WiX Engineering

Decomposing distributed monolith with Node.js

Vilius, Guild Master

 viliusl@wix.com |  @viliusl |  linkedin/viliusl |  github.com/viliusl

WiX Engineering

Decomposing distributed monolith with ~~Node.js~~

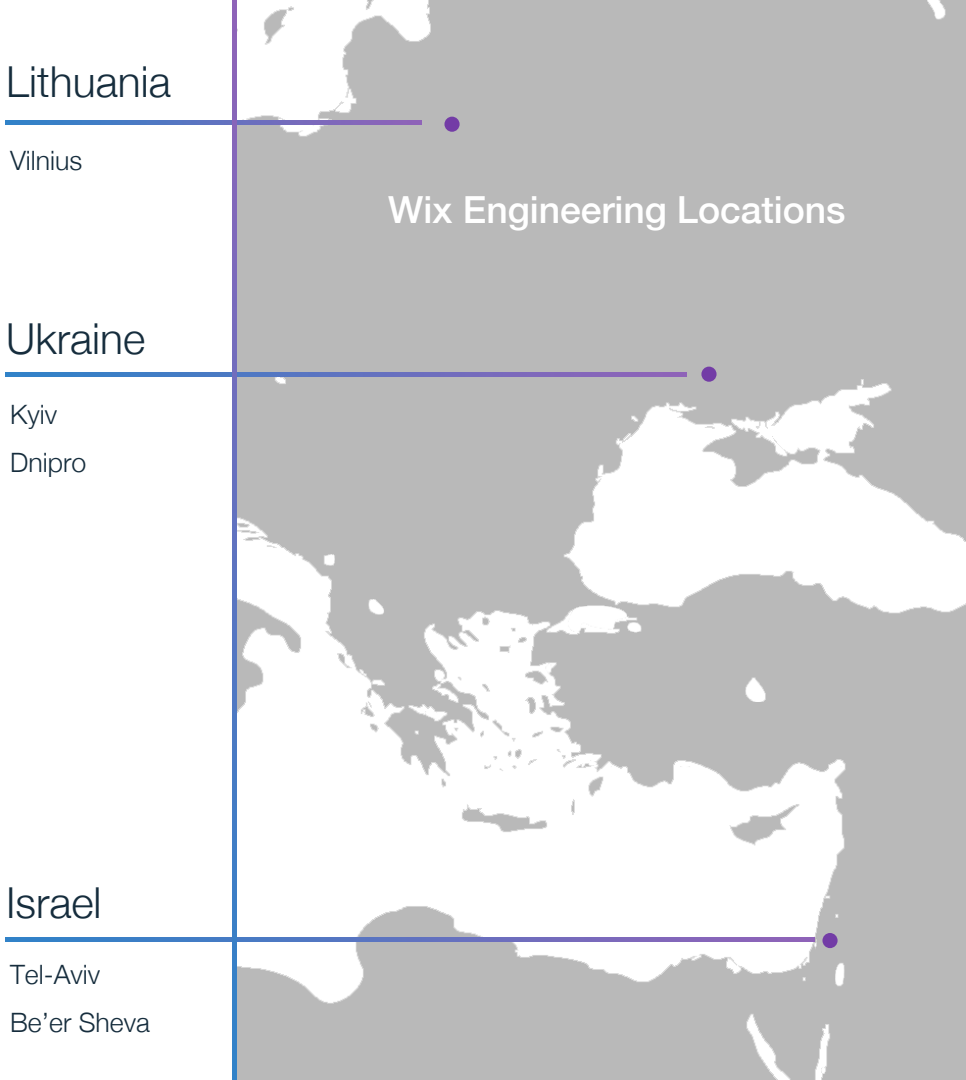
Stack of your choice

Vilius, Guild Master

viliusl@wix.com | @viliusl | linkedin/viliusl | github.com/viliusl

Hi.

I am Vilius



Hi.

I am Vilius

Lithuania

Vilnius

AGENDA

Microservices and Distributed Monolith (DM)

Why am I talking about DM in first place?

Short history and joys of adding 2nd stack

Decomposing Distributed Monolith

“Microservice architectural style is an approach to developing a single application as a suite of small services, each running in its own process and communicating with lightweight mechanisms, often an HTTP resource API”

Martin Fowler

01

What is a distributed monolith

A form of a binary coupling, when in order to interact and coexist in a given system you are **required** to use a set of “official” libraries.

Client library becomes “Only” official way to access the service

What’s wrong with it?

Nearly impossible to adopt new architectures, languages, etc.

But hey, DRY, code reuse - it’s good right?

“The evils of too much coupling between services are far worse than the problems caused by code duplication”

Sam Newman, Building Microservices

You potentially lose

- polyglot;
- organizational/technical decoupling;
- temporal decoupling;

Ok, so what is an alternative?

Contracts and protocols!

Yeah, but central logging, distributed tracing, context passing….

You don’t need binary coupling

- Standardization via protocols and contracts and independent libraries
- Compliancy/contract tests

02

Do you suffer from Distributed Monolithitis?

To know, you have to answer to 2 following questions honestly

Does it take months to upgrade a library across company?

Does it take months to upgrade a library across company?

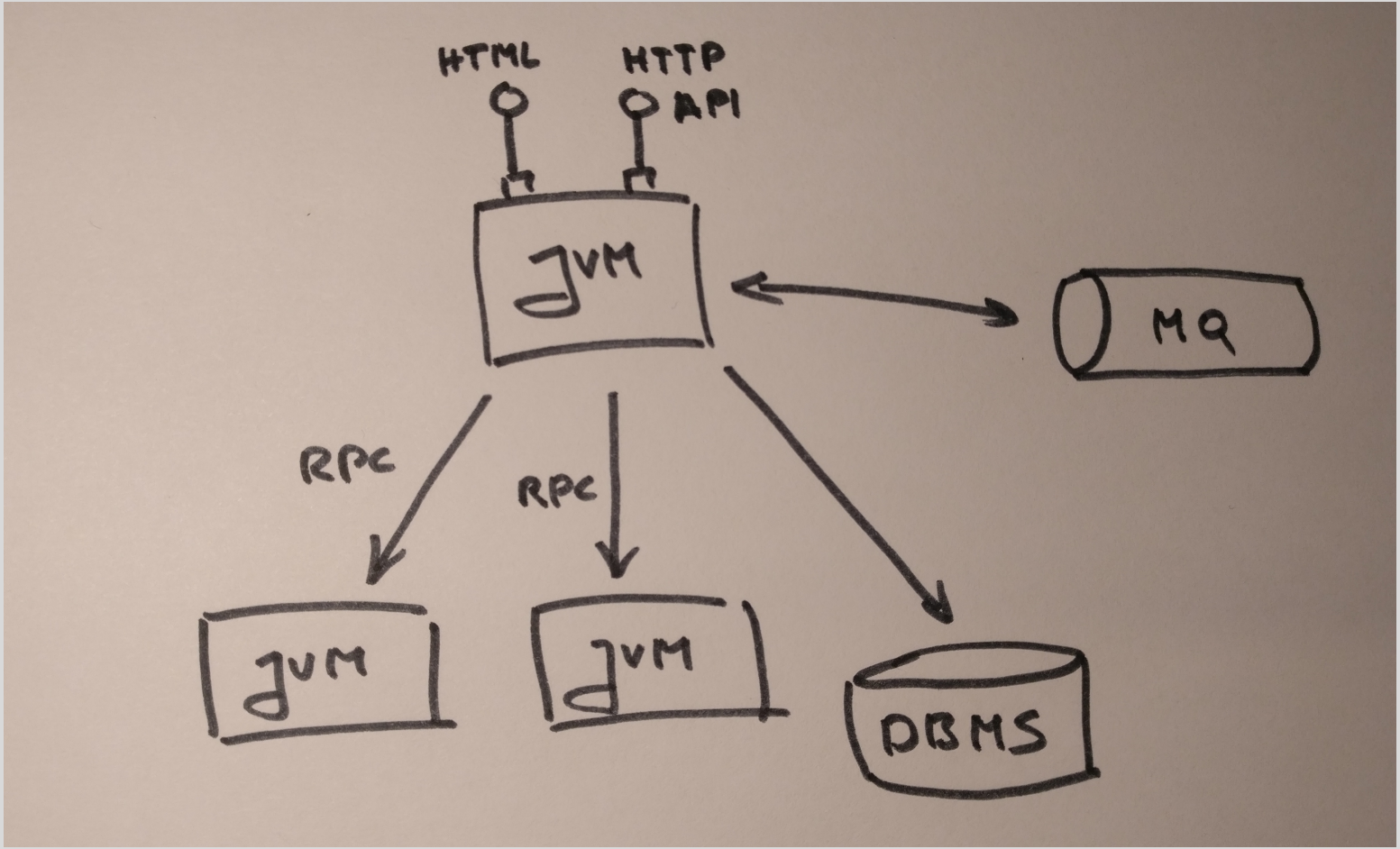
Does it take ~ a year to introduce a new stack?

