Contract Testing

Toni Masotti
masotti@bikeleasing.de
Bikeleasing-Service

November 25, 2023

Outline

1. Problem statement
   - Company growth
   - Test Typology
2. Contract Testing
   - Pact
3. Demo Time
4. Worth thinking about

Let's start

It all started with Bob...

...and it continues with Bob

It all started with Bob…

They do work on my local machine though 😢

Bob

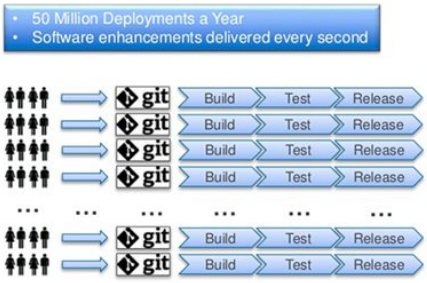
BOB ! the changes you made do not work on the live system 😡

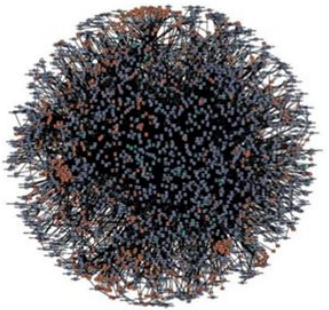
Angry chef

Bob has learnt the lesson...

MICROSERVICES
MICROSERVICES EVERYWHERE

Extreme examples...





Everybody loves microservices

- Big promise: Independent deployability

Everybody loves microservices

- Big promise: Independent deployability
- Fast feedback loops

Everybody loves microservices

- Big promise: Independent deployability
- Fast feedback loops
- Strong team focus, ideally crossfunctional

Everybody loves microservices

- Big promise: Independent deployability
- Fast feedback loops
- Strong team focus, ideally crossfunctional
- Loosely coupled architecture

Everybody loves microservices

- Big promise: Independent deployability
- Fast feedback loops
- Strong team focus, ideally crossfunctional
- Loosely coupled architecture
- Fast onboarding

Everybody loves microservices

- Big promise: Independent deployability
- Fast feedback loops
- Strong team focus, ideally crossfunctional
- Loosely coupled architecture
- Fast onboarding
- Cost-efficiency on the long run

Everybody loves microservices

- Big promise: Independent deployability
- Fast feedback loops
- Strong team focus, ideally crossfunctional
- Loosely coupled architecture
- Fast onboarding
- Cost-efficiency on the long run
- Cloud native

Everybody loves microservices

- Big promise: Independent deployability
- Fast feedback loops
- Strong team focus, ideally crossfunctional
- Loosely coupled architecture
- Fast onboarding
- Cost-efficiency on the long run
- Cloud native
- Tech-agnostic approach

Everybody loves microservices

- Big promise: Independent deployability
- Fast feedback loops
- Strong team focus, ideally crossfunctional
- Loosely coupled architecture
- Fast onboarding
- Cost-efficiency on the long run
- Cloud native
- Tech-agnostic approach
- Scalability out of the box

Everybody loves microservices

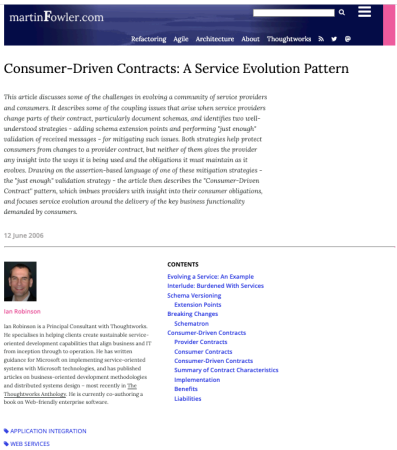
- Big promise: Independent deployability
- Fast feedback loops
- Strong team focus, ideally crossfunctional
- Loosely coupled architecture
- Fast onboarding
- Cost-efficiency on the long run
- Cloud native
- Tech-agnostic approach
- Scalability out of the box
- Improved Security (no SPOF, no SPOC)

