Microservices.
Test smarter, not harder.

Beth Skurrie (Pactflow)
@bethesque

PACTFLOW

DIUS
WHERE IDEAS ARE ENGINEERED

#YOWPER

A big problem

Problem:
Long time to market

Microservices!

Solved a problem!
New problem...

Microservices

Solved problems

- Feature time to market

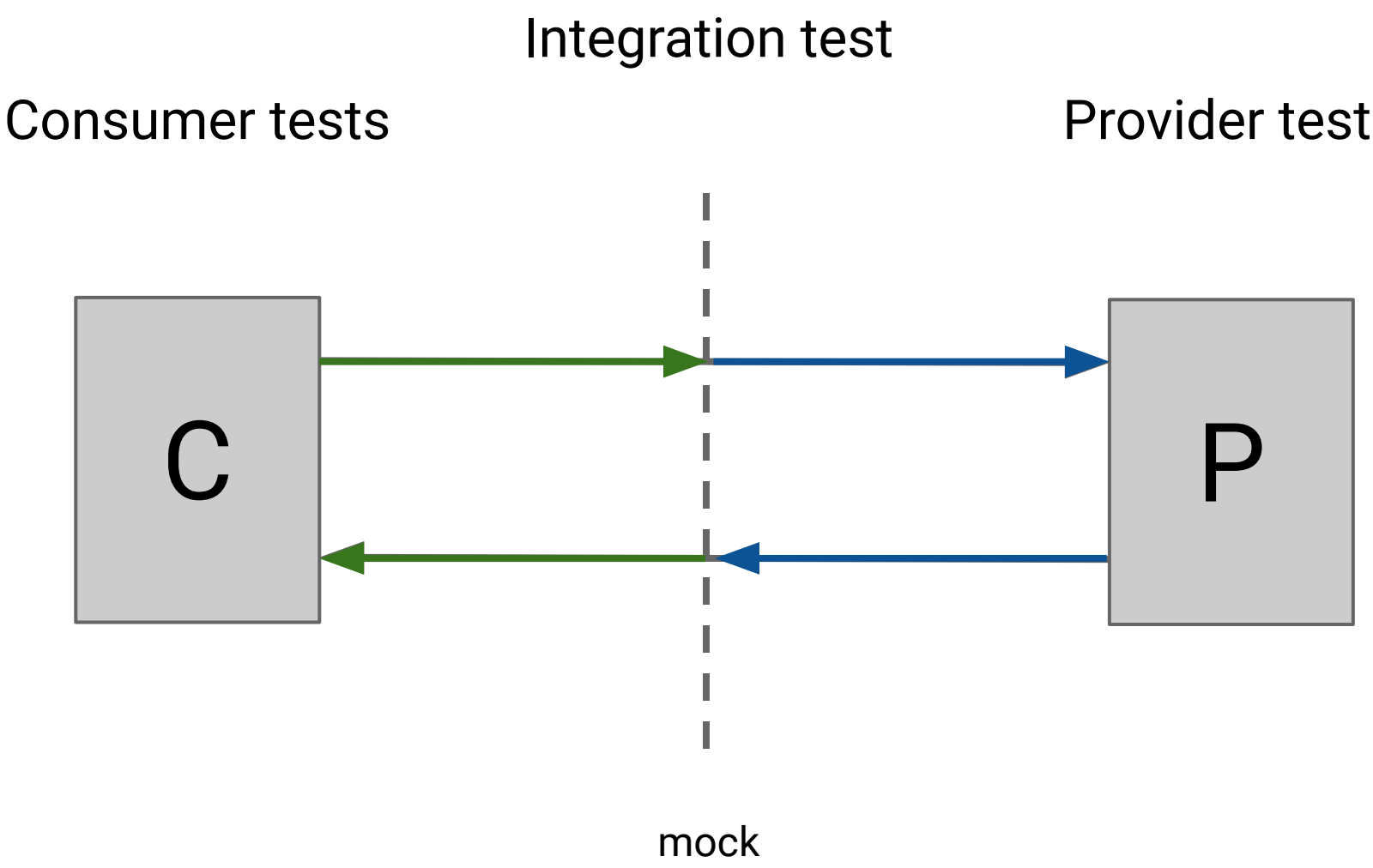
New problems

- E2E integration tests
  - Slow tests
  - Easy to break
  - Hard to fix
  - Scale badly
  - Lots of set up
  - Flakey tests ignored
  - Takes dev time away from features

E2E test suites re-couple your decoupled microservices

To E2E test, or not to E2E test?

What if there was another way?


