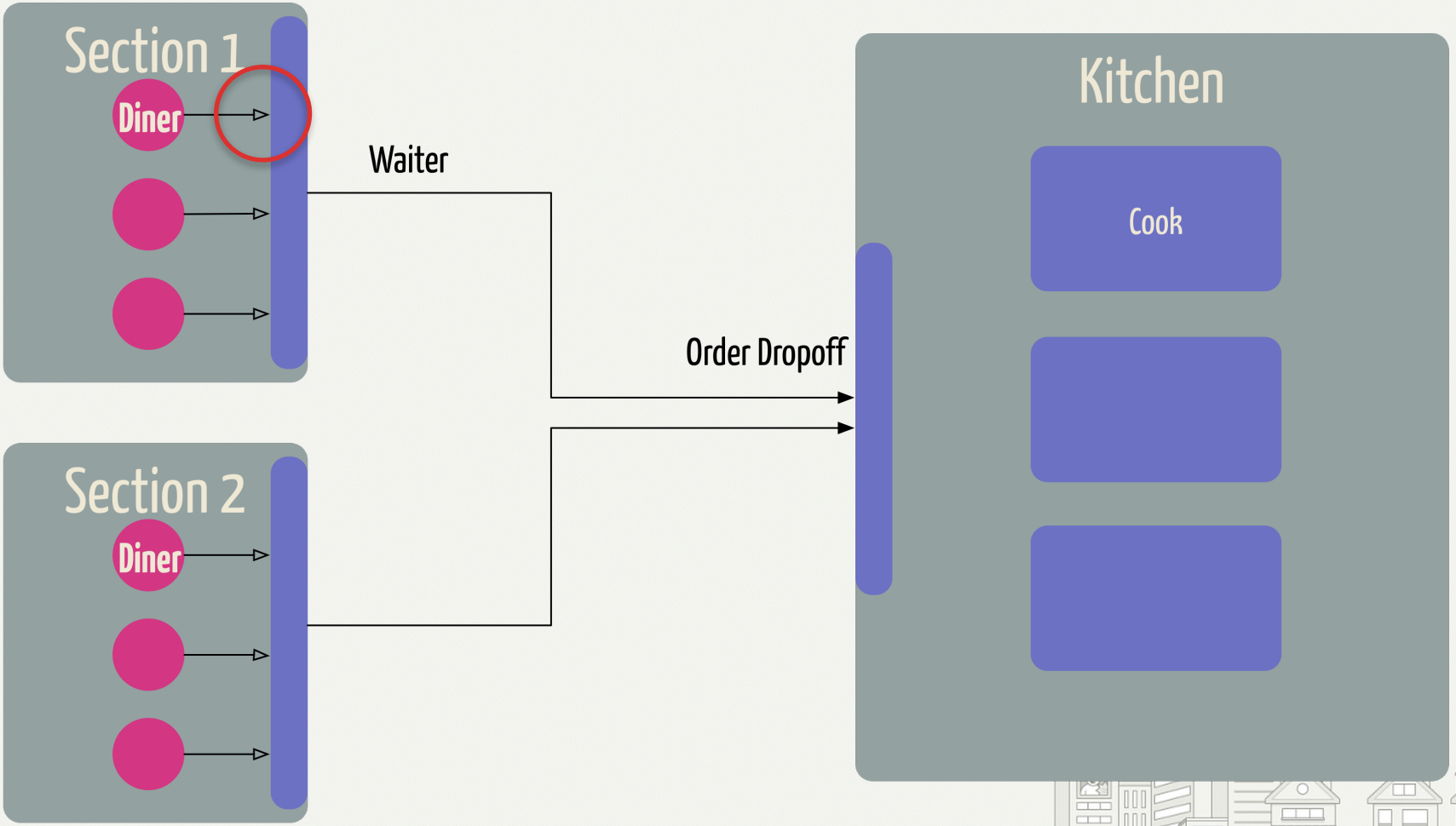
* Clients inform each other
* Processes are no longer independent