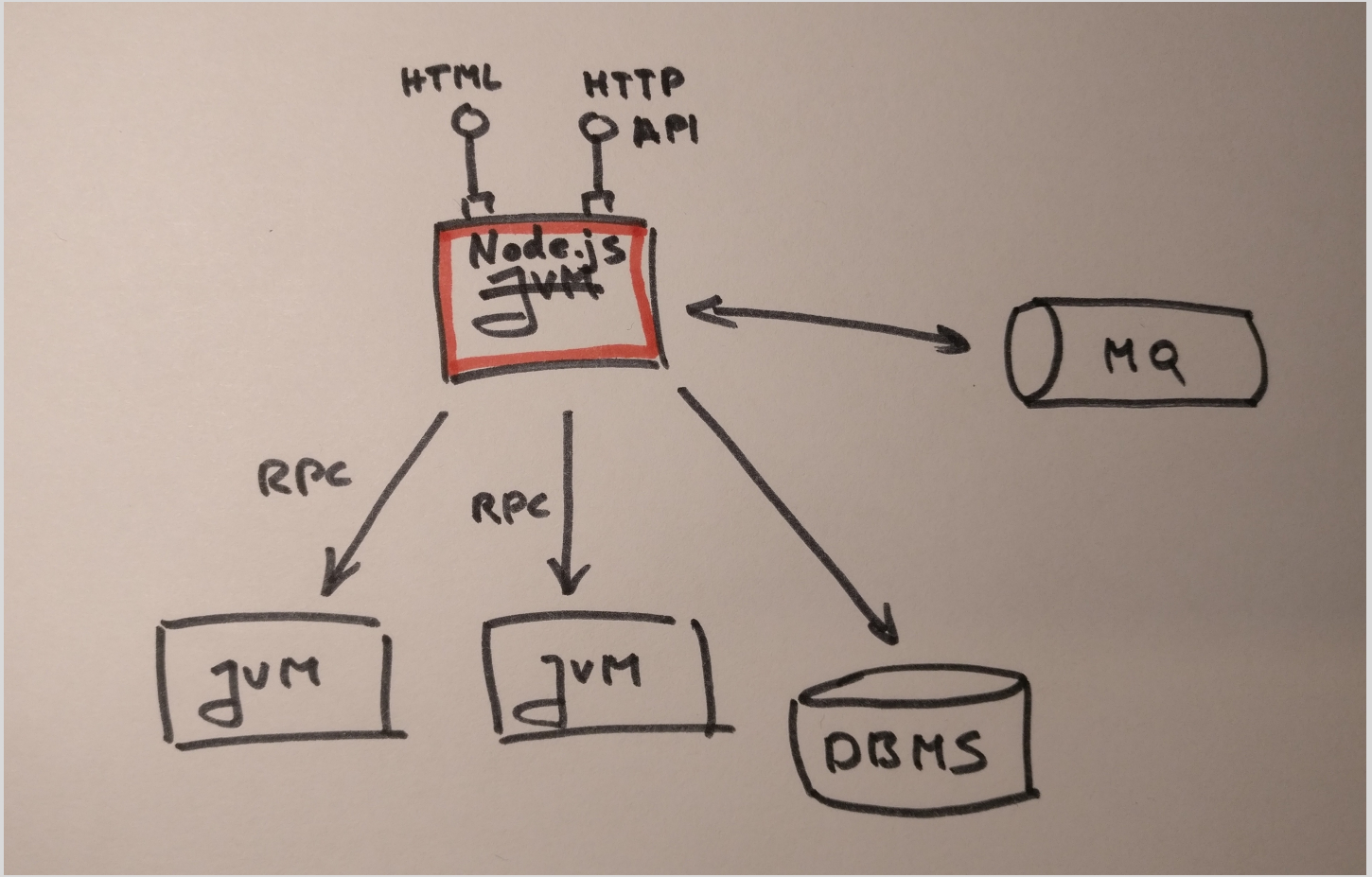
03

Introducing a Node.js stack @ Wix

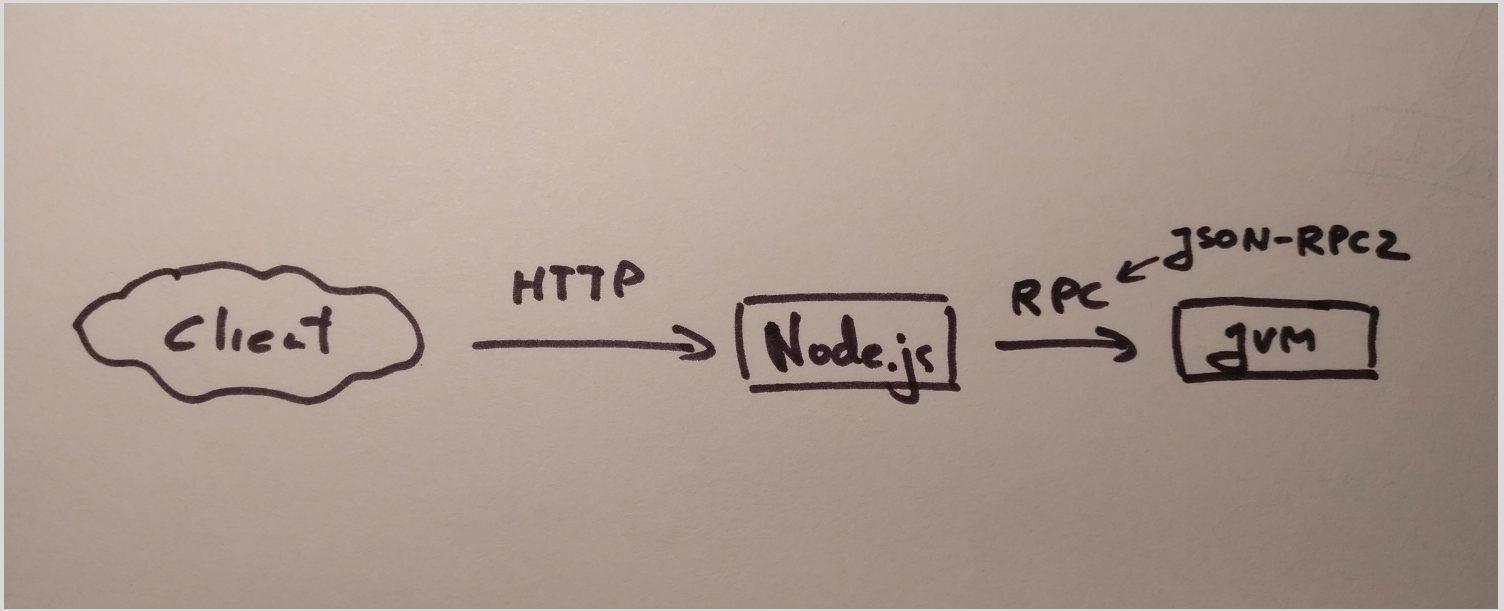
“Target of this project is to enable coding front-end servers in Wix using Node.js. We are aiming at the pattern of front-end servers and service-servers – a pattern that is used in different companies”

