PACT

Pacticipant:

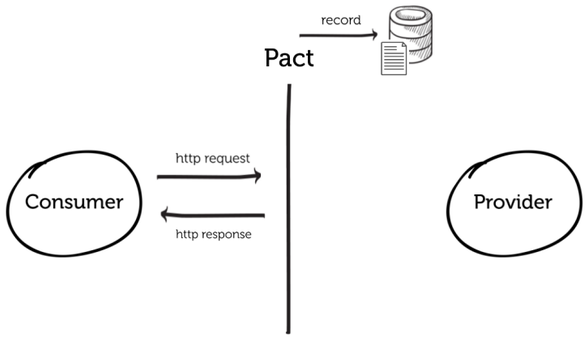
PACT FUNDAMENTALS WORKSHOP
INTRODUCTION TO CONTRACT TESTING

brought to you by DiUS

HOW PACT WORKS

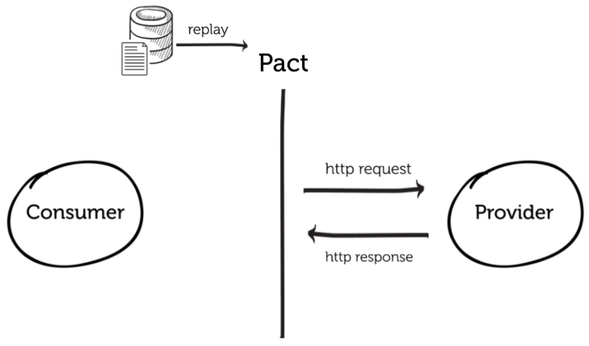
DEVELOPMENT PROCESS

Step 1: define consumer expectations




```
Given “User A exists”
When I Receive “a GET request for user A”
With “these headers and query”
Respond with “200 OK”
And “User A' details in the body”
```

Step 2: verify expectations on provider



KEY CONCEPTS

Service Consumer
A component that initiates a HTTP request to another component (the service Provider). Note that this does not depend on the way the data flows - whether it is a GET or a PUT / POST / PATCH, the Consumer is the initiator of the HTTP request.

Service Provider
A server that responds to an HTTP request from another component (the service consumer).

Mock Service Provider
Used by tests in the Consumer project to mock out the actual service Provider, meaning that integration-like tests can be run without requiring the actual service Provider to be available.

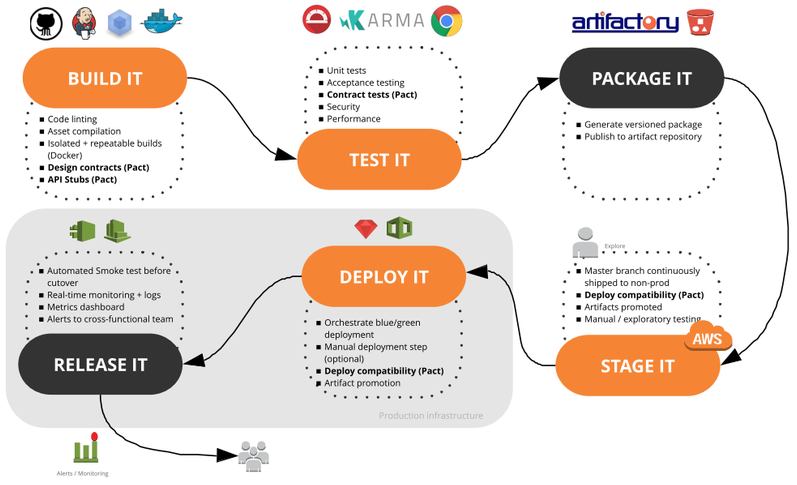
Pact file
A file containing the JSON serialised requests and responses that were defined in the Consumer tests. This is the Contract.

Pact verification
To verify a Pact, the requests contained in a Pact file are replayed against the Provider code, and the responses returned are checked to ensure they match those expected in the Pact file.

Provider state
A name describing a “state” (like a fixture) that the Provider should be in when a given request is replayed against it e.g. “when user John Doe exists” or “when user John Doe has a bank account”.

A Provider state name is specified when writing the Consumer specs, then, when the pact verification is set up in the Provider the same name will be used to identify the set up code block that should be run before the request is executed.

CONTINUOUS DELIVERY



NOTES

WORKSHOP

NOTES

PREREQUISITES

GET STARTED!

Need a VM?

```
https://s3-ap-southeast-2.amazonaws.com/pact-workshop-dius/PactWorkshop32.zip
```

In your terminal

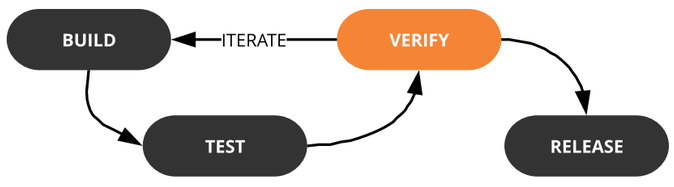
```
git clone  https://github.com/DiUS/pact-workshop-js.git
cd pact-workshop-js
npm i
git checkout --track origin/step1
```

STEPS

1. Create consumer {C}
2. Create unit tests - integration fail! {C}
3. Pacts to the rescue! {C}
4. Verify the provider
5. Provider verification failed!
6. Fix the client's expectations {C}
7. Verify provider - again
8. Test for missing query parameter {C}
9. Verify Provider
10. Handle invalid query
11. Provider states {C}
12. Handle provider states on Provider
13. Collaborating on contracts with the Pact Broker: Deploy with confidence!

TIP: Reset your workspace

```
git reset HEAD --hard
git checkout --track origin/step1
```



TIP: Test with your own Broker

Update `./consumer/test/publish.js` with credentials sent in the pre-event email

YOUR ORGANISATION

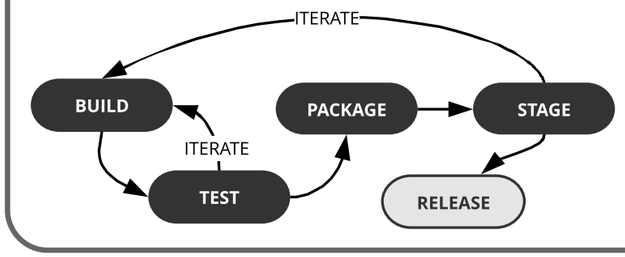
PACT READINESS QUESTIONNAIRE

1. What are the biggest challenges to continuous delivery in your project\organisation?

2. What are the biggest challenges to introducing Pact into your organisation?

BUILD/DEPLOY PIPELINE

Draw your CD process: What steps do you take to get from commit to production? Are there manual steps? Do you use GitHub flow or a custom process to promote code to production?



TIP: Identifying stages in the process

Identify common elements and group under a header such as "test". Each stage may have many steps.

YOUR PROJECT

ARCHITECTURE

Draw your architecture : Include all collaborating components, what language they are in, who manages them. What are the *contracts* you are most interested in?

Key

Annotate your architecture with to identify your technology mix.

J - JavaScript
JVM - Java\JVM
R - Ruby
N - .NET
G - Golang
P - Python
S - Swift / Objective-C
PHP - PHP
O - Other