Integration test
Consumer tests
Provider tests
C
P
mock

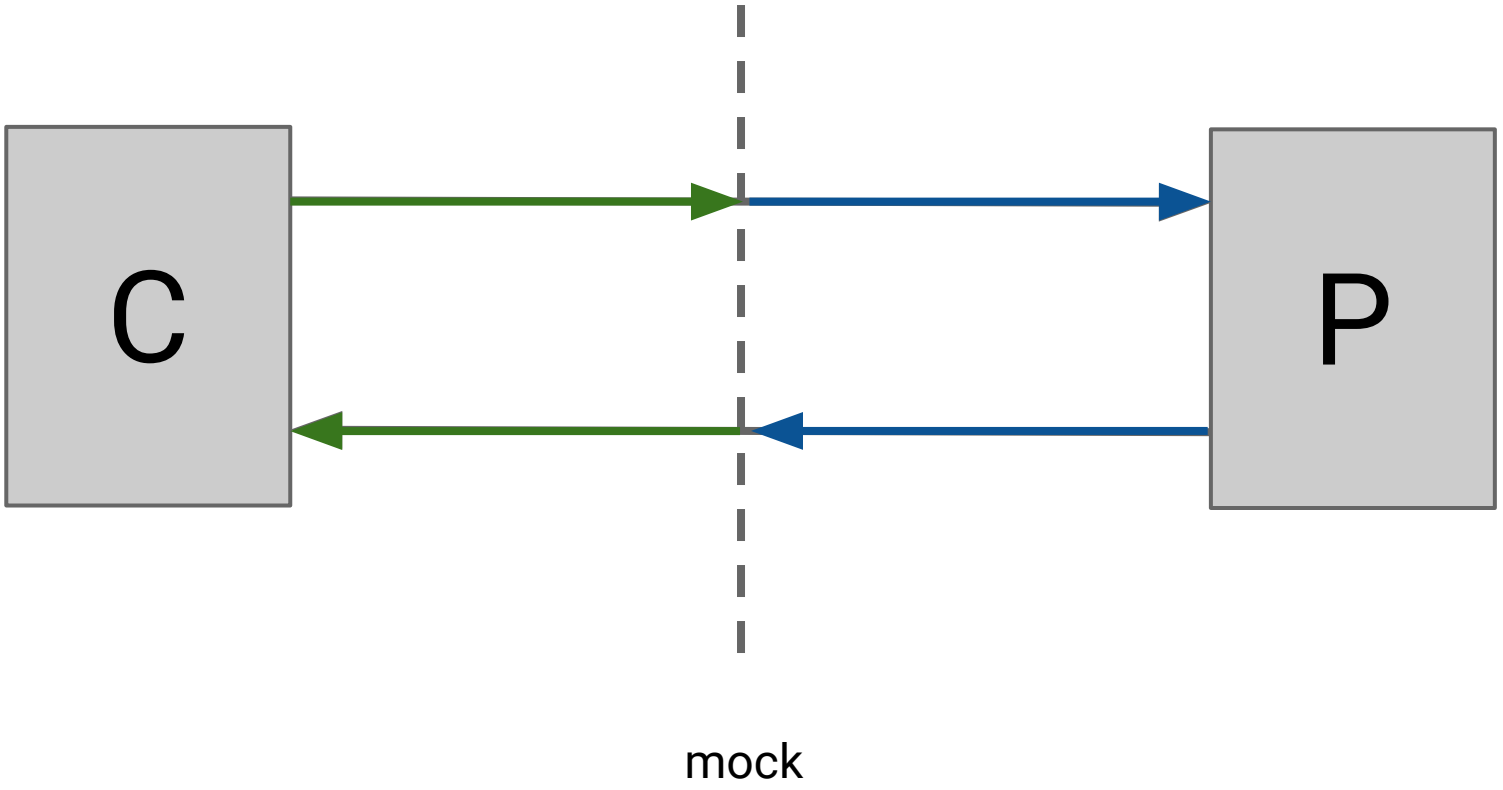
Test symmetry

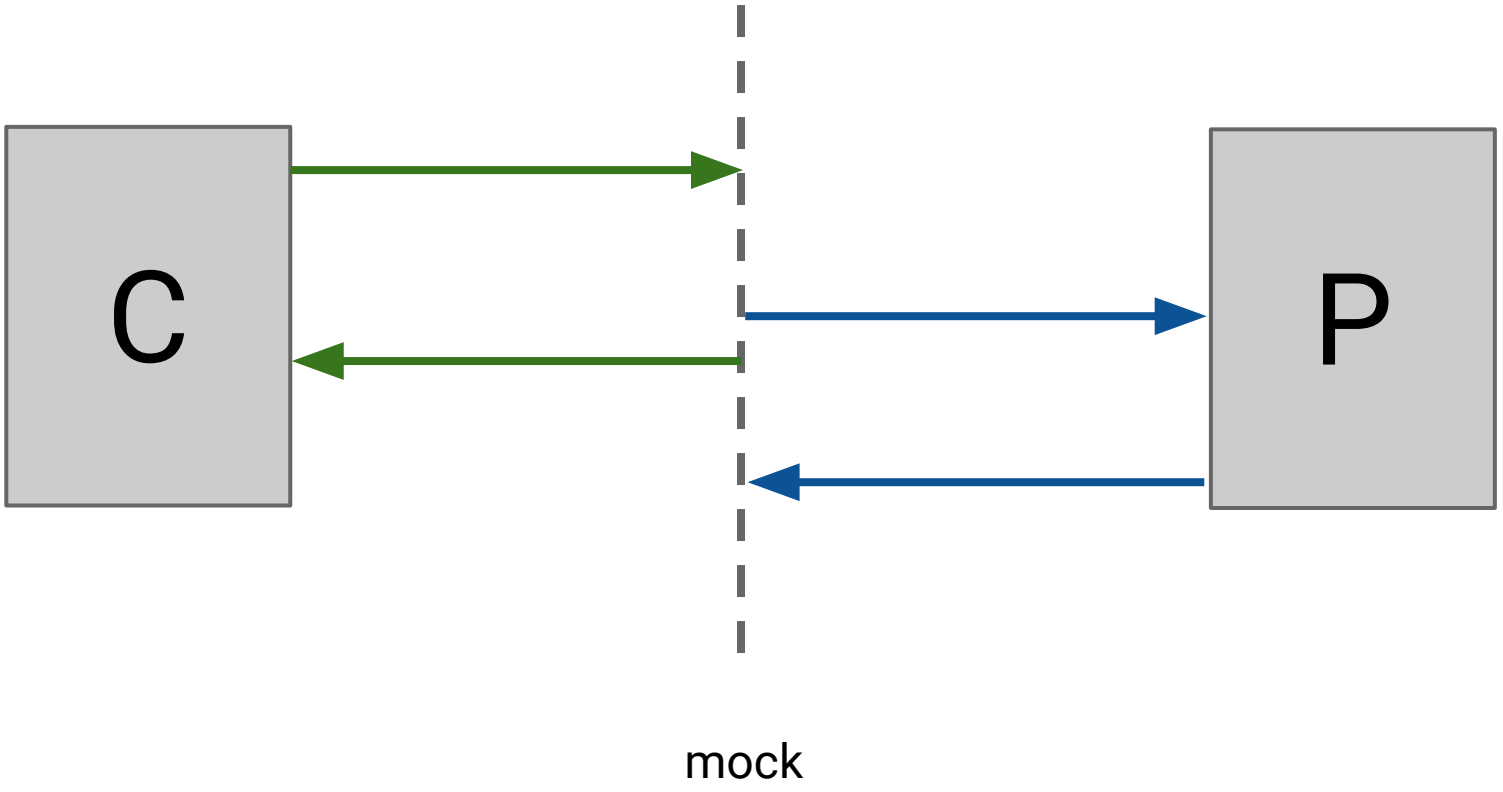
Solved problems

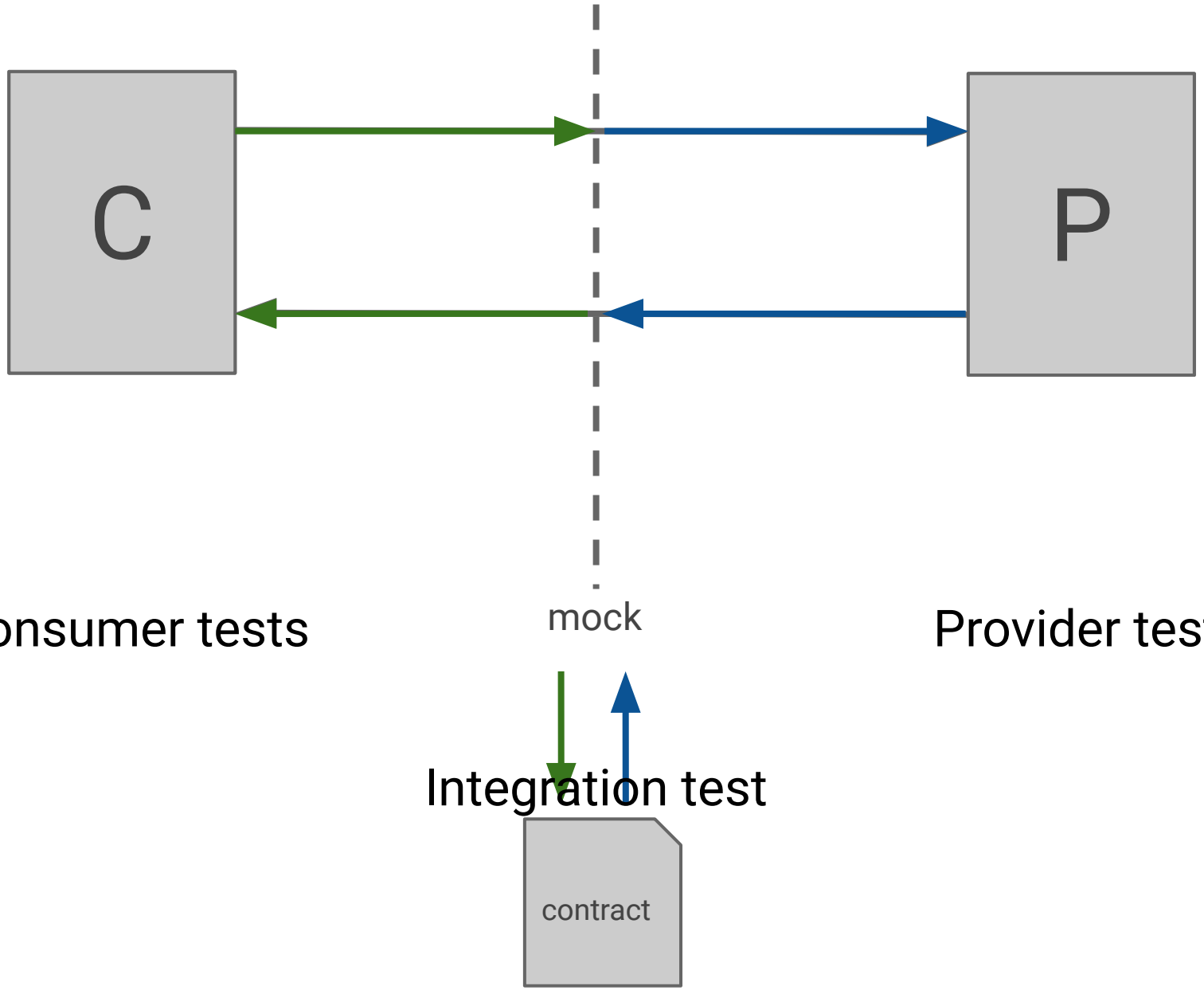
- Fast feedback
- Few dependencies
- No dedicated environment
- Reliable
- Easy to debug