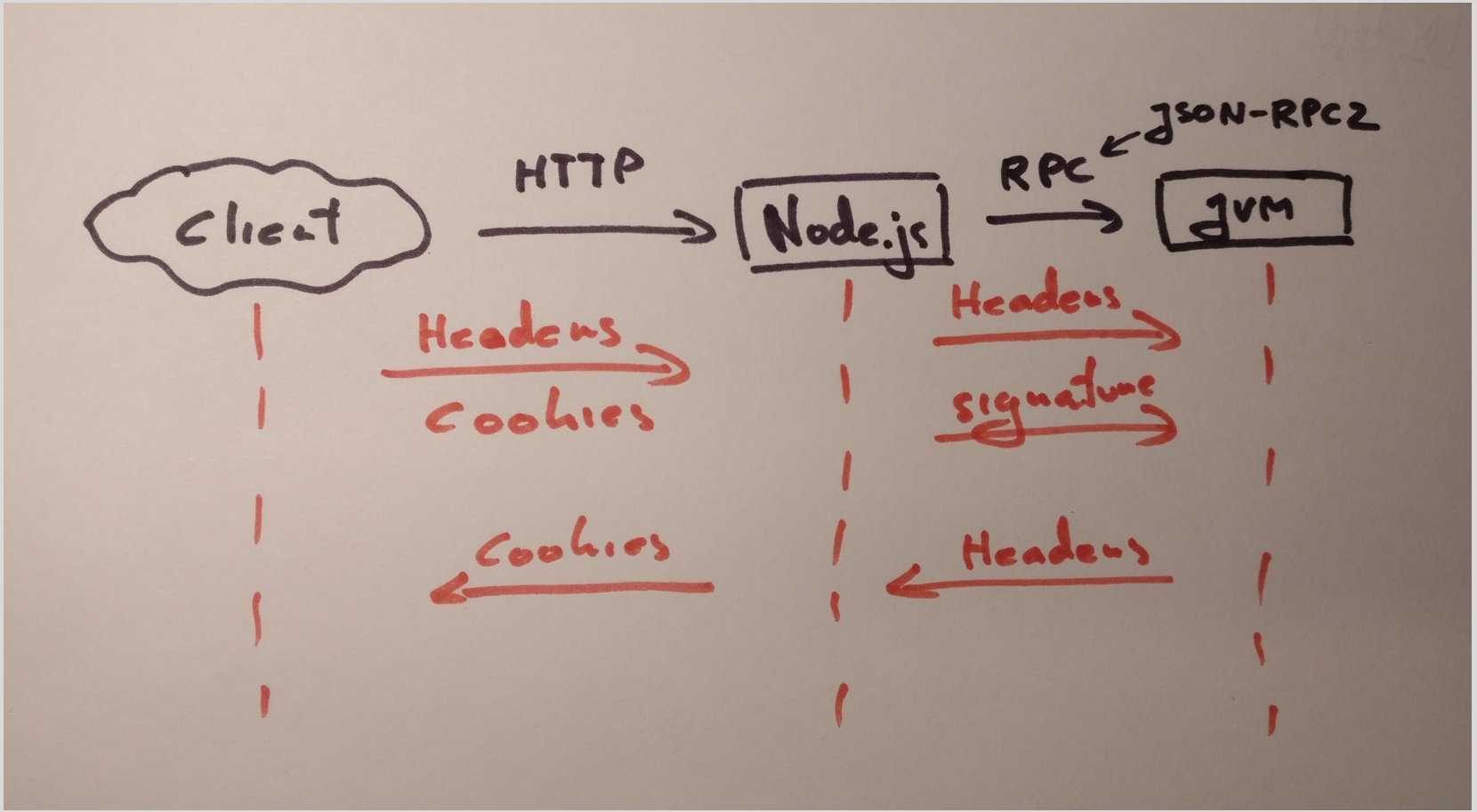
Yoav Abrahami


HTML
HTTP
API
JVM
MQ
RPC
RPC
JVM
JVM
DBMS

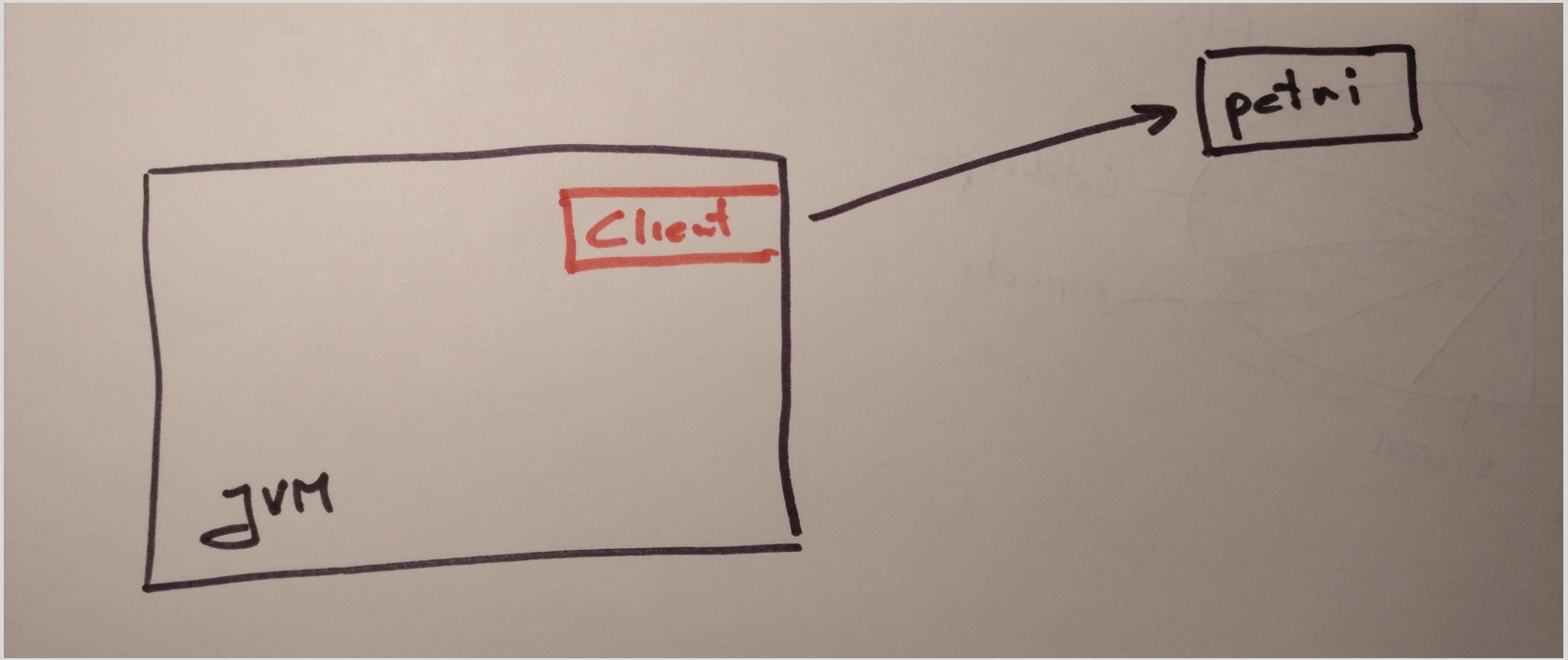