Everybody loves microservices

- Big promise: Independent deployability
- Fast feedback loops
- Strong team focus, ideally crossfunctional
- Loosely coupled architecture
- Fast onboarding
- Cost-efficiency on the long run
- Cloud native
- Tech-agnostic approach
- Scalability out of the box
- Improved Security (no SPOF, no SPOC)
- Optimized Time to Market

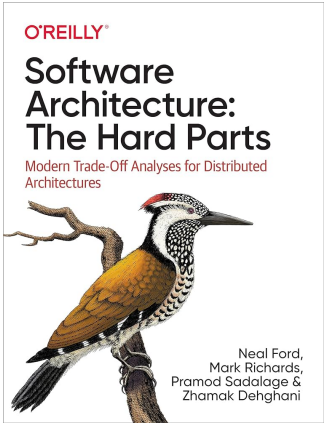
Everybody loves microservices

> *Chief among the benefits of service-enabling an enterprise's application landscape are* ***increased organizational agility*** *and* ***reduced overall cost*** *of implementing change.*
>
> *A SOA increases organizational agility by placing high-value business functions in* ***discrete, reusable services****, and then connecting and orchestrating these services to satisfy core business processes.*
>
> *It reduces the cost of change by* ***reducing the dependencies*** *between services, allowing them* ***to be rapidly recomposed and tuned*** *in response to change or unplanned events.*



Is always a tradeoff

> *There is only one hard rule in software development:* ***Everything is a tradeoff****.*
>
> *– Neal Ford*



1 Problem statement
- Company growth
- Test Typology

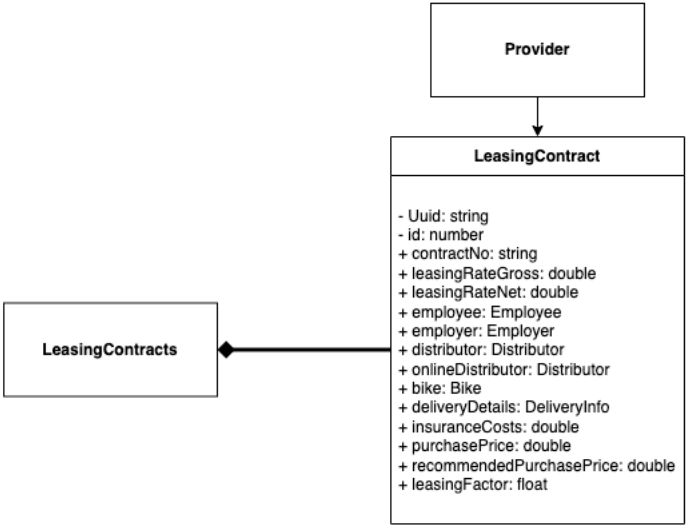
2 Contract Testing
- Pact

3 Demo Time

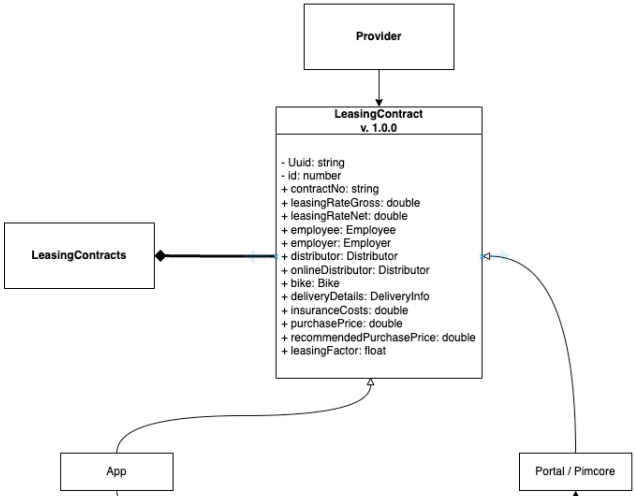
4 Worth thinking about

Bob's company is being very successfull...