New problems

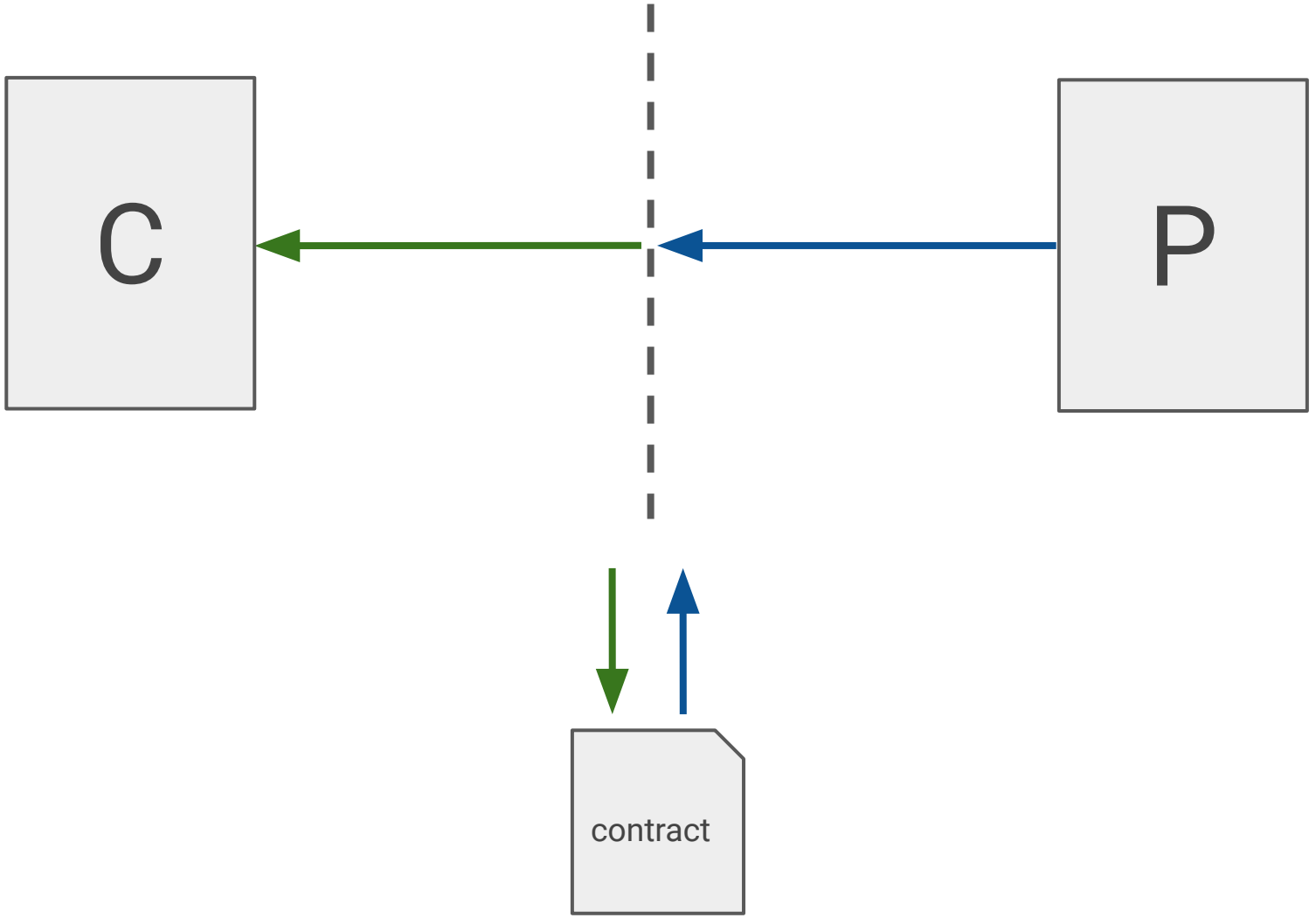
- Hard to keep both sides in sync


C
P
mock


C
P
mock


C
P
Consumer tests
mock
Provider tests
Integration test
contract

Message Contract



Contracts

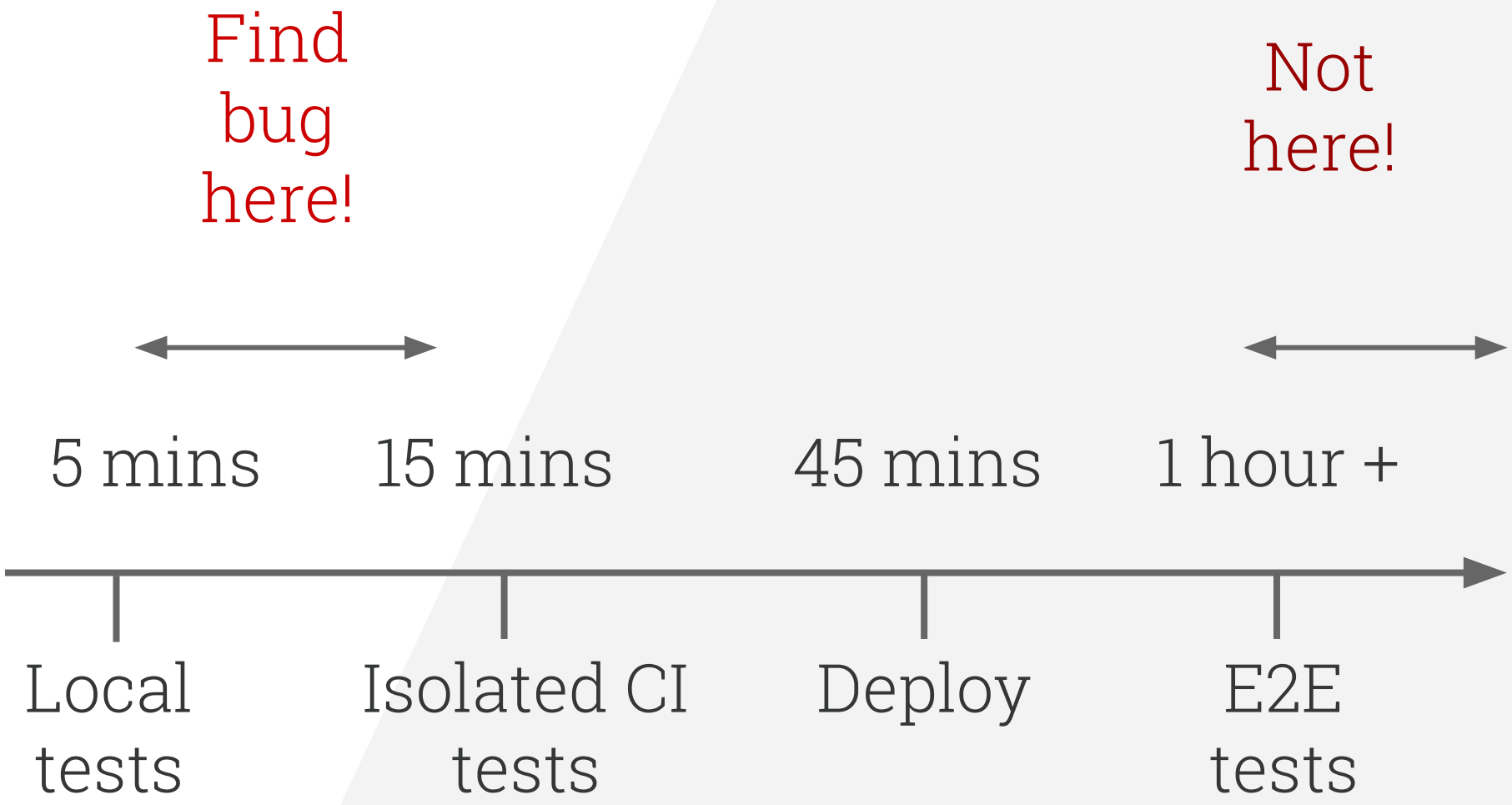
Solved problems

- Keeping tests in sync

New problems

- ???

How long between writing the code and finding the bug?


Find bug here!
Not here!
5 mins
15 mins
45 mins
1 hour +
Local tests
Isolated CI tests
Deploy
E2E tests

