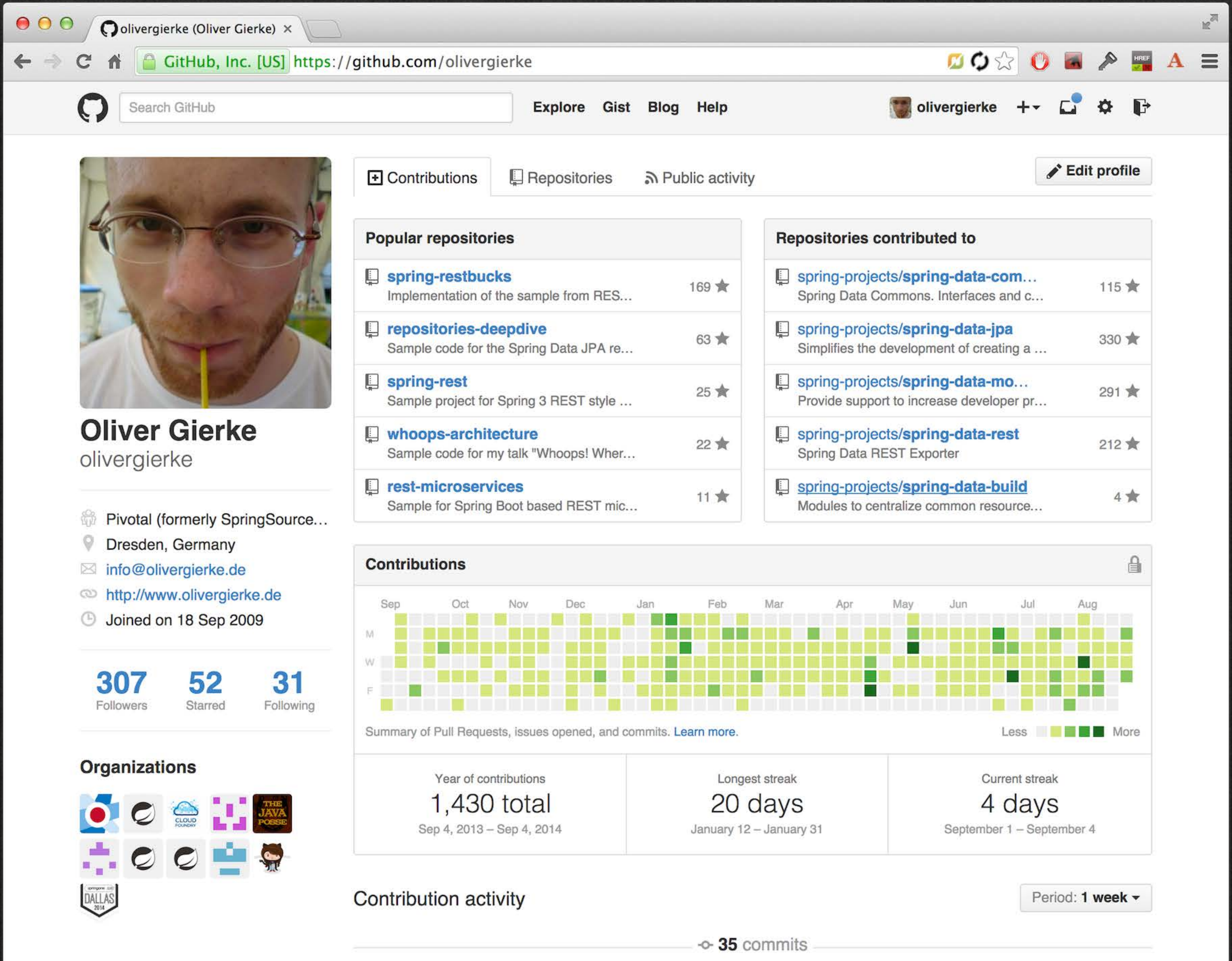
Whoops! Where did my architecture go?

Architecture management for Java applications

Oliver Gierke



olivergierke (Oliver Gierke)
GitHub, Inc. [US] https://github.com/olivergierke
Search GitHub
Explore
Gist
Blog
Help
olivergierke

Oliver Gierke
olivergierke

Pivotal (formerly SpringSource…
Dresden, Germany
info@olivergierke.de
http://www.olivergierke.de
Joined on 18 Sep 2009

307 Followers
52 Starred
31 Following

Organizations

Popular repositories

spring-restbucks Implementation of the sample from RES…	169 ★
repositories-deepdive Sample code for the Spring Data JPA re…	63 ★
spring-rest Sample project for Spring 3 REST style …	25 ★
whoops-architecture Sample code for my talk "Whoops! Wher…	22 ★
rest-microservices Sample for Spring Boot based REST mic…	11 ★

Repositories contributed to

spring-projects/**spring-data-com…** Spring Data Commons. Interfaces and c…	115 ★
spring-projects/**spring-data-jpa** Simplifies the development of creating a …	330 ★
spring-projects/**spring-data-mo…** Provide support to increase developer pr…	291 ★
spring-projects/**spring-data-rest** Spring Data REST Exporter	212 ★
spring-projects/**spring-data-build** Modules to centralize common resource…	4 ★

Contributions

Summary of Pull Requests, issues opened, and commits. Learn more.