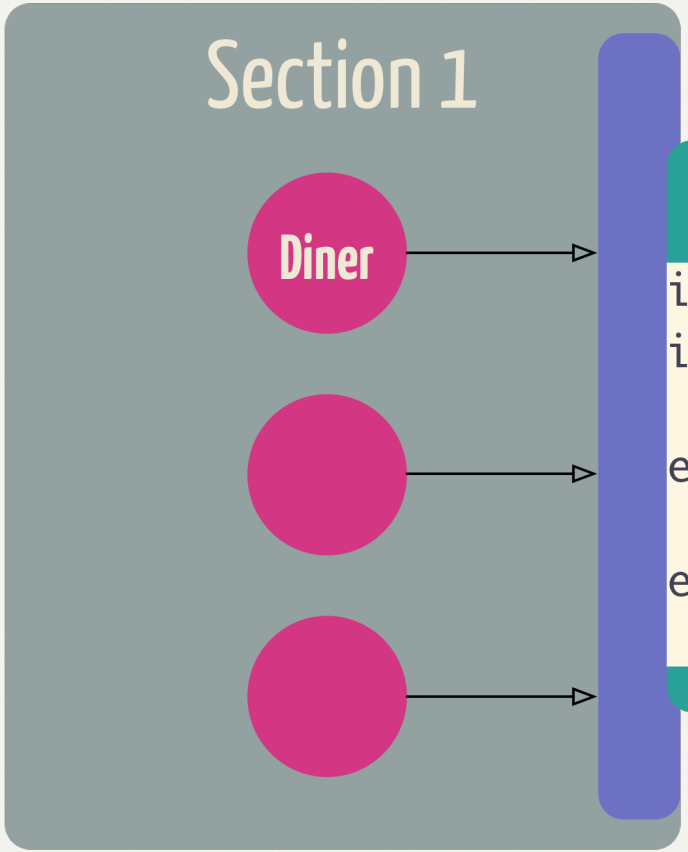
* Propagate failure faster

* Requires distributed datastore
* Forces decisions about consistency

Module 2:

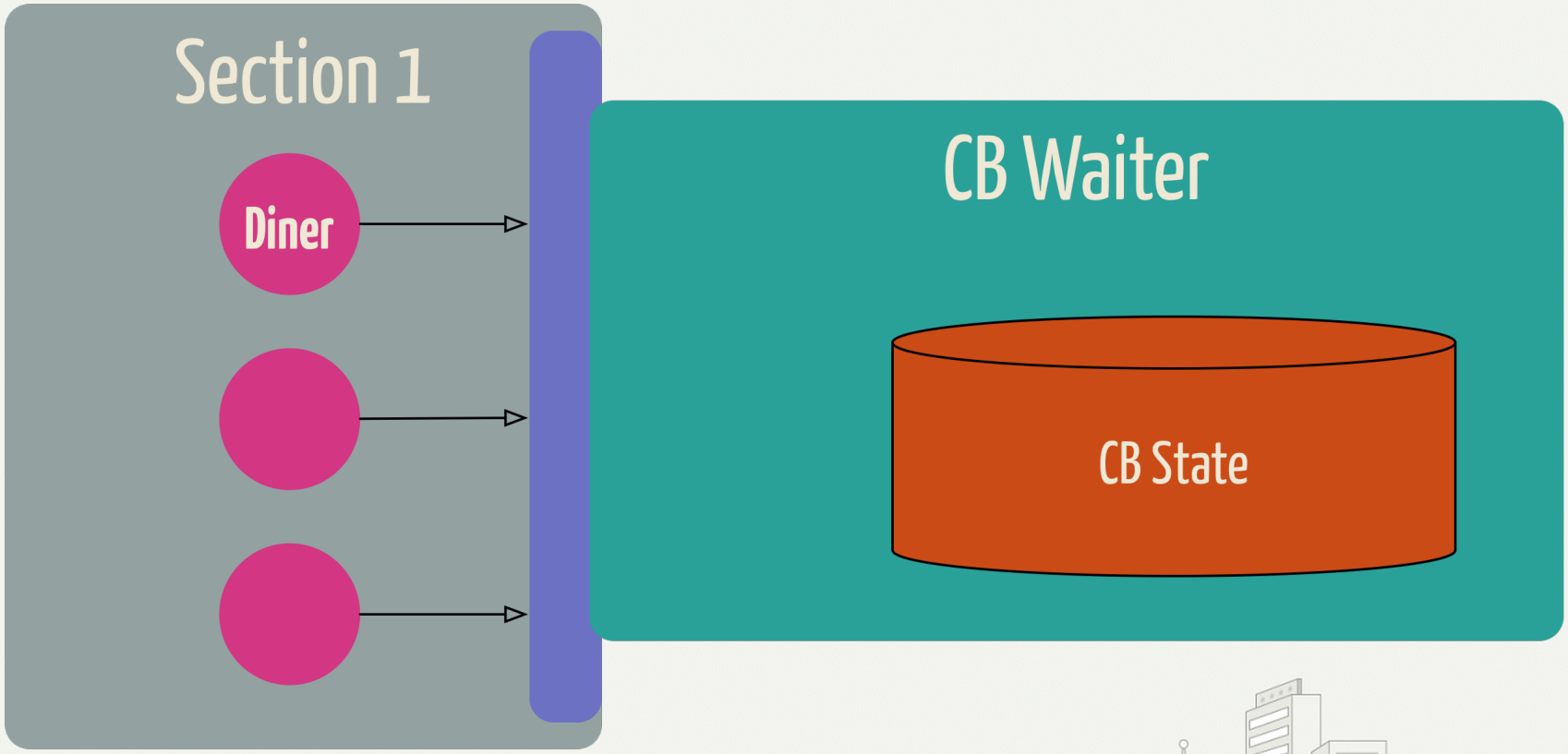
What should we do in response?