Contracts tests

Know before you commit

Contracts tests

Make changes with
speed and confidence

Contracts tests

Deploy independently

(Consumer) Contracts tests

Better API design

Contracts tests

Are **not** functional tests

Contracts tests

Are **not** an API specification

(Consumer) Contracts tests

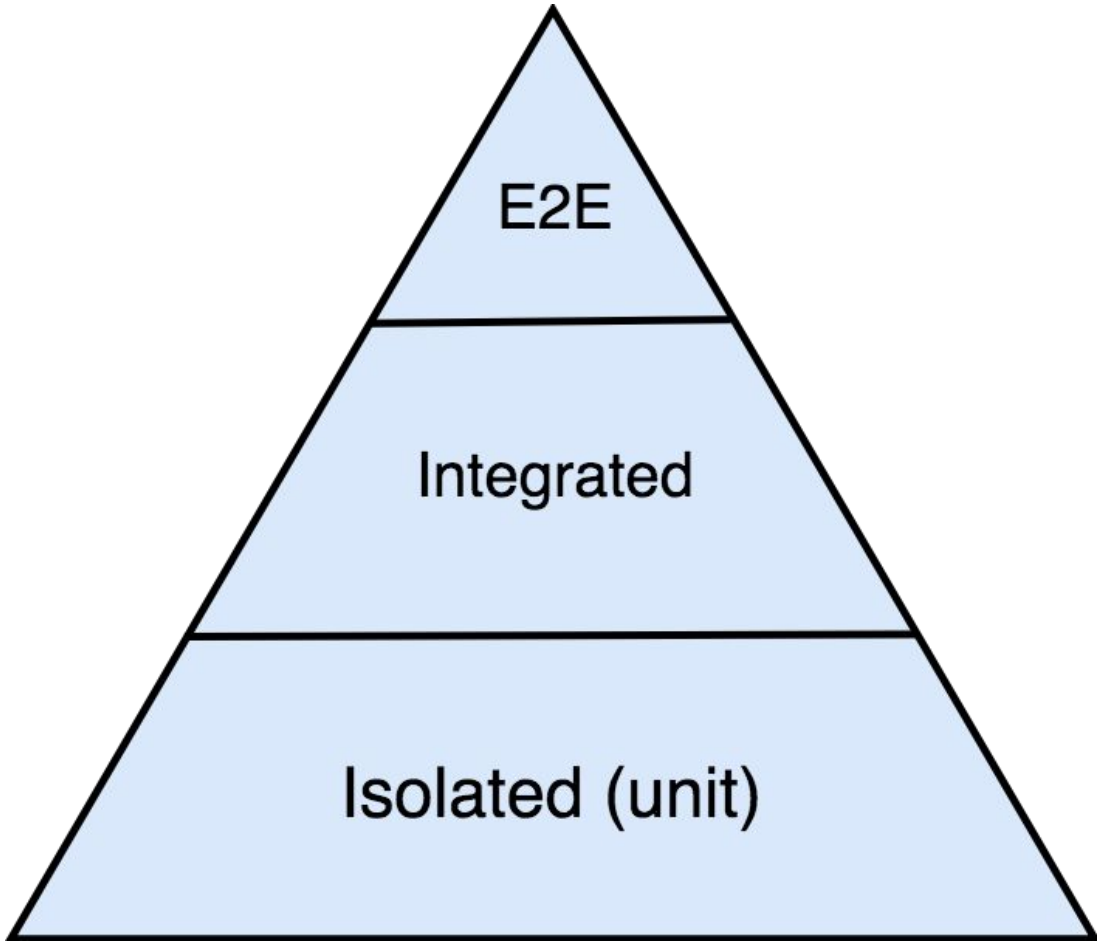
Are **not** good for “public” APIs (many, unknown consumers)

Contracts tests

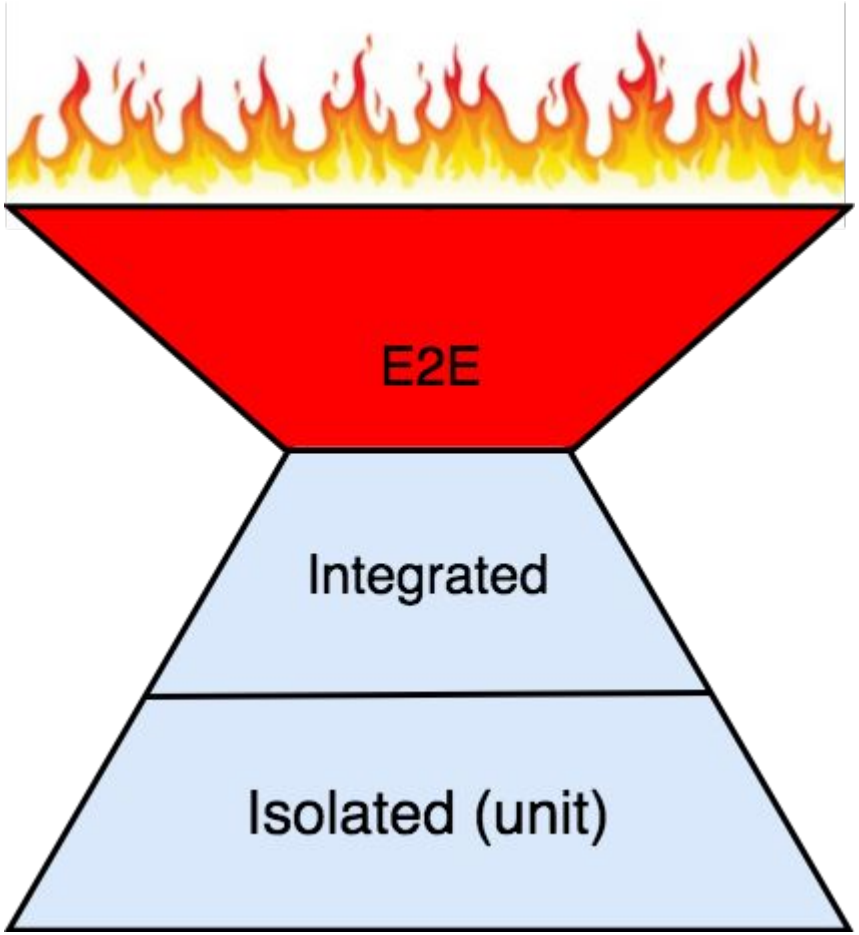
Are **not** a silver bullet!

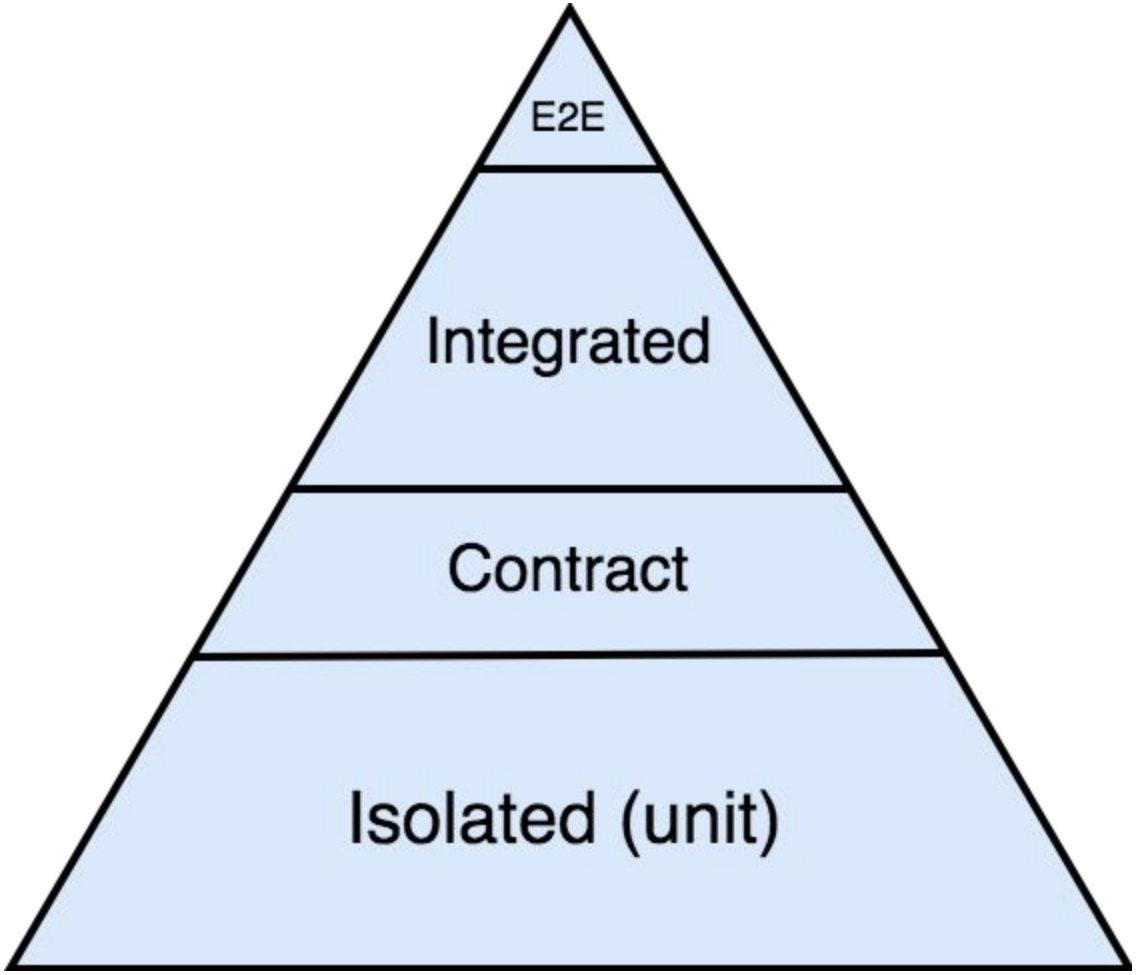
Business value

Code correctness



The Testing Pyramid of Hell




E2E
Integrated
Contract
Isolated (unit)