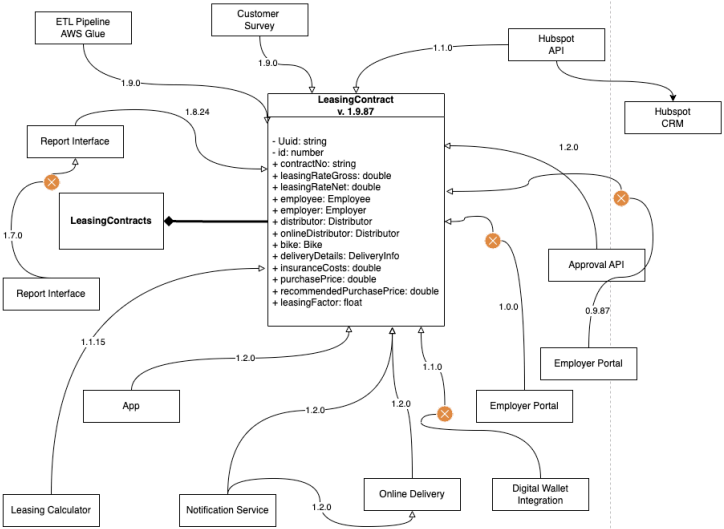
A company growth... so its architecture



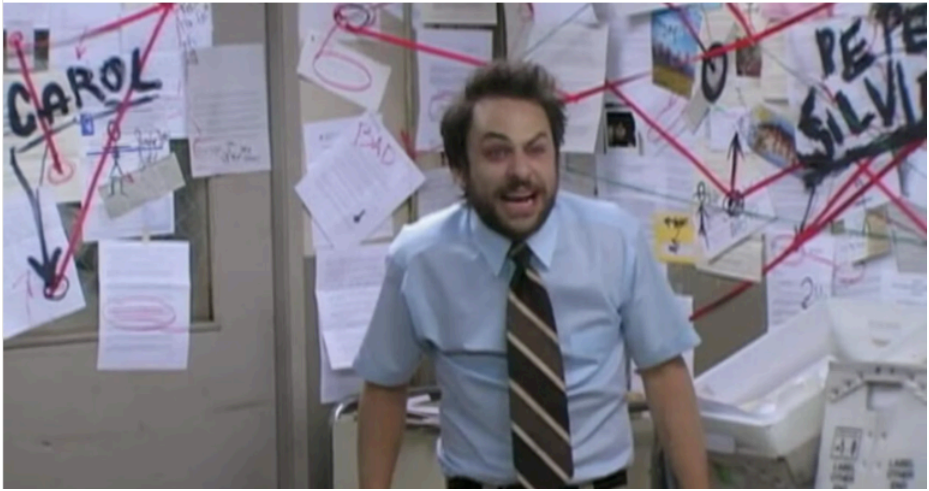
A company growth... so its architecture



A company growth... so its architecture



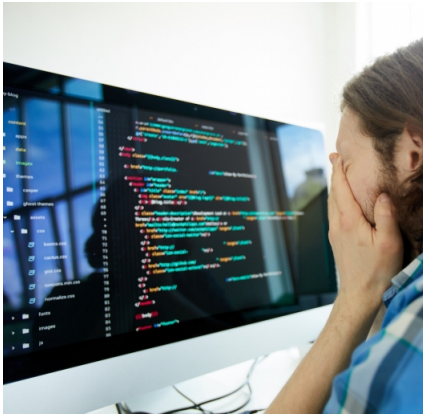
Bob is happy...



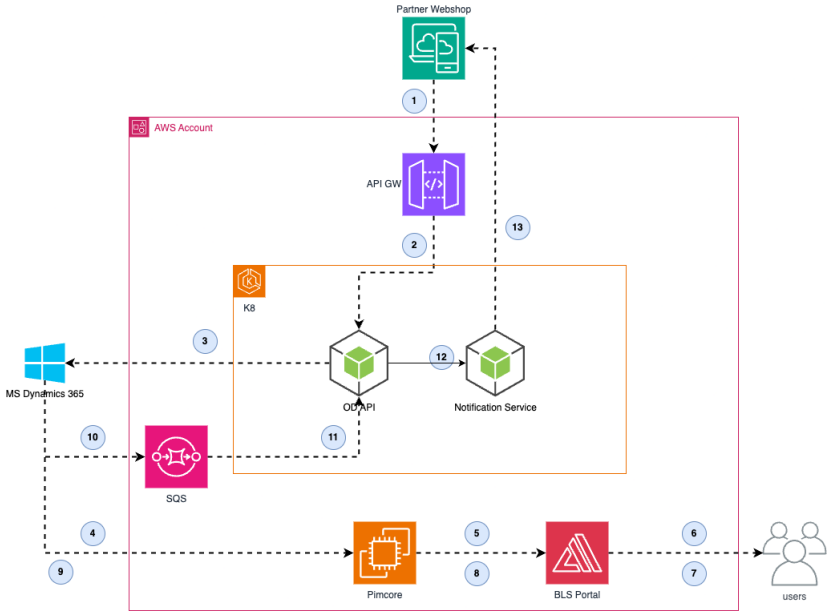
The Problem...