Year of contributions	Longest streak	Current streak
1,430 total	20 days	4 days
Sep 4, 2013 – Sep 4, 2014	January 12 – January 31	September 1 – September 4

Contribution activity

Period: **1 week**

35 commits

Oliver Gierke
@olivergierke

Spring Data Project Lead @ Pivotal, OpenSource enthusiast, musician, producer... Soul Power!

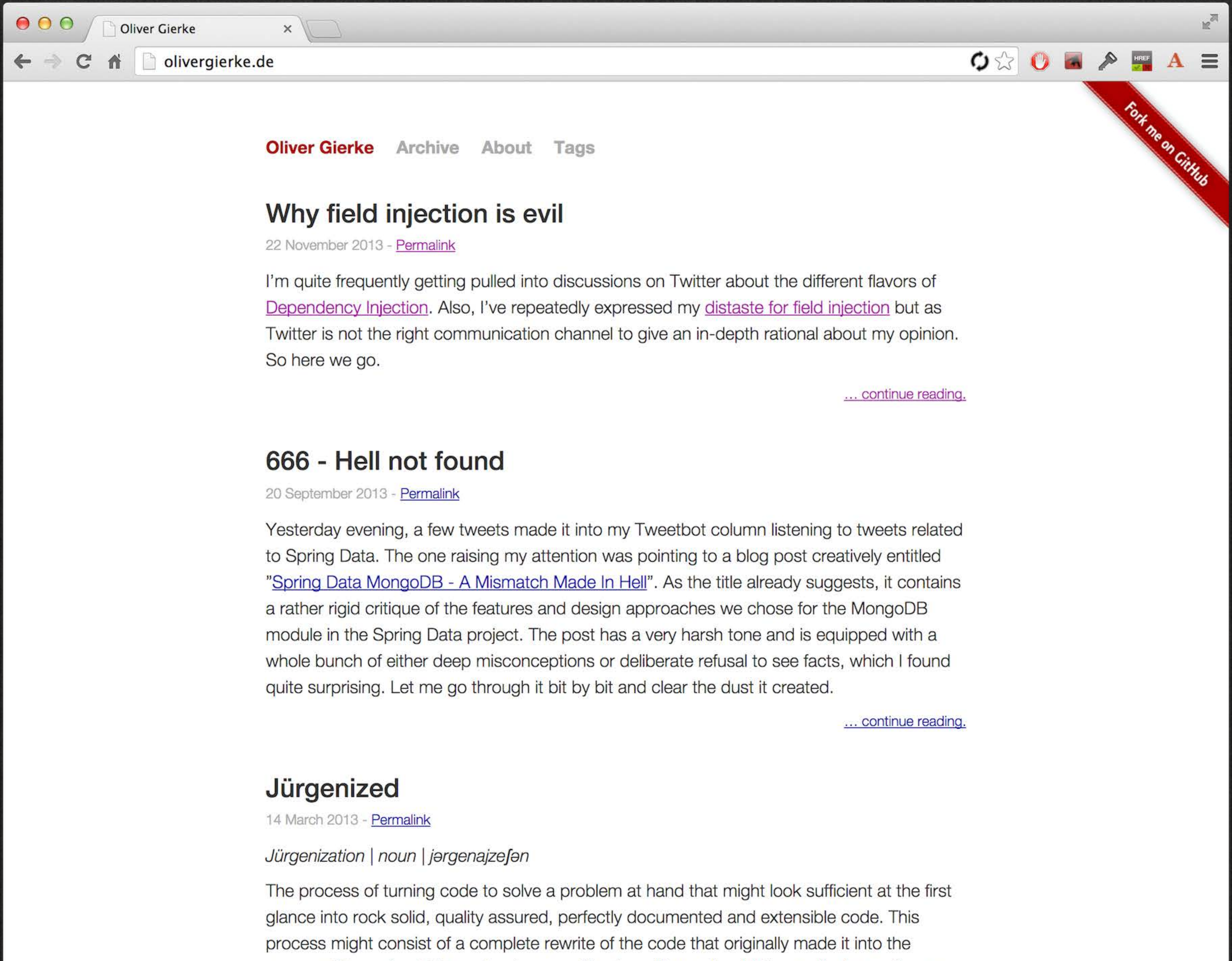
Dresden, Germany
olivergierke.de
Joined October 2008

161 Photos and videos

Retweeted by Oliver Gierke
Stéphane Nicoll @snicoll · 2h
Spring Framework 4.1 GA is here! spring.io/blog/2014/09/0… 4.0.7 and 3.2.11 are also released today
40 5 View summary

Retweeted by Oliver Gierke
Sam Brannen @sam_brannen · 1h
#Spring Framework 4.1 GA is here! Take it for a spin an let us know your thoughts. :) bit.ly/1vS10Yv #java
9 View summary

Retweeted by Oliver Gierke
sındnp uɐıʇsıɹɥɔ @cdupuis · 4h



Oliver Gierke Archive About Tags

Why field injection is evil

22 November 2013 - Permalink

I'm quite frequently getting pulled into discussions on Twitter about the different flavors of Dependency Injection. Also, I've repeatedly expressed my distaste for field injection but as Twitter is not the right communication channel to give an in-depth rational about my opinion. So here we go.