HTML
HTTP API
Node.js
JVM
MQ
RPC
RPC
JVM
JVM
DBMS

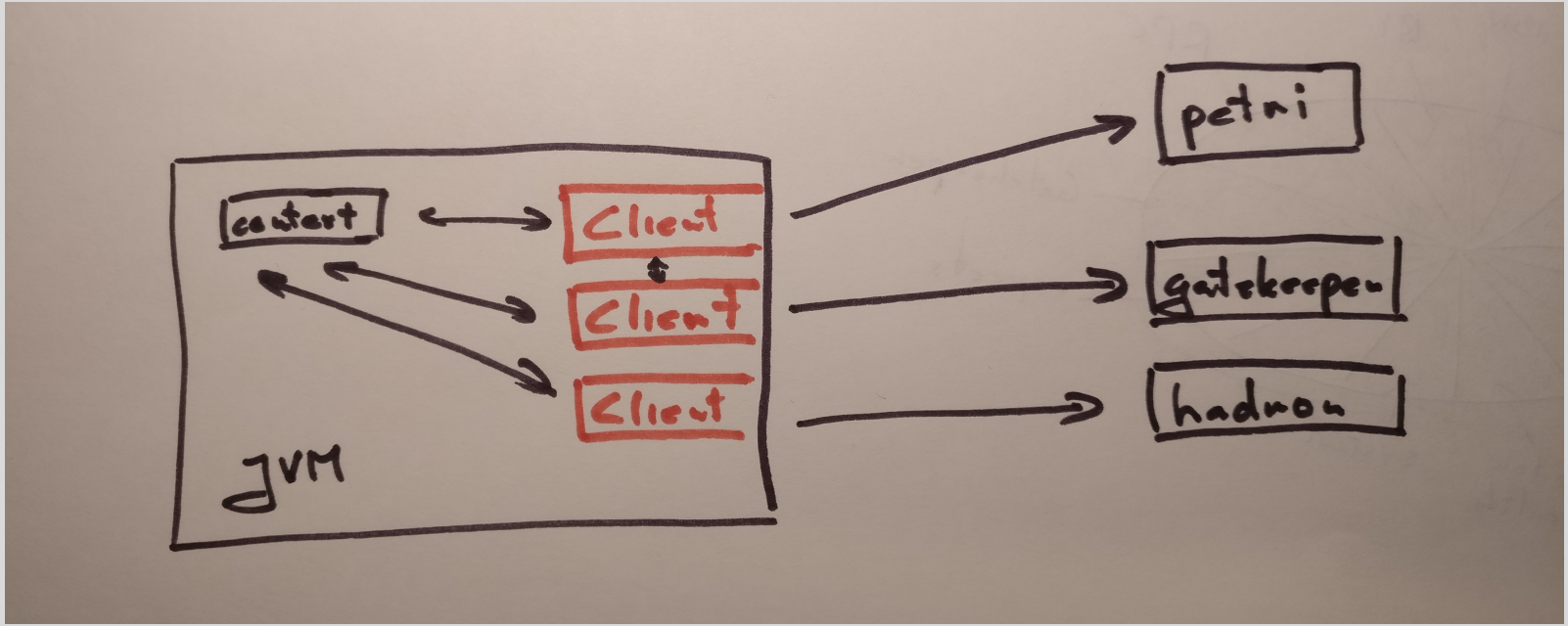
Does not look too hard right?