> *Every time we touch one service, all the others break :(*

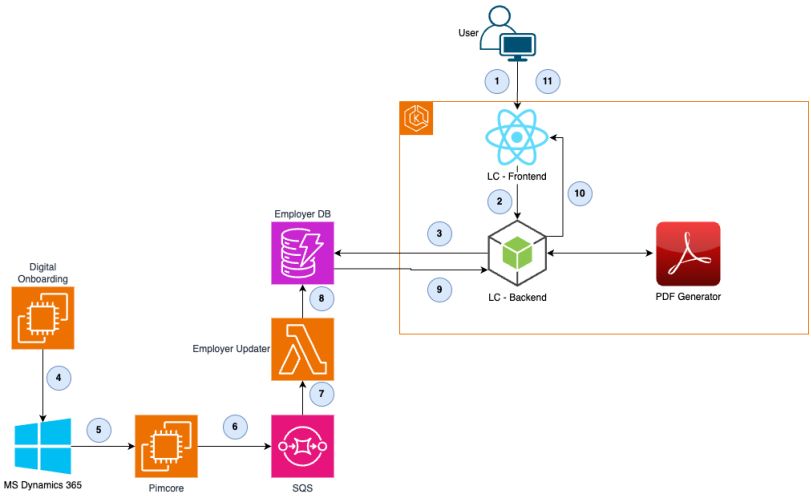
- Dependency hell
- Devs don't have confidence in deploying changes
  - Consequences of changes are hard to track
  - Distributed architectures are hard to test



Architecture example: Online Delivery



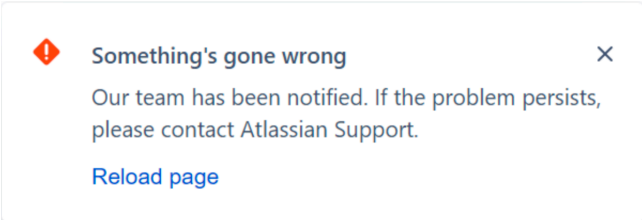
Architecture example: Leasing Calculator



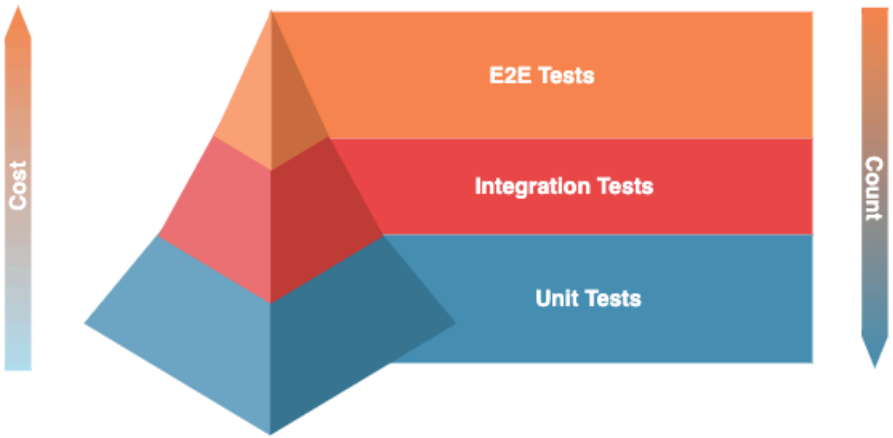
Updating a microservice: the sad Atlassian case

```
{
  "users": "Mike",
  "address": "Something"
}
```

```
{
  "user": "Mike",
  "address": "Something"
}
```