… continue reading.

666 - Hell not found

20 September 2013 - Permalink

Yesterday evening, a few tweets made it into my Tweetbot column listening to tweets related to Spring Data. The one raising my attention was pointing to a blog post creatively entitled "Spring Data MongoDB - A Mismatch Made In Hell". As the title already suggests, it contains a rather rigid critique of the features and design approaches we chose for the MongoDB module in the Spring Data project. The post has a very harsh tone and is equipped with a whole bunch of either deep misconceptions or deliberate refusal to see facts, which I found quite surprising. Let me go through it bit by bit and clear the dust it created.

… continue reading.

Jürgenized

14 March 2013 - Permalink

Jürgenization | *noun* | *jərgenajzeʃən*

The process of turning code to solve a problem at hand that might look sufficient at the first glance into rock solid, quality assured, perfectly documented and extensible code. This process might consist of a complete rewrite of the code that originally made it into the

Background

5 years of consulting

Lots of code reviews

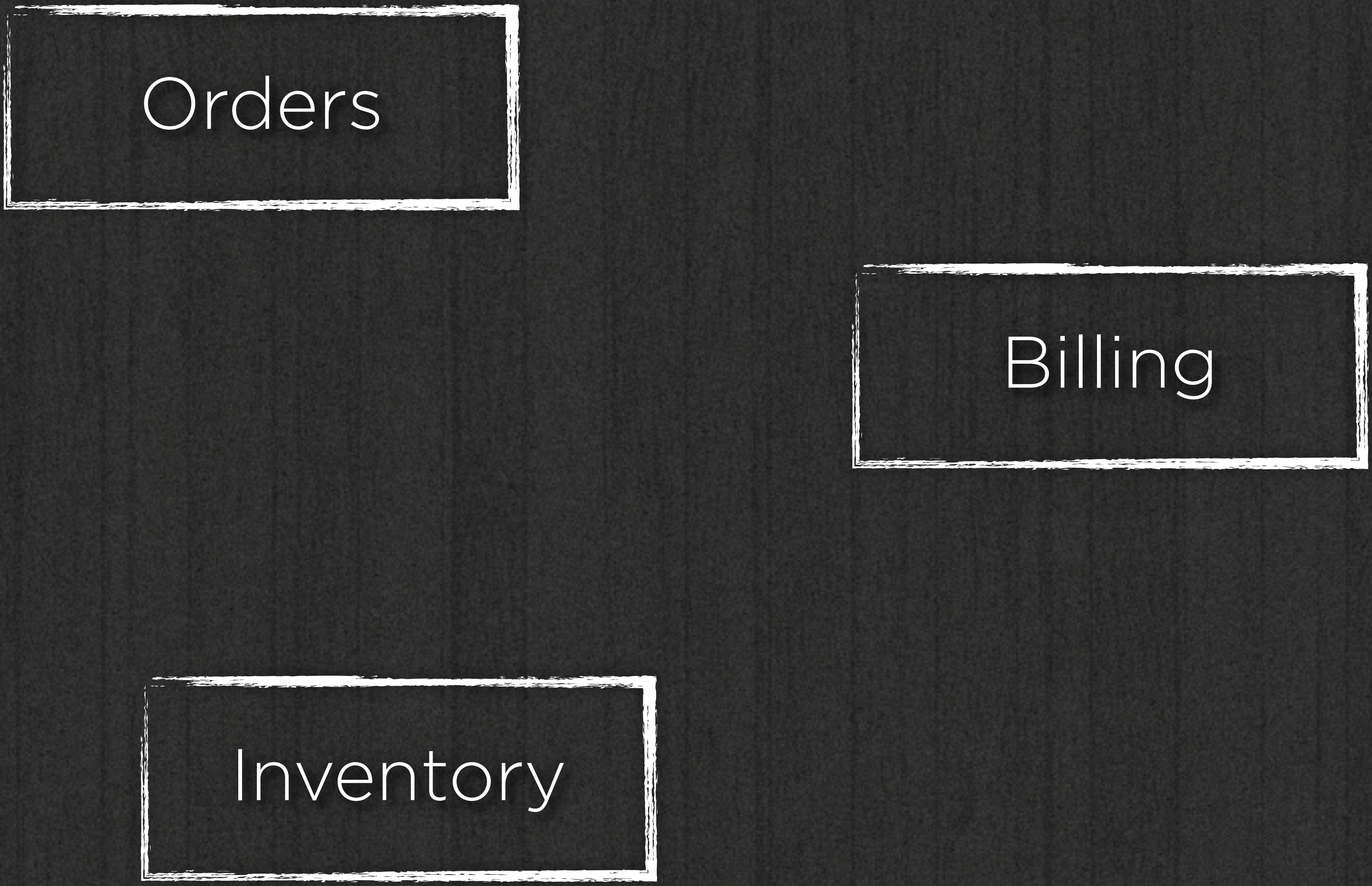
Eoin Woods‘ talk on InfoQ

Lot of input from fellows

> "If you think architecture is expensive, try no architecture.