Speed up your releases

Do less

- Integration testing

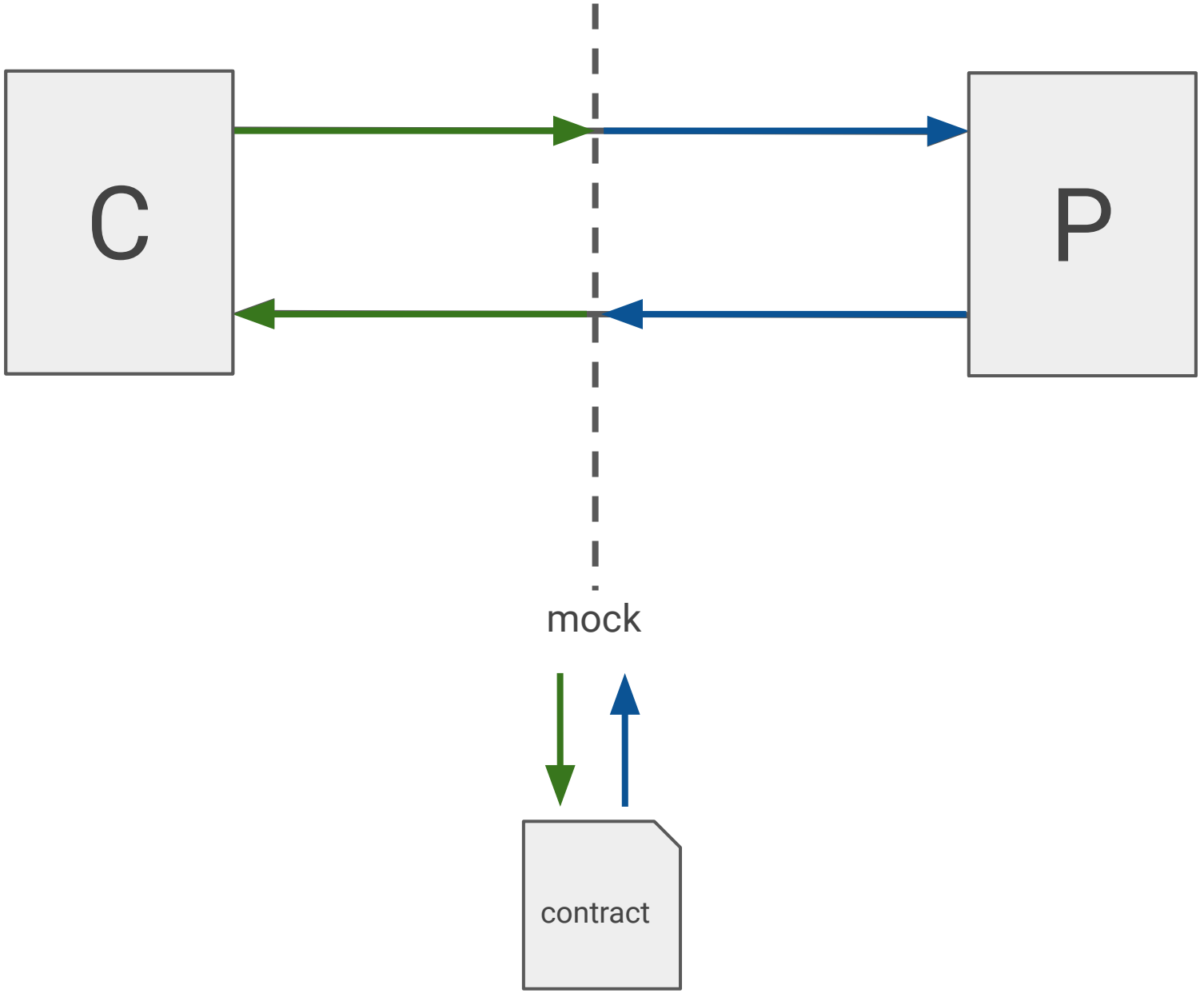
Do more

- Contract testing
- Aggregated logging
- Metrics
- Semantic monitoring
- Alerting
- Correlation IDs

My contract testing journey

pact.io

- Open source
- Multiple languages
  - JVM
  - .NET
  - Javascript
  - Python
  - Go
  - + more
- HTTP contracts
- Message contracts


C
P
mock
contract


Consumer
CI
?
Provider
CI

Too many microservices?
You need another microservice!

Pact Broker

Problem 1: Contract exchange
Solution: Pact Broker

WHEN

the provider receives

<some request>

THEN

it will return

<some response>

WHEN

the provider receives

a GET request for /alligators/Mary

THEN

it will return

a 200 OK response

with a JSON body {“name”: “Mary”}

WHEN
the provider receives
a GET request for /alligators/Mary
THEN
it will return

a 200 OK response

a 404 Not Found response

GIVEN

<the provider is in a certain state>

WHEN

the provider receives

<some request>

THEN

it will return

<some response>

Problem 2: Data setup, code coverage
Solution: Provider states

A contract testing journey

- Automate the contract exchange
- **You still need to think about test data**

Problem 3: brittle tests

Problem 3: brittle tests
Solution: flexible matching

A contract testing journey

- Automate the contract exchange
- You still need to think about test data
- Contracts should focus on the messages, not the technology
- **Contracts should be as flexible as possible - but no more**