Client
HTTP
Node.js
RPC
JSON-RPC2
JVM

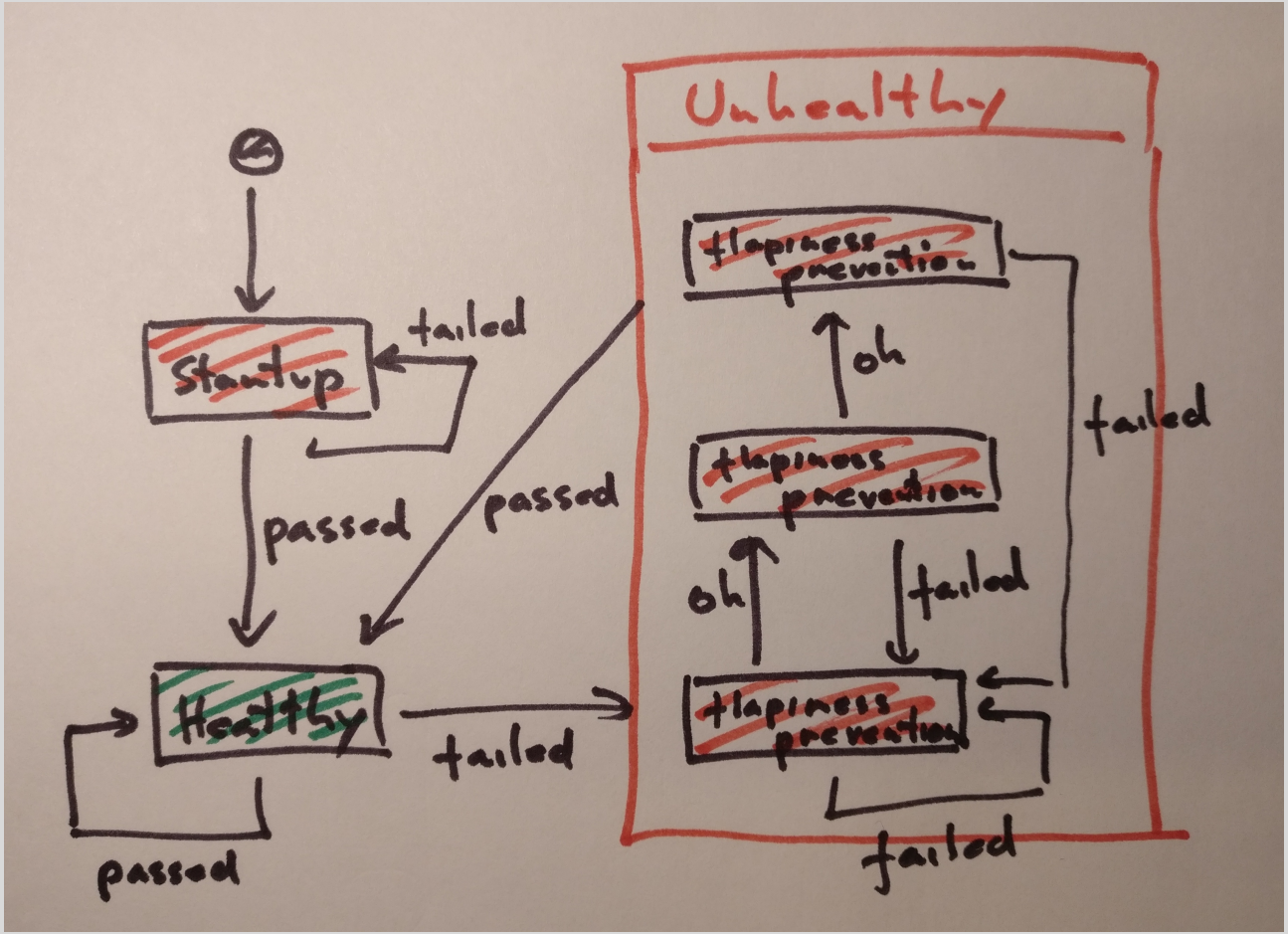

Client
HTTP
Node.js
RPC
JSON-RPC2
JVM
Headers
Cookies
Headers
signature
Cookies
Headers

But wait, there was more…


petni
Client
JVM


petri
context
Client
Client
Client
gatekeeper
hadron
JVM

Oh, and ops contracts…


Unhealthy
Startup
failed
passed
passed
Healthy
failed
passed
flapiness prevention
ok
flapiness prevention
failed
ok
failed
flapiness prevention
failed
failed

But not was gloom and doom

Build/Deploy pipeline already had support/extensions for non-official stack

Contract for deploying application was defined, sane, simple

Contract for configuration was clear and defined

Underlying protocols were standardized, business services as proper microservices

04

Decomposing Distributed Monolith

There were never a goal to decompose, fix, re-architect existing stack

The goal was to introduce new one to be leaner, be faster, make us go faster

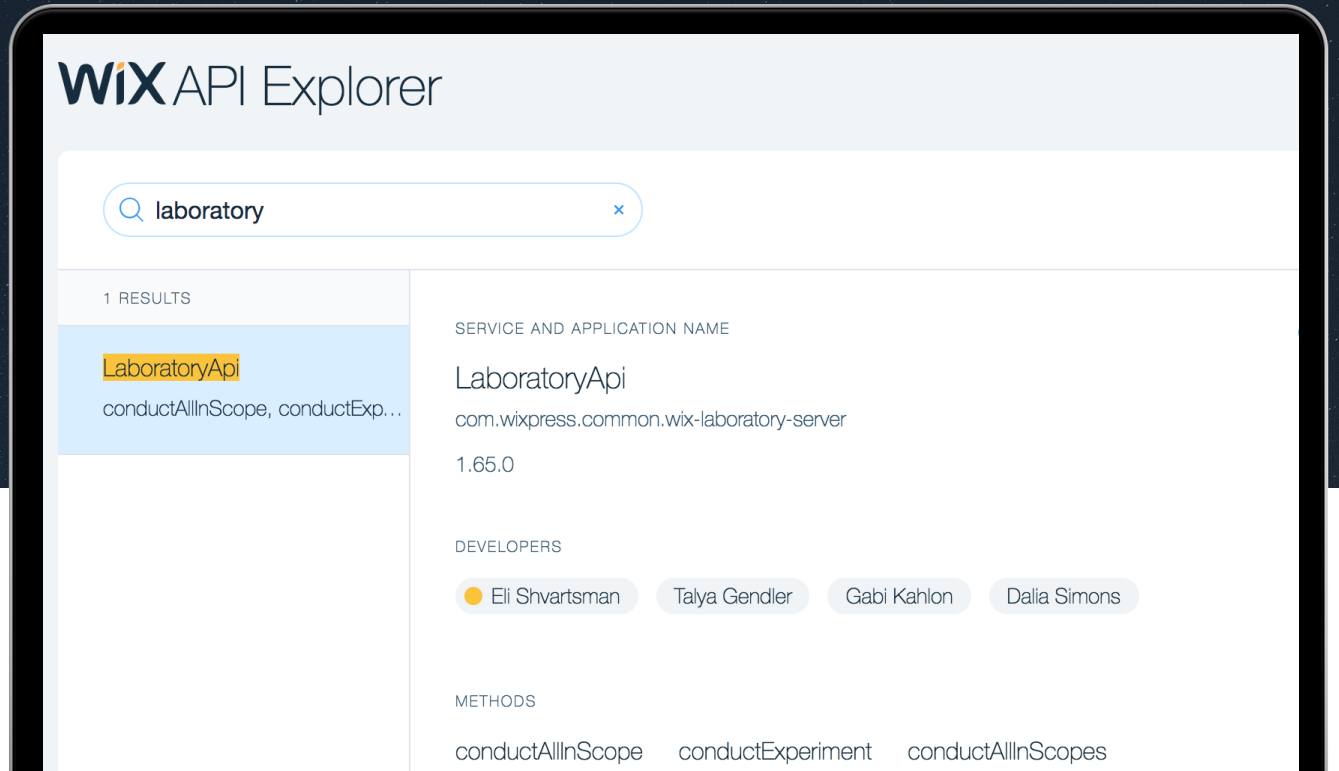
Decomposition is just a side-effect if you want to actually be leaner, go faster