Macro VS. Micro Architecture

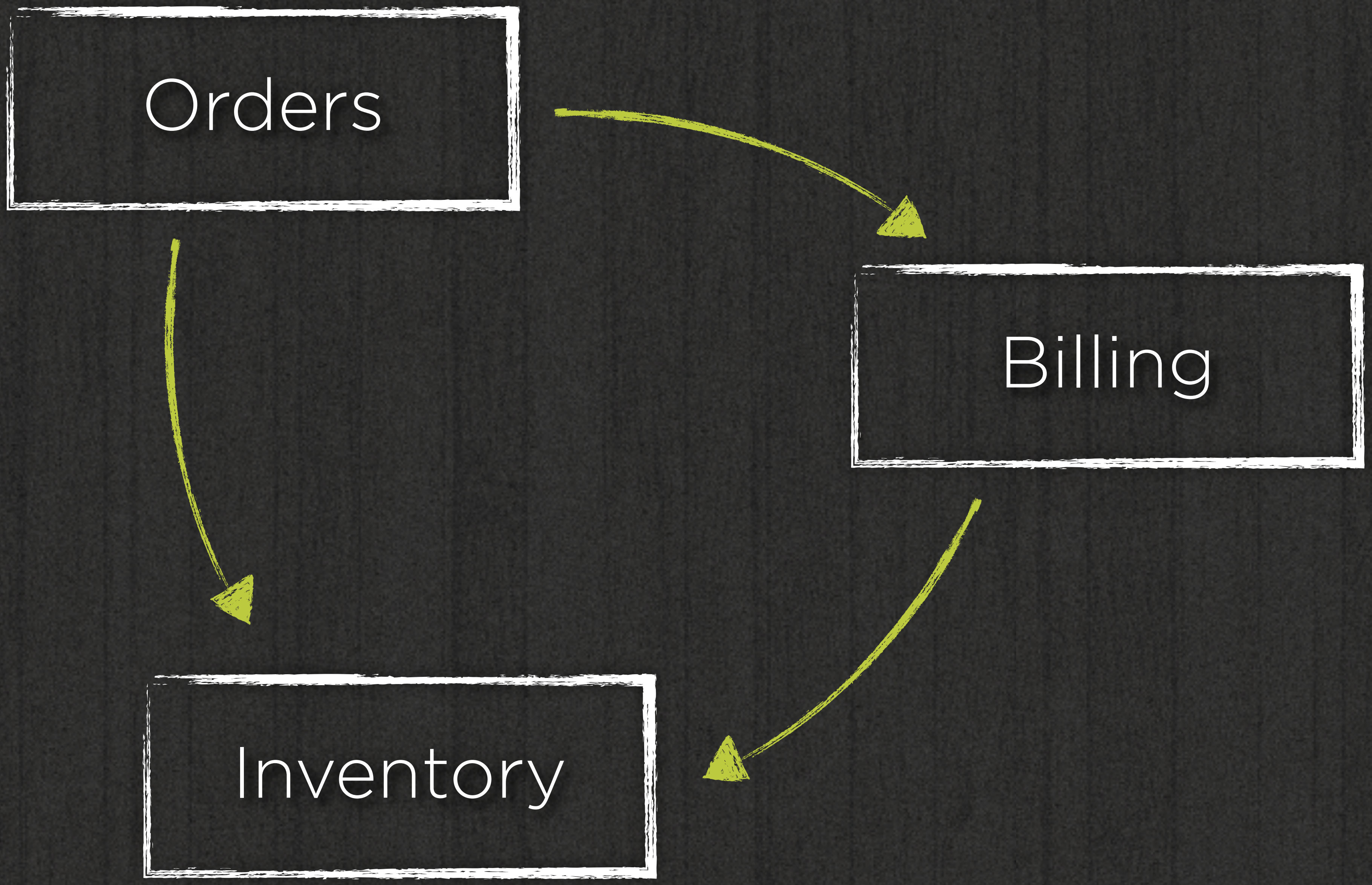
Macro VS. Micro Architecture



Orders

Billing

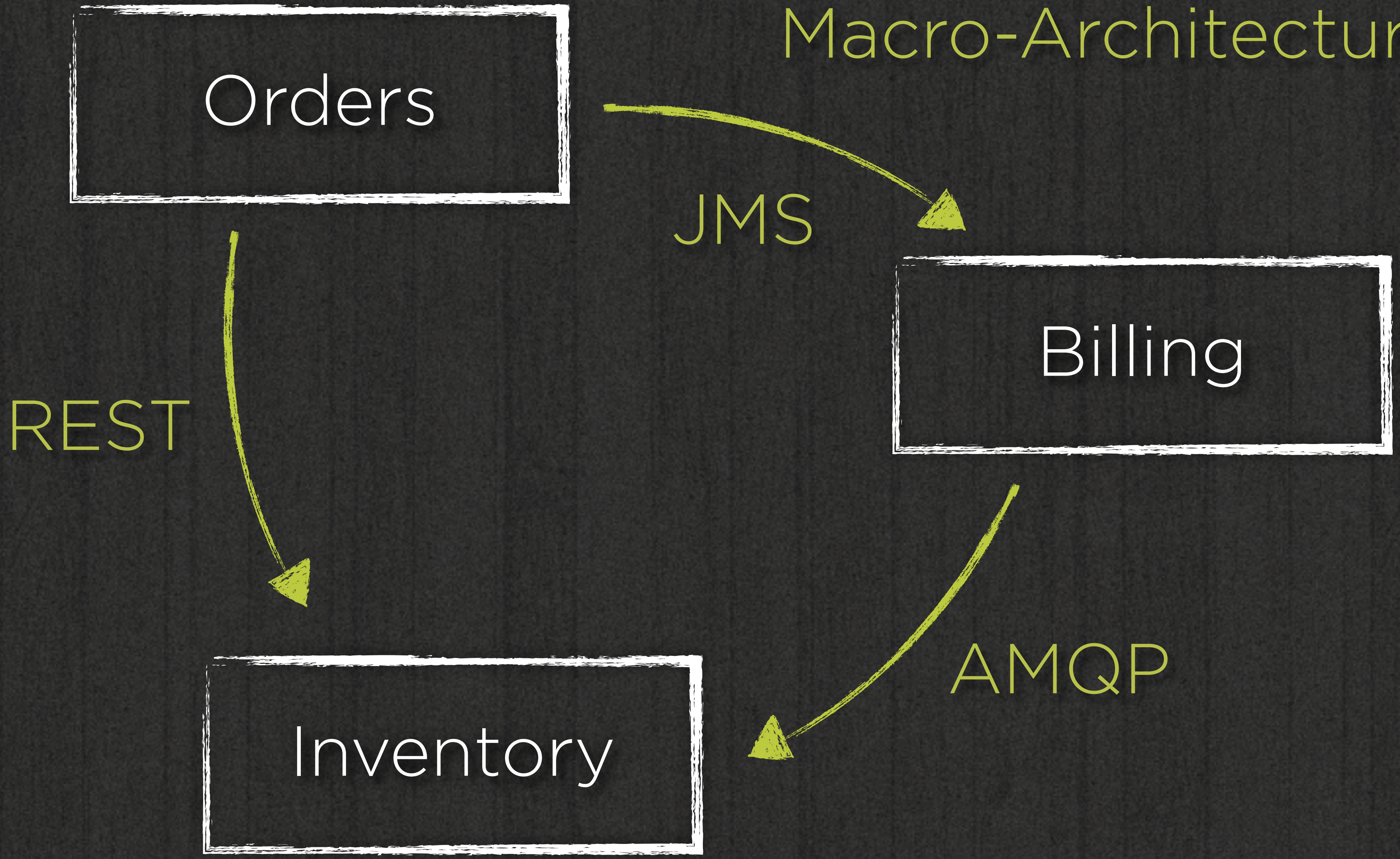
Inventory


Orders

Billing

Inventory