Problem 4: Dealing with contract changes

The other service needs to know when a contract has changed

Contracts are not a substitute for good communication between teams

Pact Broker webhooks

Contracts are STILL not a substitute for good communication between teams

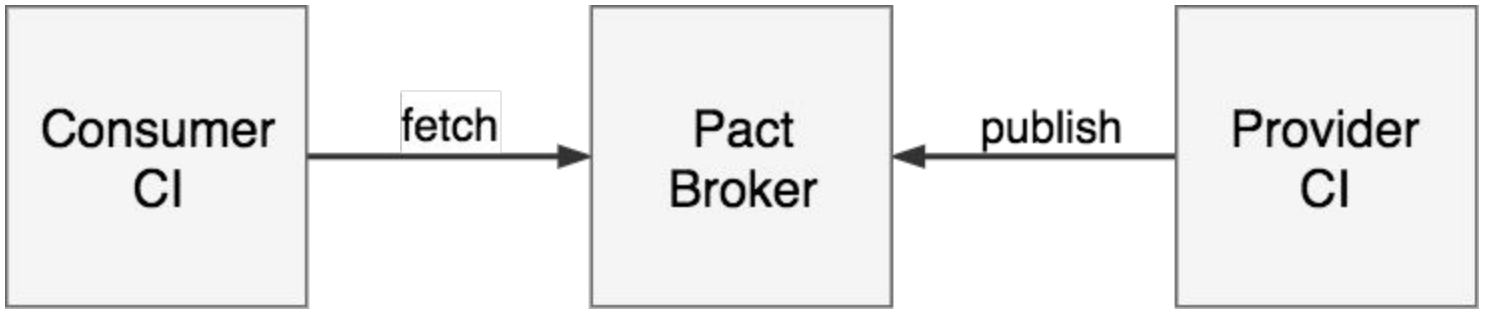
A contract testing journey

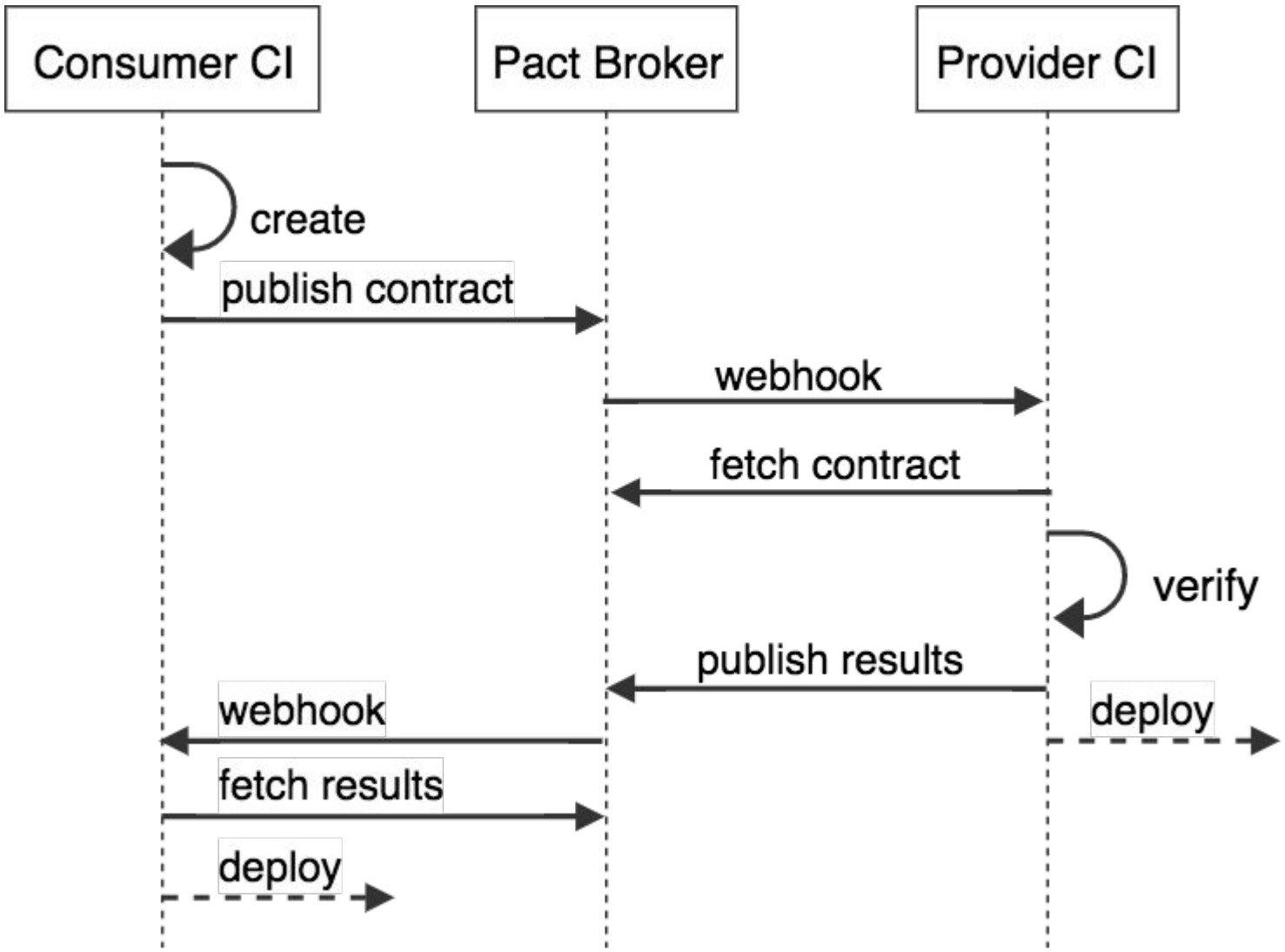
- Automate the contract exchange
- You still need to think about test data
- Contracts should focus on the messages, not the technology
- Contracts should be as flexible as possible - but no more
- **The provider needs to know when a contract has changed**
- **Remember: contracts are not a substitute for good communication between teams**

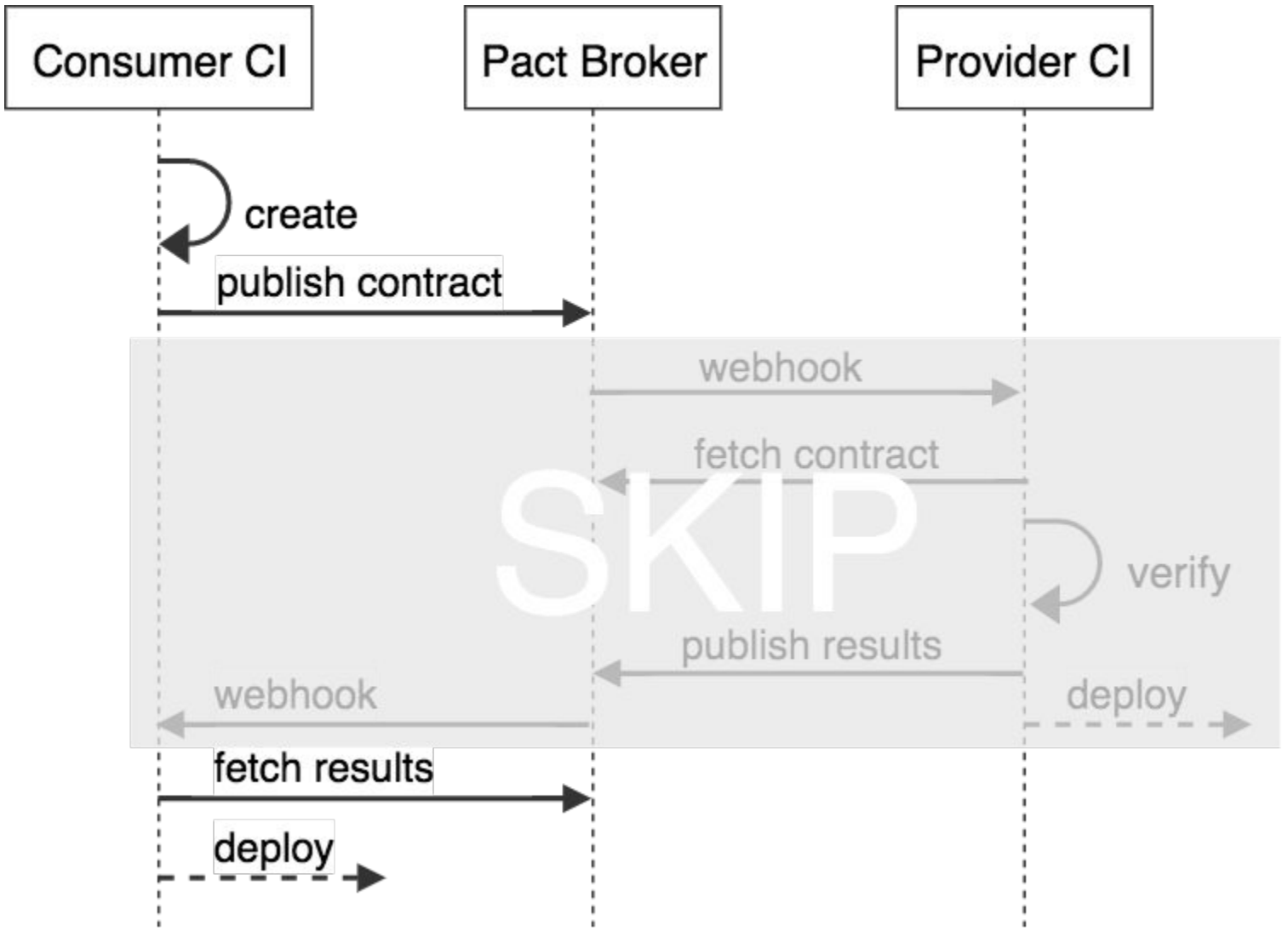

Consumer
CI
?
Provider
CI

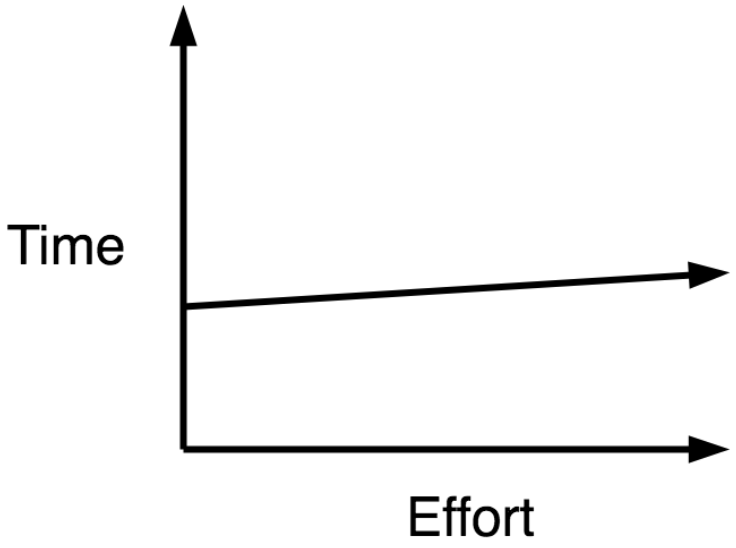
Problem 5: Communicating verification results back to consumer

Pact Broker verifications

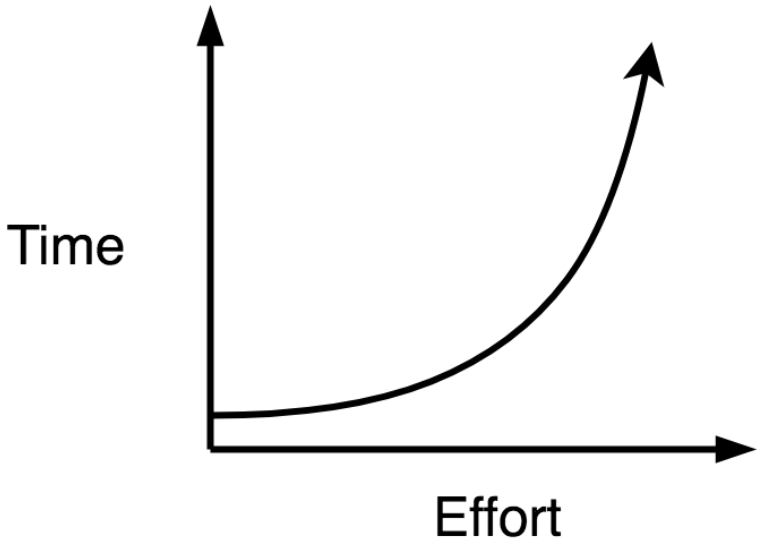



Consumer CI
Pact Broker
Provider CI
create
publish contract
webhook
fetch contract
verify
publish results
webhook
deploy
fetch results
deploy


Consumer CI
Pact Broker
Provider CI
create
publish contract
webhook
fetch contract
SKIP
verify
publish results
webhook
deploy
fetch results
deploy



Contract tests



E2E tests

If you can't deploy your services independently,
you don't have microservices.
You have a distributed monolith.