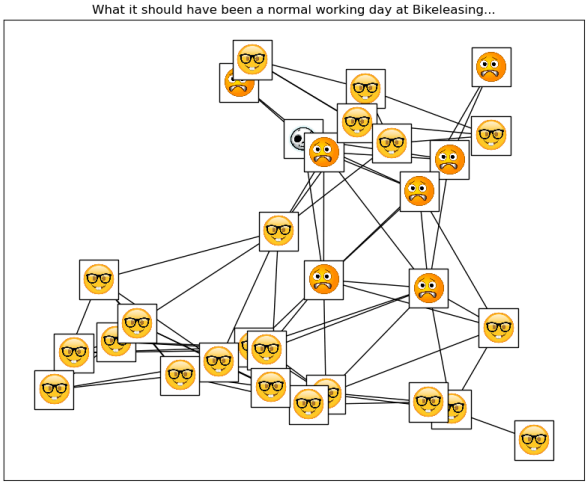
Test Typology



The problem with Unit Tests alone

- Local changes, even if tested, do not assure that the system as whole is working
- Mocks are not guaranteed to really represent the other part of the system

Embedded Animation



Problem with E2E Tests

They're very important, **but**...

- In large architectures not reproduceable locally
- Hard or impossible to test some scenarios (e.g. system not available)
- Expensive (timely and money)
- In some cases need for a separate CI Environment
  - Expensive to provision and maintain
  - Not always possible to reproduce the same environment
  - Need for tweaks (custom endpoints, prepared databases)

Distributed $\neq$ Decoupled

Is this one or three pipelines?

1 Problem statement
- Company growth
- Test Typology

2 Contract Testing
- Pact

3 Demo Time

4 Worth thinking about

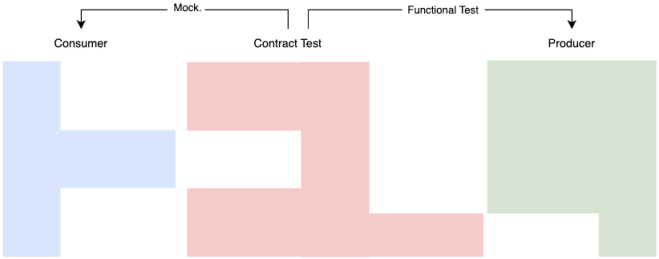
Contracts

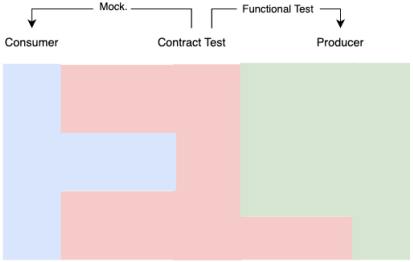
Contracts

A formal agreement between parties or individual. Synonyms: agreement, protocol, deal

— Oxford English Dictionary

Contract Test Role

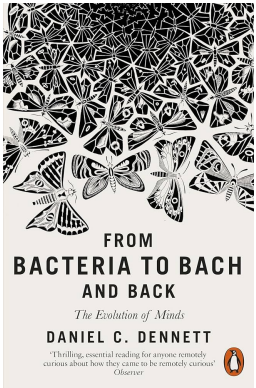


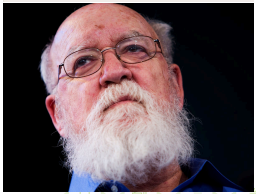


Pact - Consumer Driven Contracts

"'*A strange inversion of reasoning...*"'

- Consumer Driven Contracts
- Pact is a contract testing tool
- Pact is a specification for consumer driven contracts
- Pact is a set of frameworks that support consumer driven contracts





1 Problem statement
- Company growth
- Test Typology

2 Contract Testing
- Pact

3 Demo Time

4 Worth thinking about

Hintergrund

- ja, something

1 Problem statement
- Company growth
- Test Typology

2 Contract Testing
- Pact

3 Demo Time

4 Worth thinking about

Hintergrund

- ja, something