Macro-Architecture



Orders

JMS

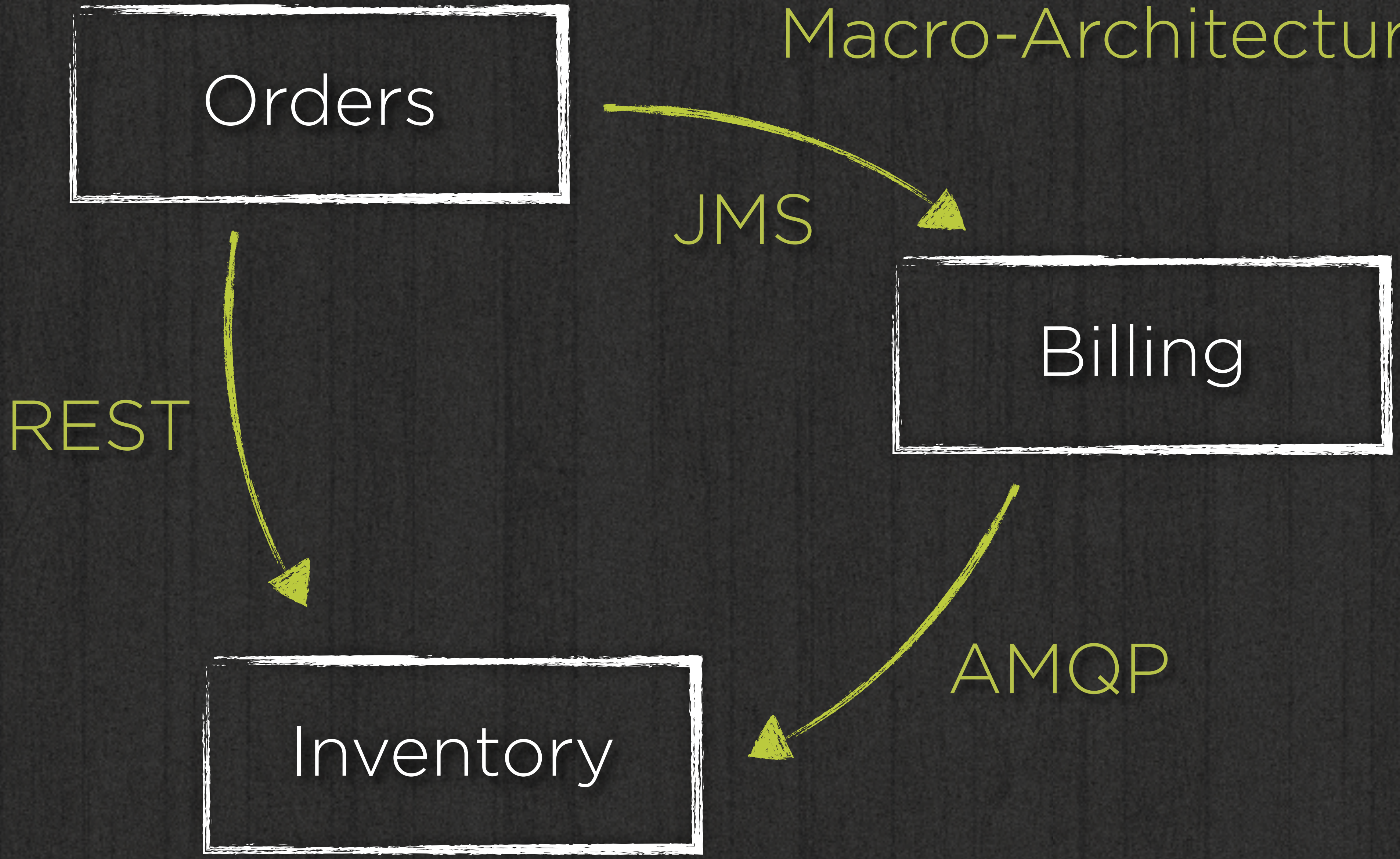
Billing

REST

AMQP

Inventory

Macro-Architecture



Orders

JMS

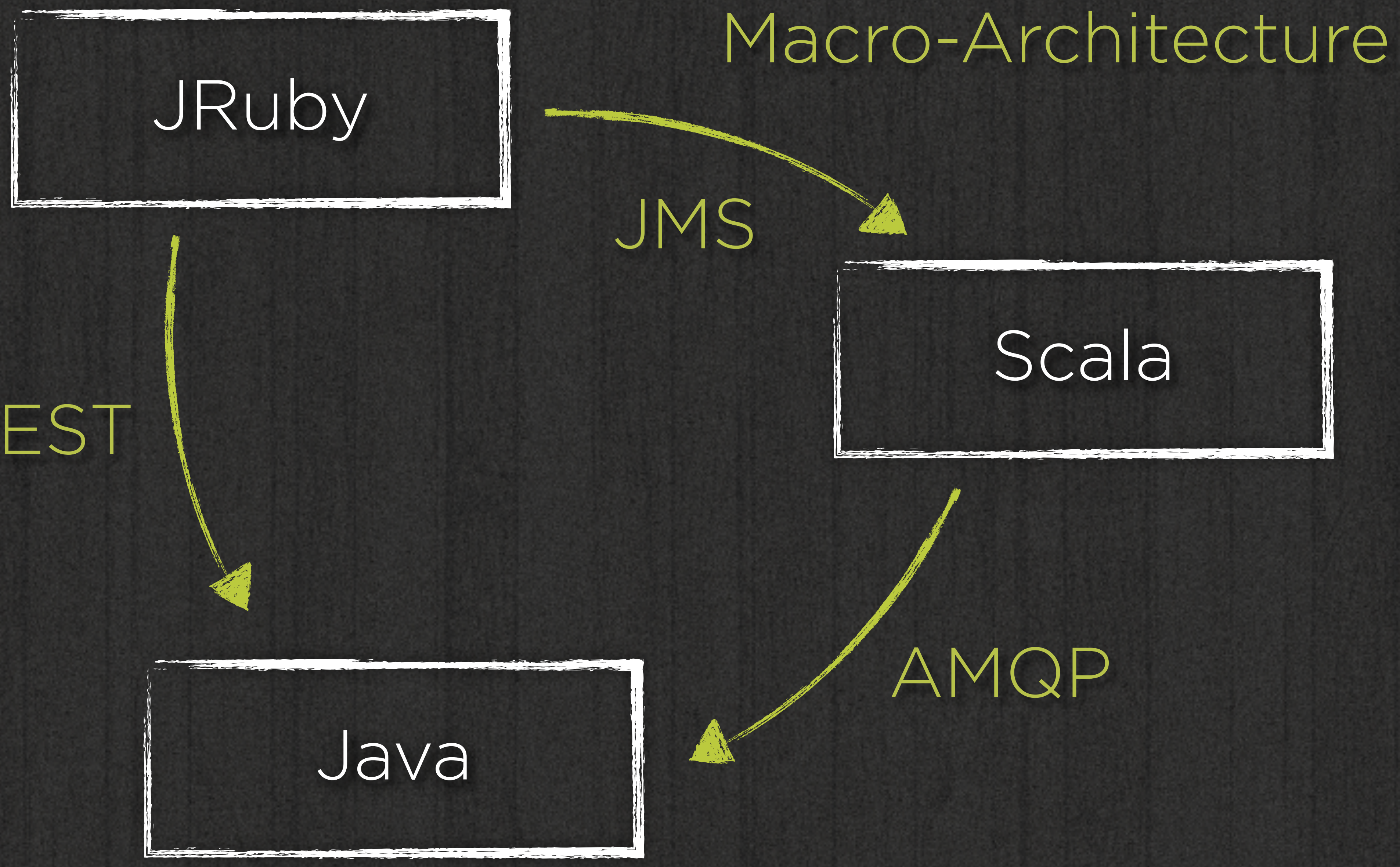
Billing