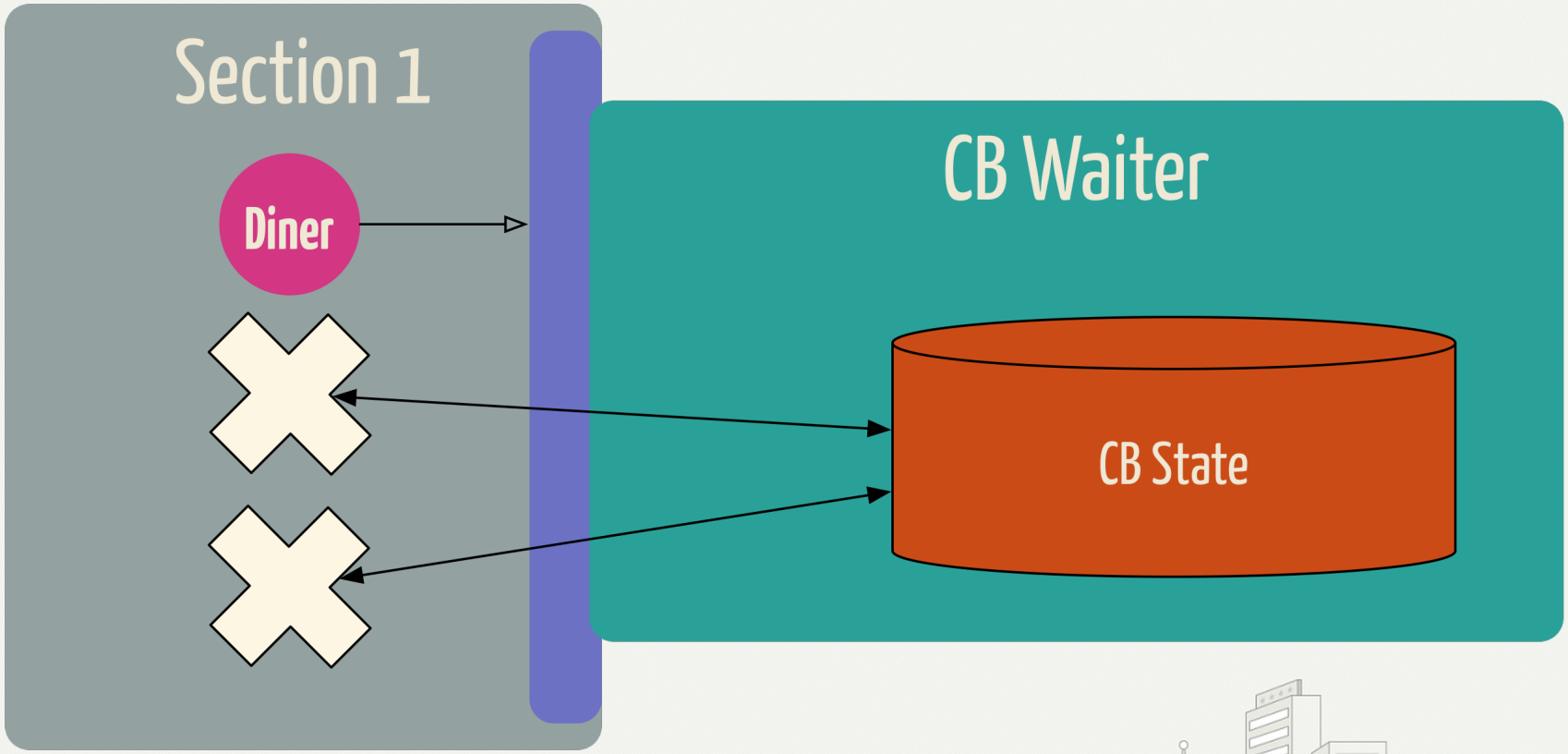

Section 1
Diner
Waiter
Kitchen
Cook
Order Dropoff
Section 2
Diner



CB Waiter

```
is_healthy = self.assess_if_healthy()
if is_healthy == HEALTHY:
  return self.issue_request()
elif is_healthy == RECOVERING:
  return issue_trial_request()
else:
  raise RequestBlockedError()
```


Section 1
Diner
CB Waiter
CB State


Section 1
Diner
CB Waiter
CB State

New Behavior:

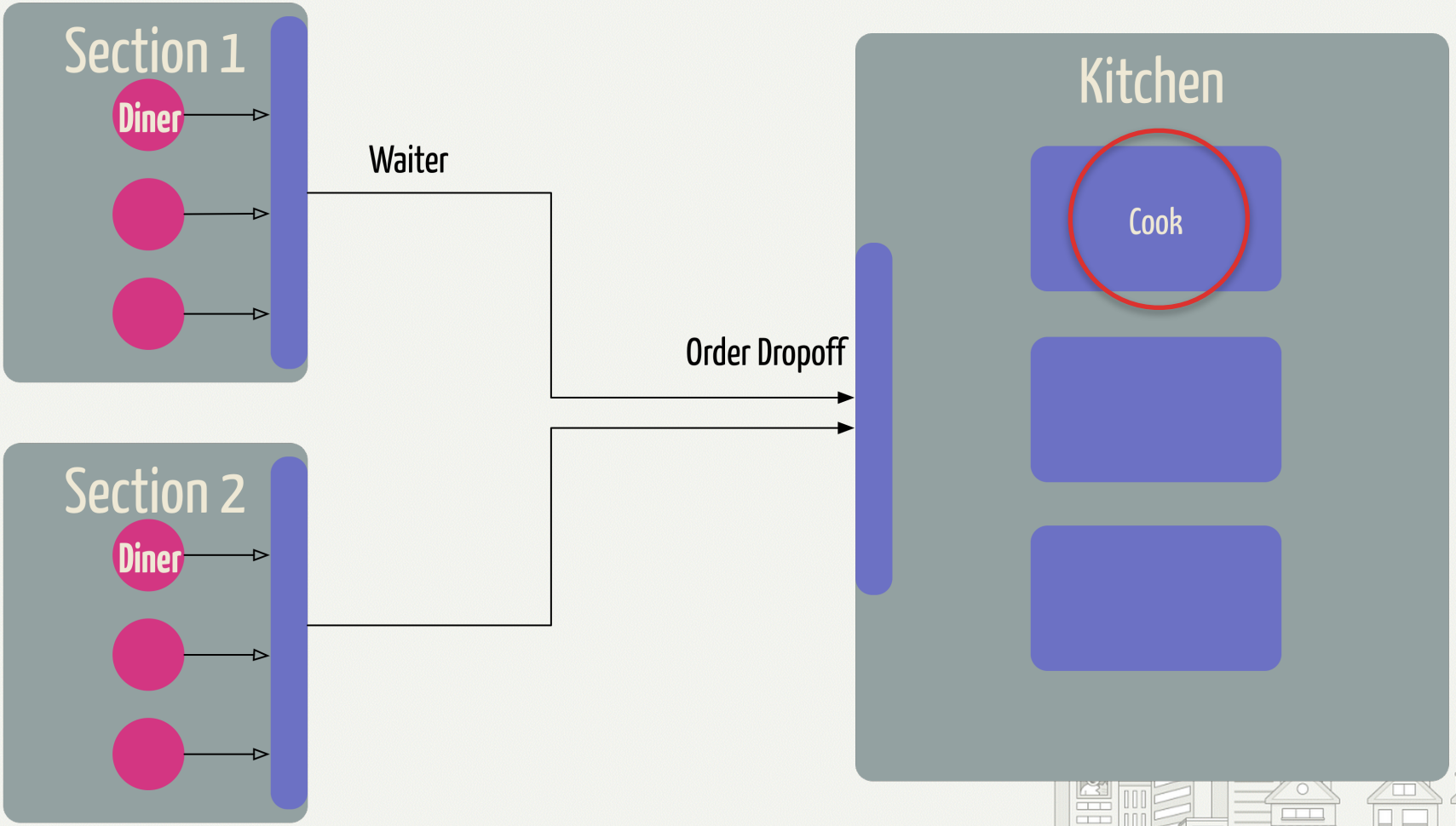
* Code can check in advance about healthiness of system
* Automatic monitoring!

* Build features on top of system health status

* Requires a single source of truth?

Module 3:

Who decides we're unhealthy?


Section 1
Diner
Waiter
Kitchen
Cook
Order Dropoff
Section 2
Diner

Kitchen

Cook

Kitchen Health Signal

```
def signal_overload(cb):
  if len(jobs) > THRESH:
    cb.mark_unhealthy()
```

New Behavior:

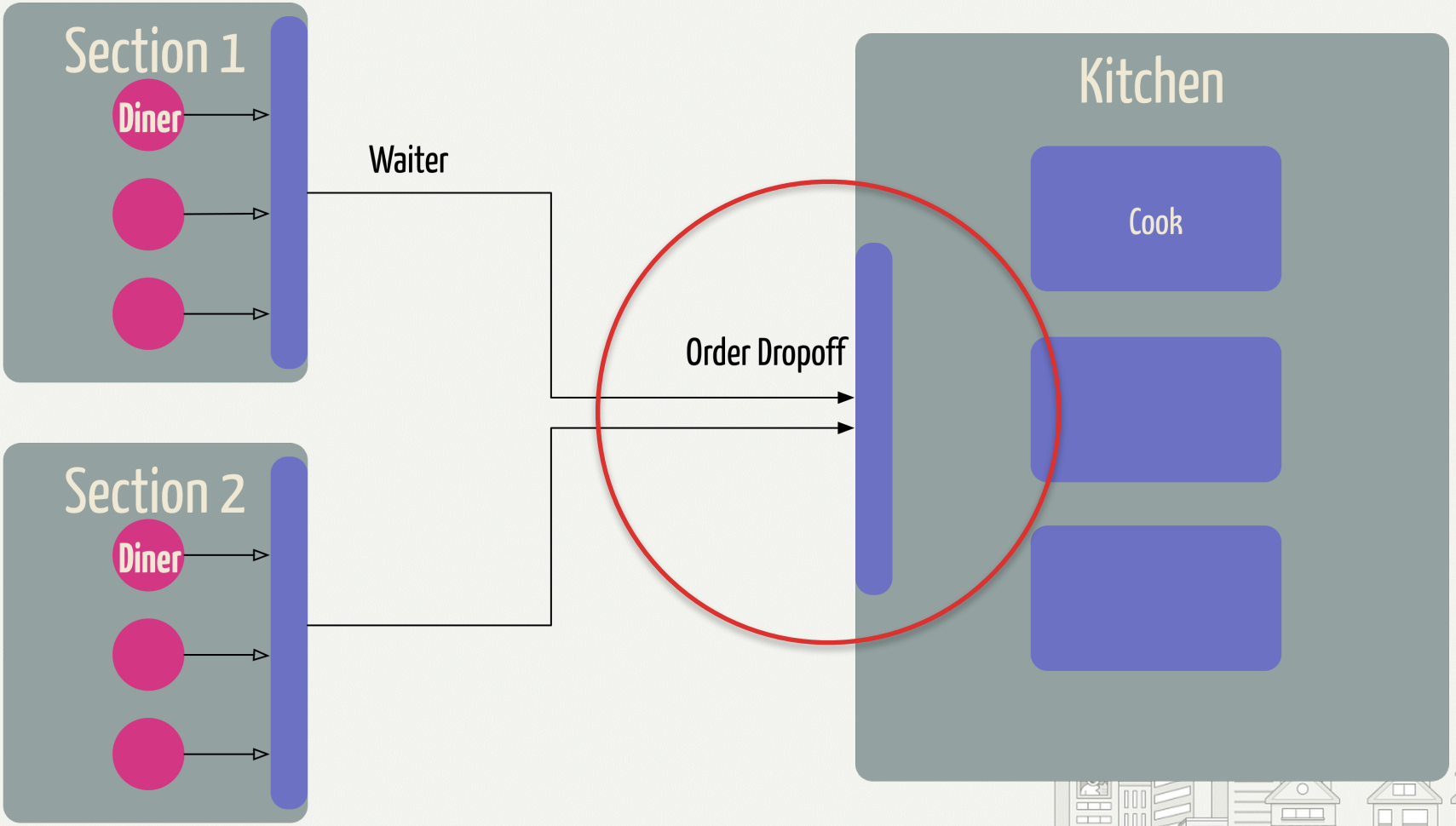
* CB gets signals from anywhere
* Signal combining logic

* Allows many (many) new signals

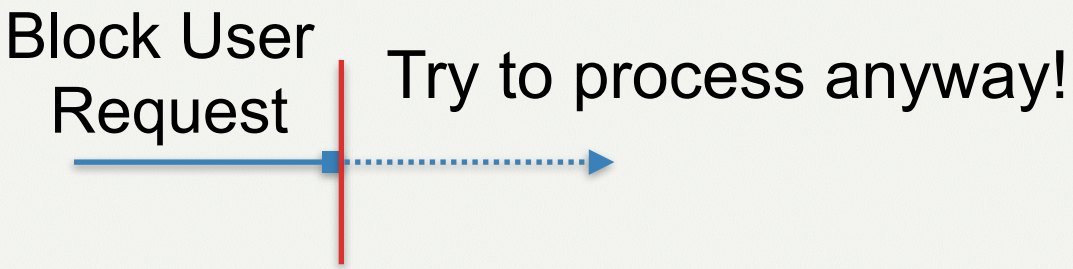
* Must combine signals
* Adds complexity to system

Module 4:

How do we recover?