Step #1 – POC

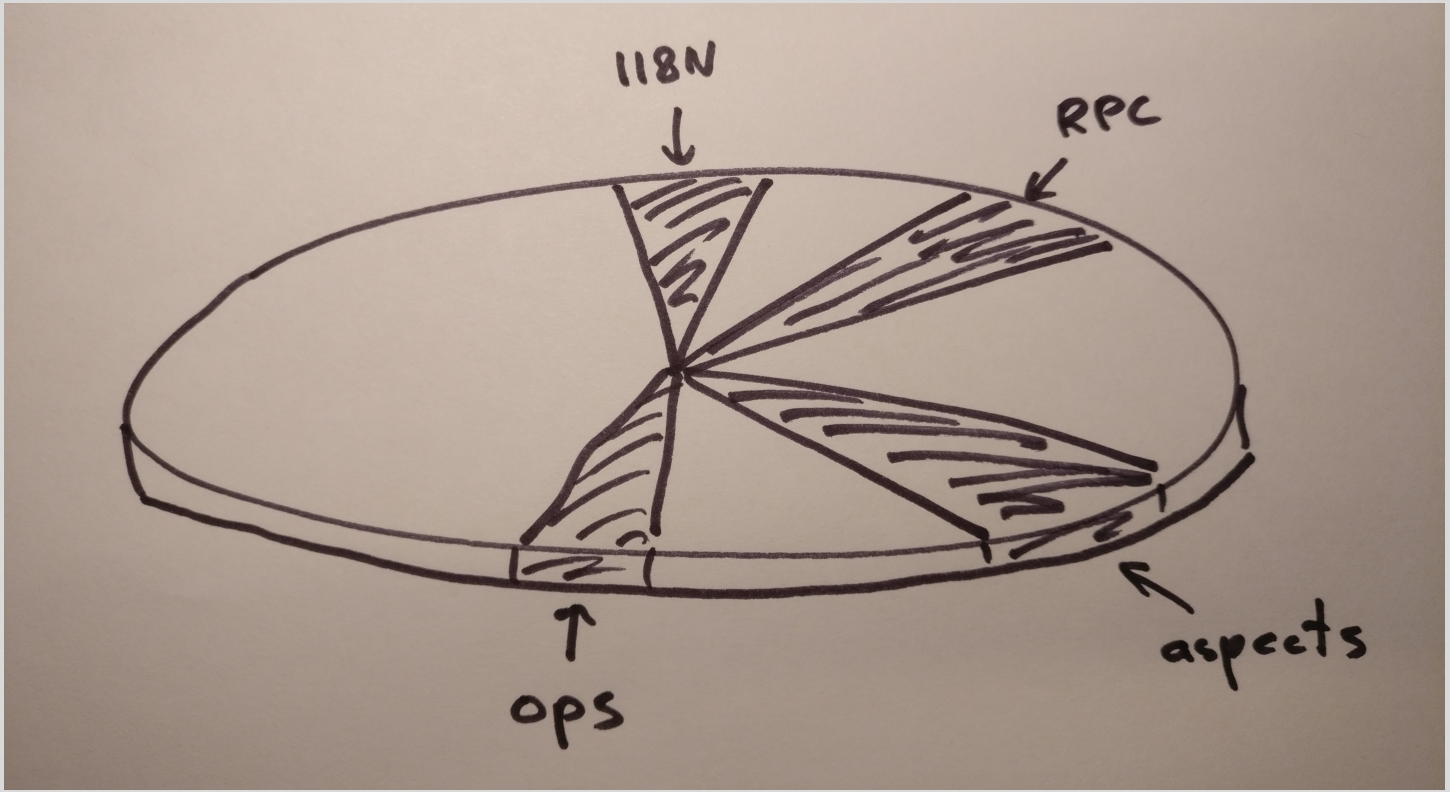
Some TODOs

```
// todo - talk to Vilius about that
var url = req => req.protocol + '://' + req.get('host') +
req.originalUrl;


// todo - add petri enricher
return new WixRpcClientSupport(
  reqContext.get(wixRequestContext),
  rpcSigner.get(options.rpcSigningKey),
  wixSessionEnricher.get(wixSession),
  biEnricher.get(wixBi)
);


module.exports.addTo = app => {
  app.get('/health/is_alive', (req, res) => res.send('Alive'));
};
```

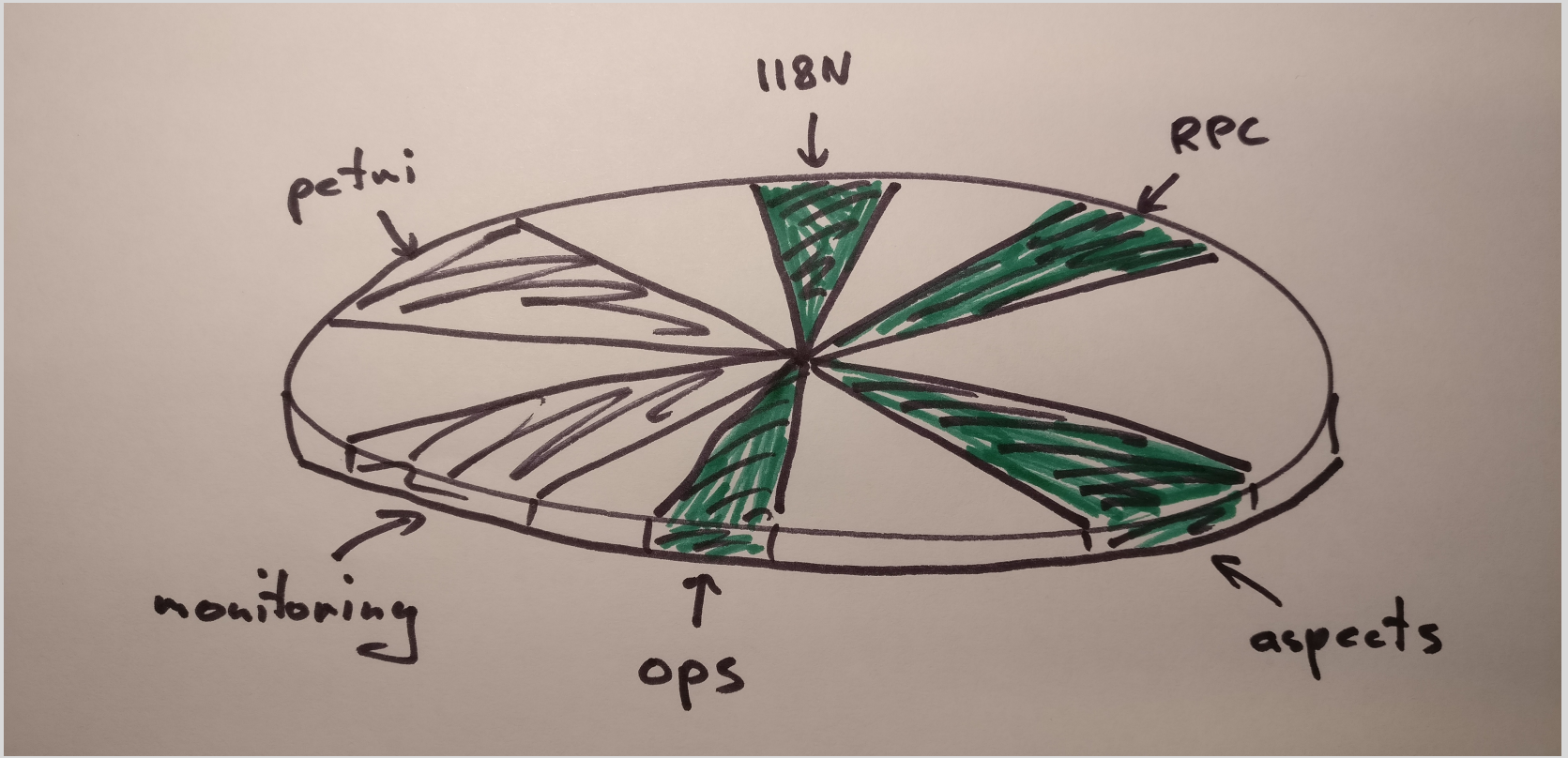
Proxy APIs for fat clients



Cover bare-minimum to run in production



And then… we got noticed!