REST

AMQP

Inventory

Micro-Architecture



Macro-Architecture

JRuby

JMS

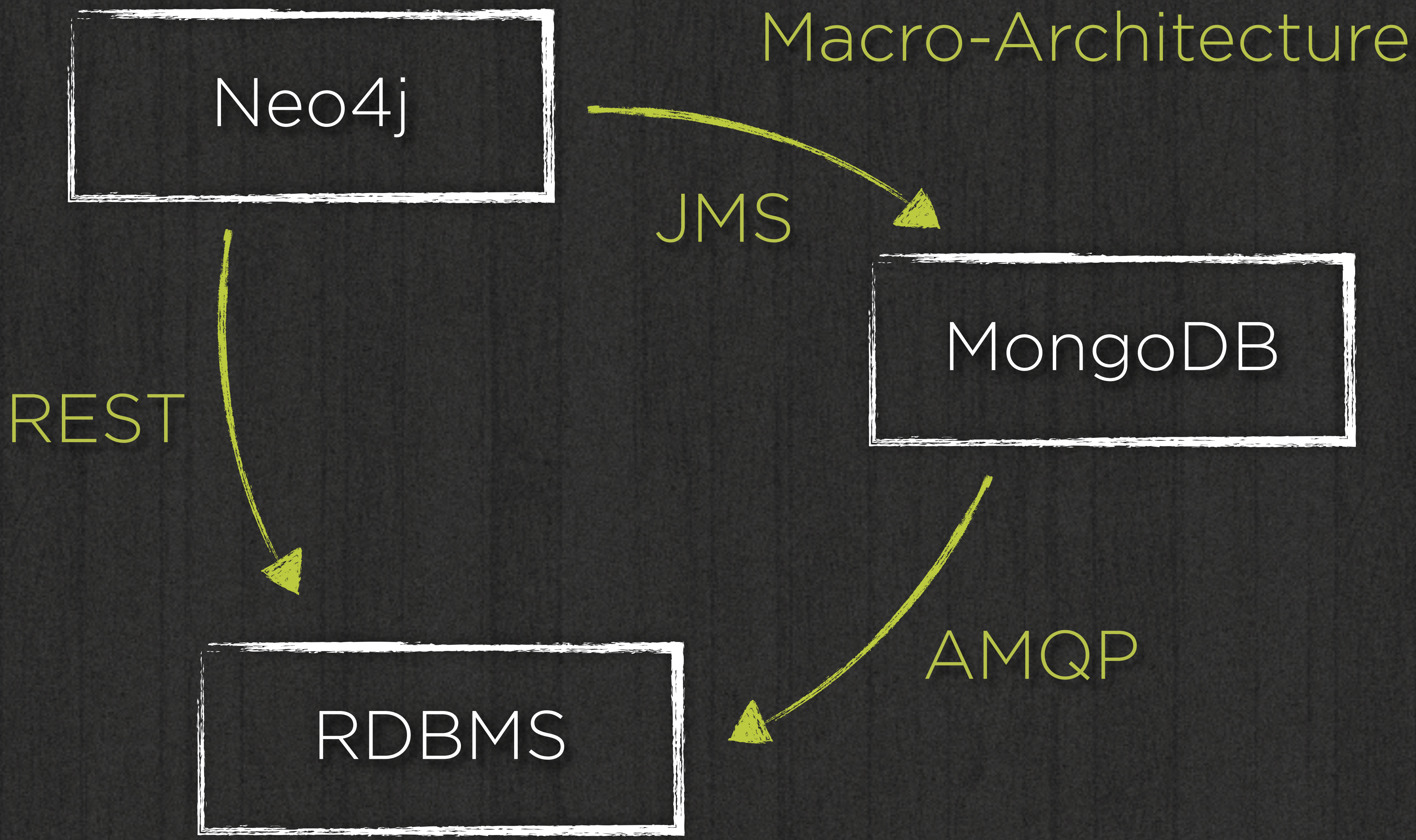
Scala

REST

AMQP

Java

Micro-Architecture



Macro-Architecture

Neo4j

JMS