Section 1
Diner
Waiter
Kitchen
Cook
Order Dropoff
Section 2
Diner

Dark launch:



* Reject but process normally
* Dangerous with side effects

Synthetic:

* Dark launching with fake requests
* Not necessarily representative

New Behavior:

* Traffic determines health
* Removal of recovery timeouts

* Faster(?) recovery
* No timeout tuning required
* Dark launching not always possible
* Synthetic can be unrepresentative

in summary

Your system will fail, have a plan!

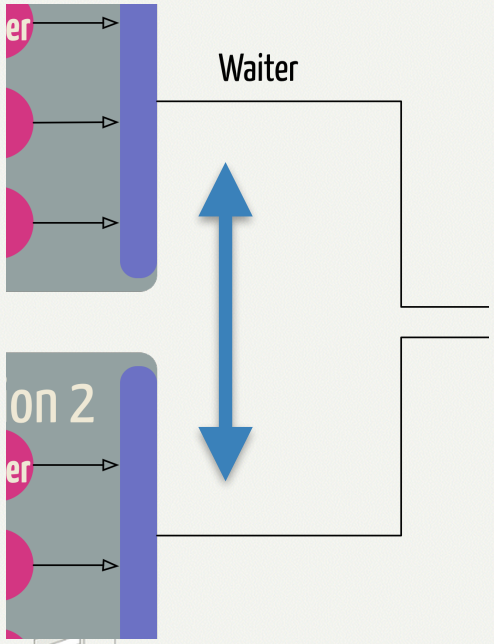
The basic CB is better than nothing

Questions to ask:

* Should our waiters all agree?
* How should I deal with unhealthiness?
* Who decides we're unhealthy?
* How do we recover?

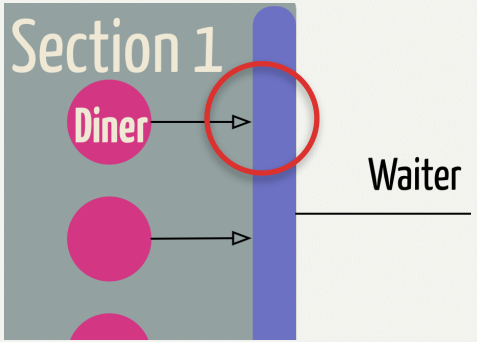
Questions to ask:

* Should our waiters all agree?
* How should I deal with unhealthiness?
* Who decides we're unhealthy?
* How do we recover?



Questions to ask:

* Should our waiters all agree?
* How should I deal with unhealthiness?
* Who decides we're unhealthy?
* How do we recover?

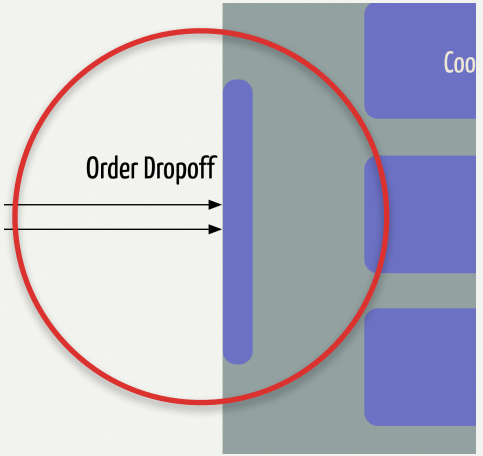


Questions to ask:

* Should our waiters all agree?
* How should I deal with unhealthiness?
* Who decides we're unhealthy?
* How do we recover?

Questions to ask:

* Should our waiters all agree?
* How should I deal with unhealthiness?
* Who decides we're unhealthy?
* How do we recover?



...and much more!

Much comes down to your use case

Questions?

striglia@yelp.com
@scott_triglia

Can’t we do better than rejecting requests?

http://techblog.netflix.com/2011/12/making-netflix-api-more-resilient.html

How do I safely test out a new circuit breaker?

https://engineering.heroku.com/blogs/2015-06-30-improved-production-stability-with-circuit-breakers/