Warning!

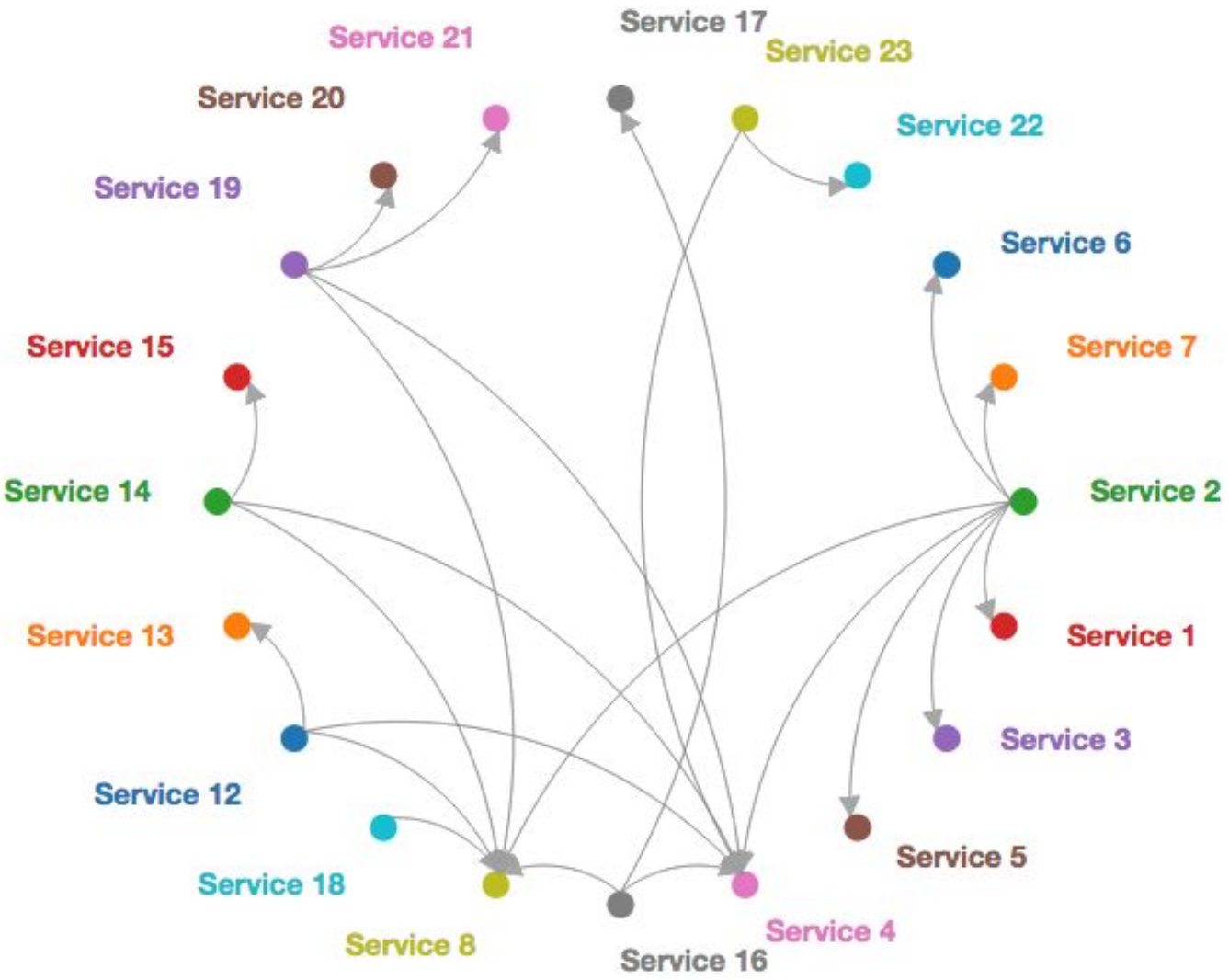
Do not build a distributed monolith

“The Matrix”

Consumer version	Provider version	Verification result
11	54	success
12	54	failure
12	55	success
13	56	success
13	57	failure

Can I deploy?

Consumer version	Provider version	Verification result
11 ✓	54 **PROD**	success
12	54	failure
12	55	success
13	56	success
13	57	unknown


Service 21
Service 17
Service 23
Service 20
Service 22
Service 19
Service 6
Service 15
Service 7
Service 14
Service 2
Service 13
Service 1
Service 3
Service 12
Service 5
Service 18
Service 4
Service 8
Service 16

Given **there is an alligator named Mary**, upon receiving **a request for an alligator** from Zoo App, with

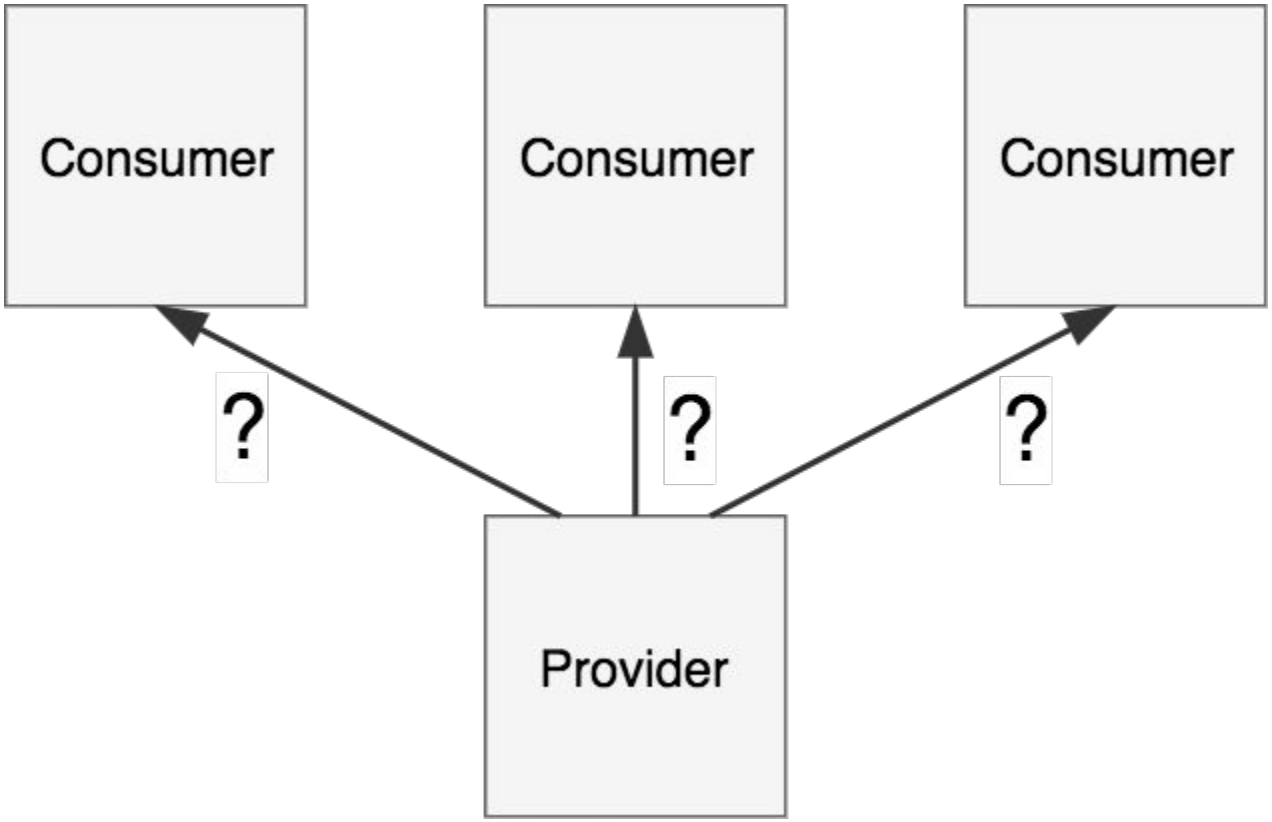
```
{
  "method": "get",
  "path": "/alligators/Mary",
  "headers": {
    "Accept": "application/json"
  }
}
```