I18N
RPC
petni
monitoring
ops
aspects

Test-kits based off of a core platform

- RPC
- Petri
- Session

```
const {expect} = require('chai'),
  testkit = require('./support/testkit'),
  http = require('wnp-http-test-client'),
  jvmTestkit = require('wix-jvm-bootstrap-testkit');

describe('petri client', function () {
  const app = testkit.server('petri').beforeAndAfter();
  const jvmTestkit = jvmTestkit.server({
    artifact:{
      groupId: 'com.wixpress.node',
      artifactId: 'wix-spjs-test-server'
    }
  }).beforeAndAfter();


  it('should conduct AB test', () =>
    givenABTest('scope', 'anExperiment')
      .then(() => conductExperiment('anExperiment'))
      .then(res => expect(res).to.equal('true'))
 );

 //...
});
```

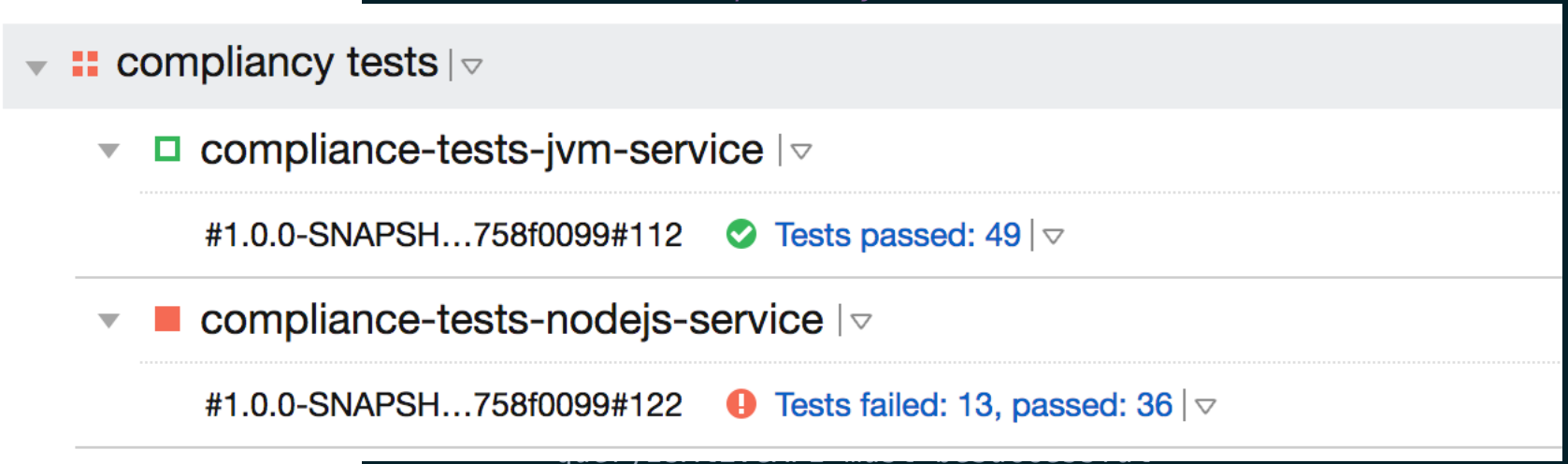
Time to get serious – compliancy tests are born.

Platform agnostic

Generic tests

- Using testkits;
- Platform-agnostic.

```
"health monitoring - protocol" >> {

  "/health/is_alive responds with failure" >> {
    healthTestDependency.becomeUnavailable()

    eventually {
      queryIsAliveAPI must beUnavailable
    }
  }

  "/health/is_alive responds with success" >> {
    healthTestDependency.becomeAvailable()
```




```
    }
  }
```

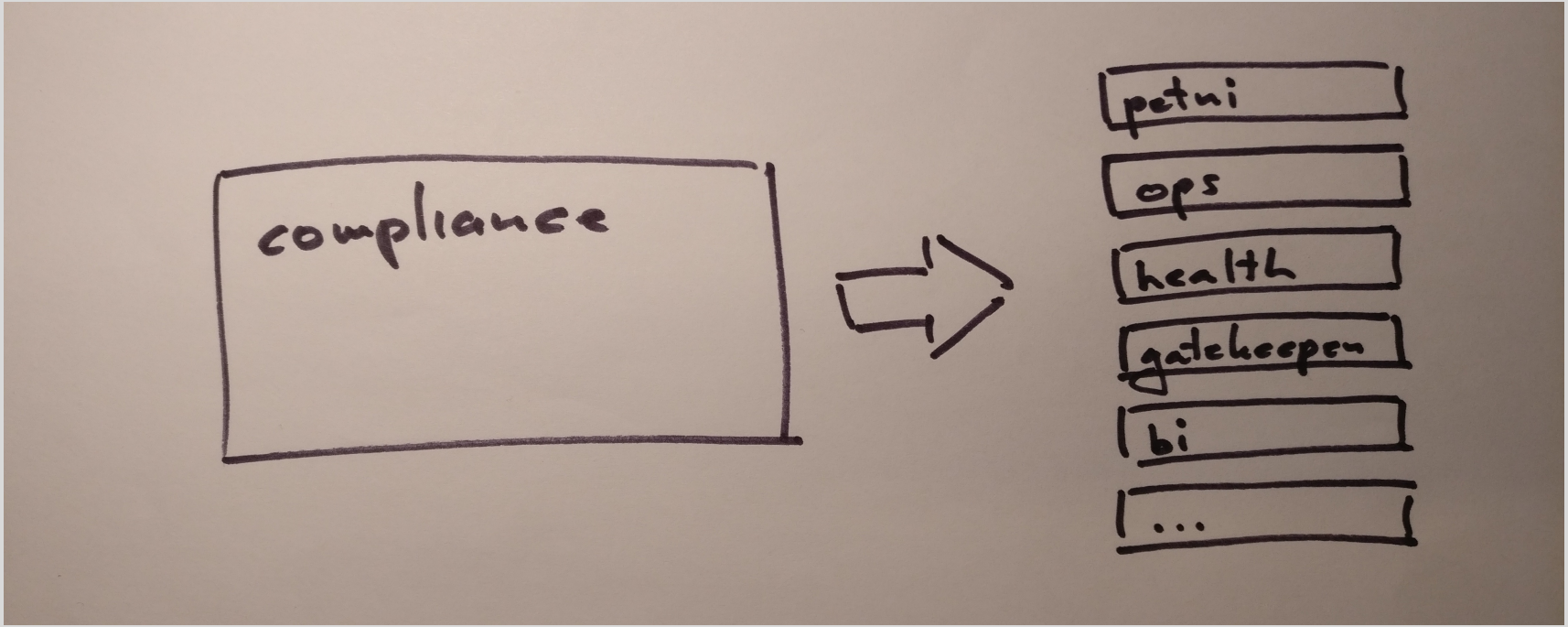
Easy to add another stack

- App with exposed APIs
- Adapters
- Runner

```
class NodeComplianceE2E extends BaseComplianceE2E
  with HealthAll
  with SecurityAll
  with I18nAll
  with PetriAll
  with BiAll
  with WebAll
  with NodeMetricKeys
  with NodeBiKeys
  with RpcAll {

  LogbackTestHelper.initLogger()

  override protected def systemUnderTest: MainService =
    new LocalNodeService
}
```