Animal Service will respond with:

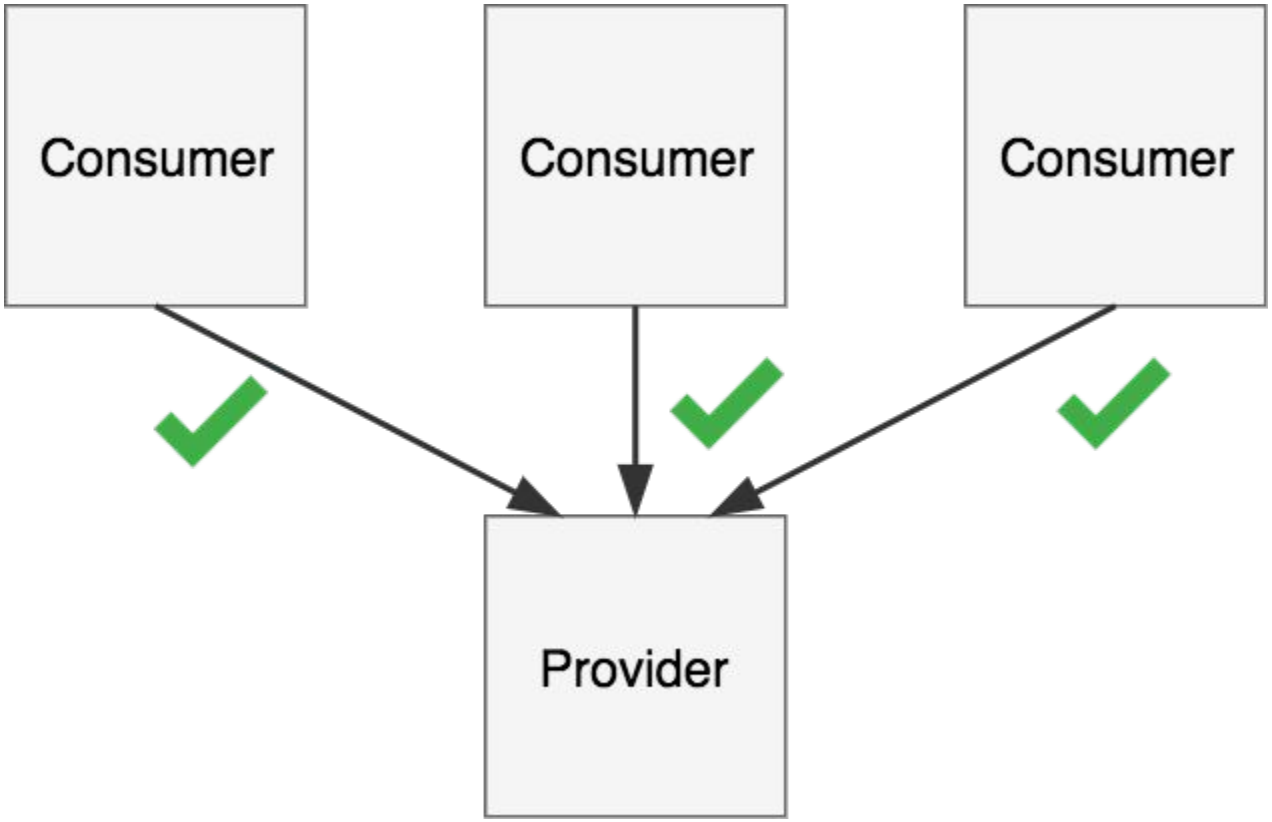
```
{
  "status": 200,
  "headers": {
    "Content-Type": "application/json;charset=utf-8"
  },
  "body": {
    "name": "Mary"
  }
}
```

What about Swagger/OAS?

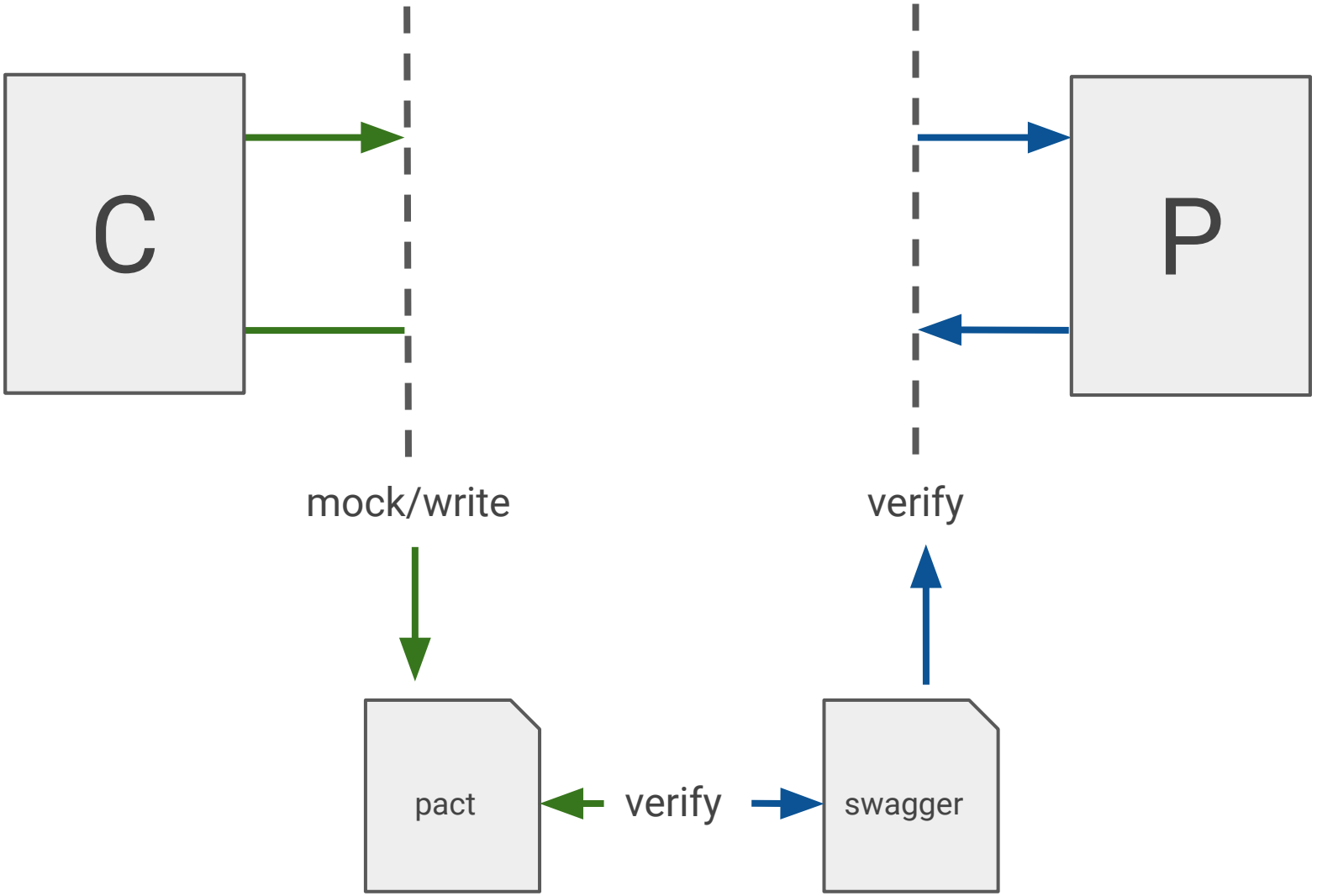
Provider contracts



Consumer contracts



Pact+Swagger



What are the next problems to solve for Pact?

- OAS support
- Improved Broker workflow

Contracts

Solved problems

- Shipping code faster

New problems

- ???

Microservices.
Test smarter, not harder

pact.io

pactflow.io

slack.pact.io

@pact_up

Beth Skurrie (Pactflow)
@bethesque

PACTFLOW

DIUS
WHERE IDEAS ARE ENGINEERED

#YOWPER