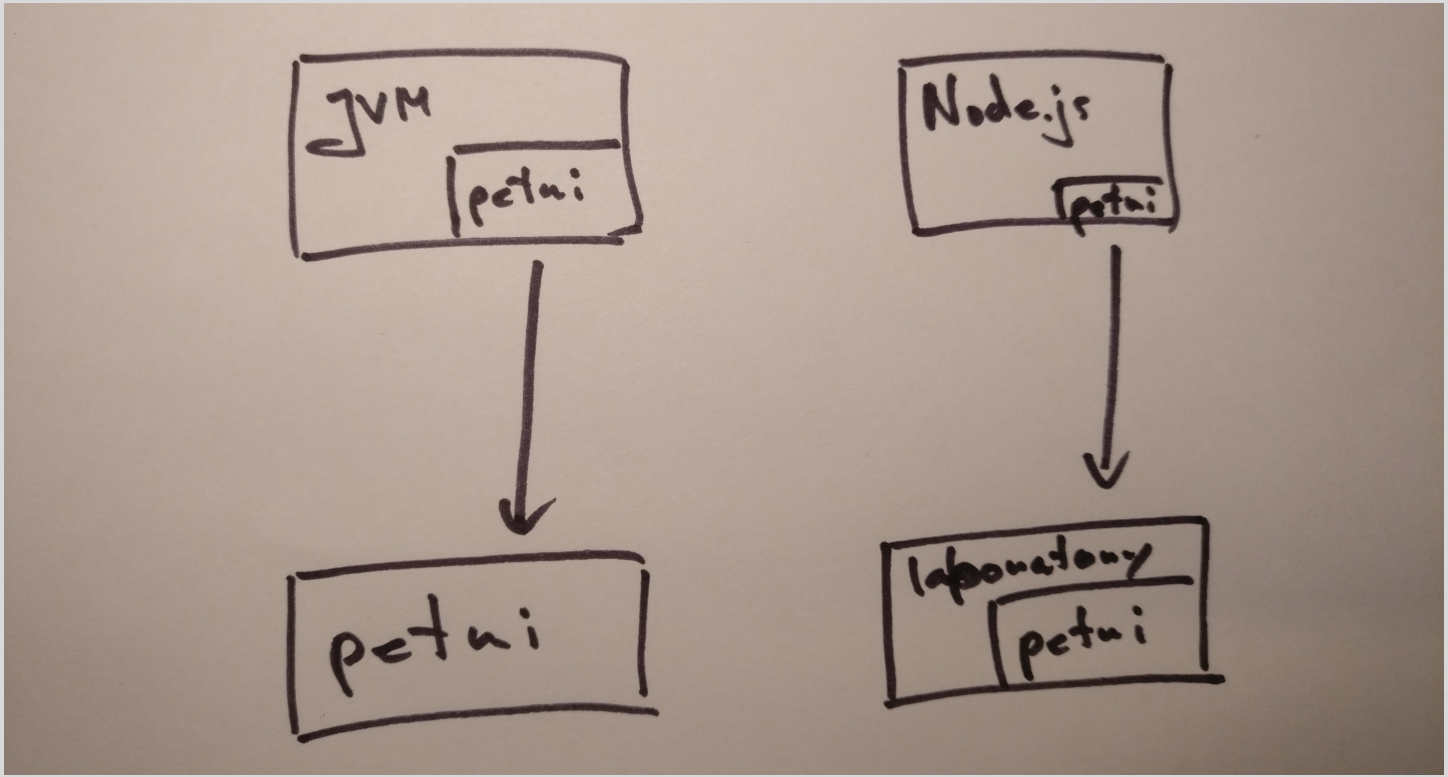
What is a contract and what is not?

Isn’t it like… monolithic?

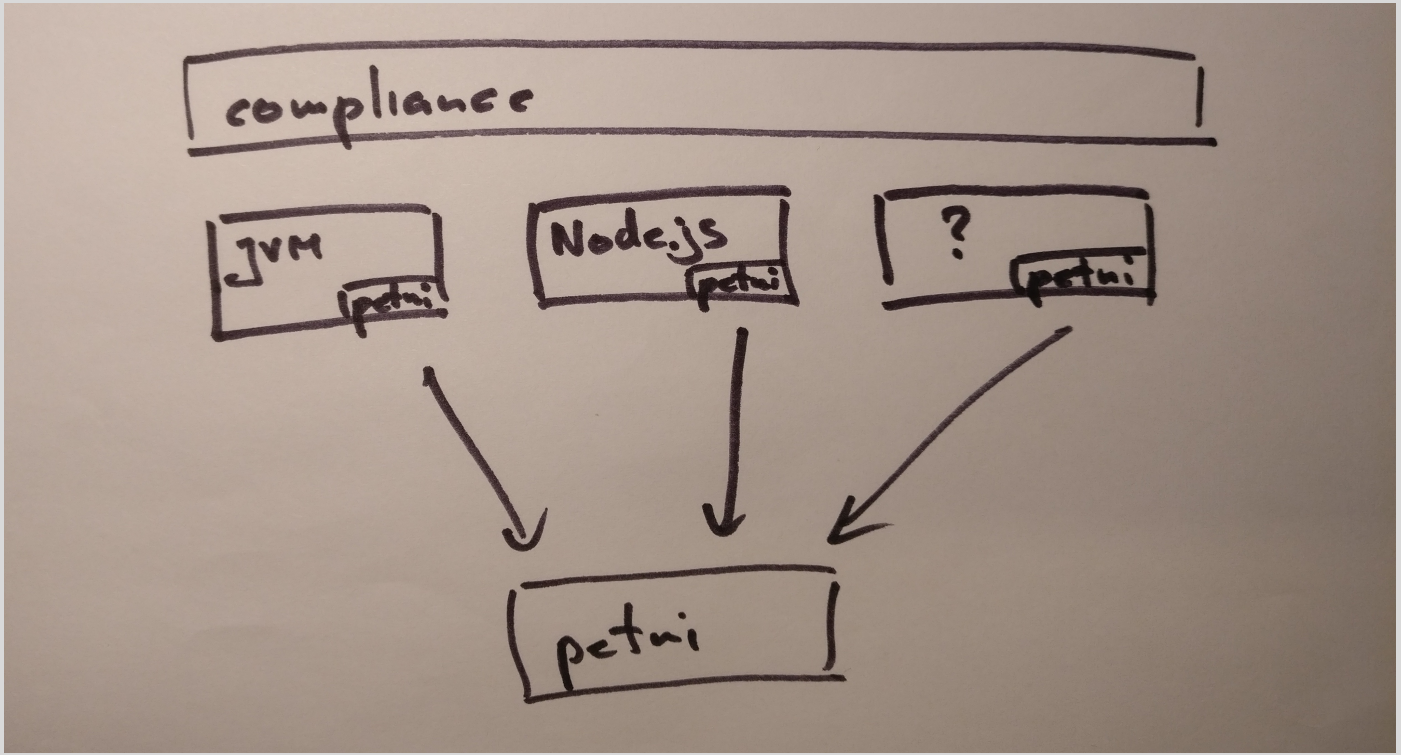

compliance
petni
ops
health
gatekeepen
bi
...

We are at the junction

Proxy



API's and compliance



So that we could pass a Litmus test

“can I actually take a team of engineers who are interested in X becoming a legit thing in my service and actually build something without convincing the rest of the company?”

Ben Christensen, Facebook

WiXEngineering

Thank You

 viliusl@wix.com

 @viliusl

 linkedin/viliusl

 github.com/viliusl

WiX Engineering

Questions?

 viliusl@wix.com

 @viliusl

 linkedin/viliusl

 github.com/viliusl