MongoDB

REST

AMQP

RDBMS

Micro-Architecture

from Stefan Tilkov -
Breaking the Monolith

Roadmap

Divide and conquer

Of layers and slices

A plain Java based approach

Architecture 101

Know your dependencies

Explicit / Visible dependencies

Granularity

Modules

Layers

Vertical slices

Subsystems

Granularity

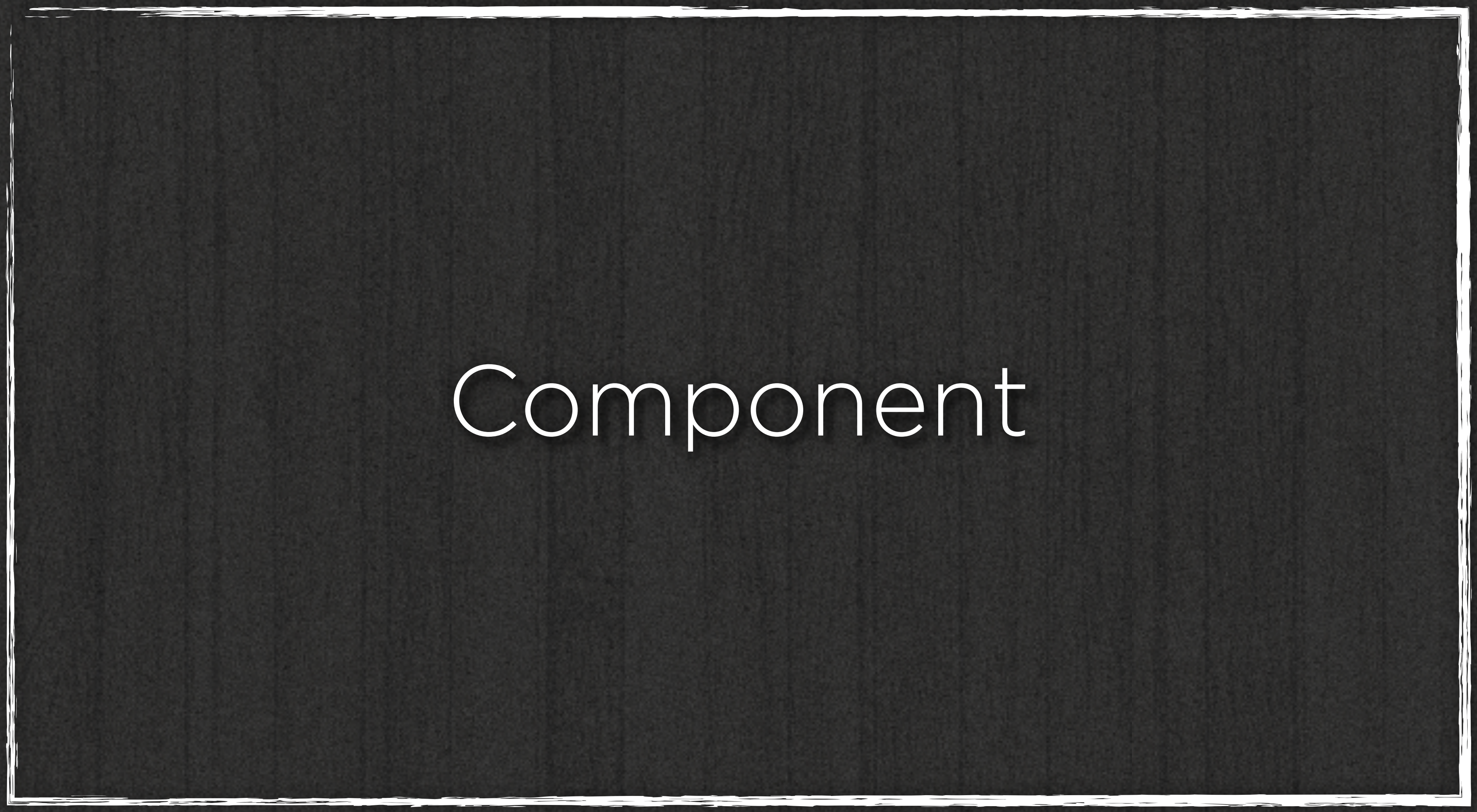
Java ARchive

Package

Class

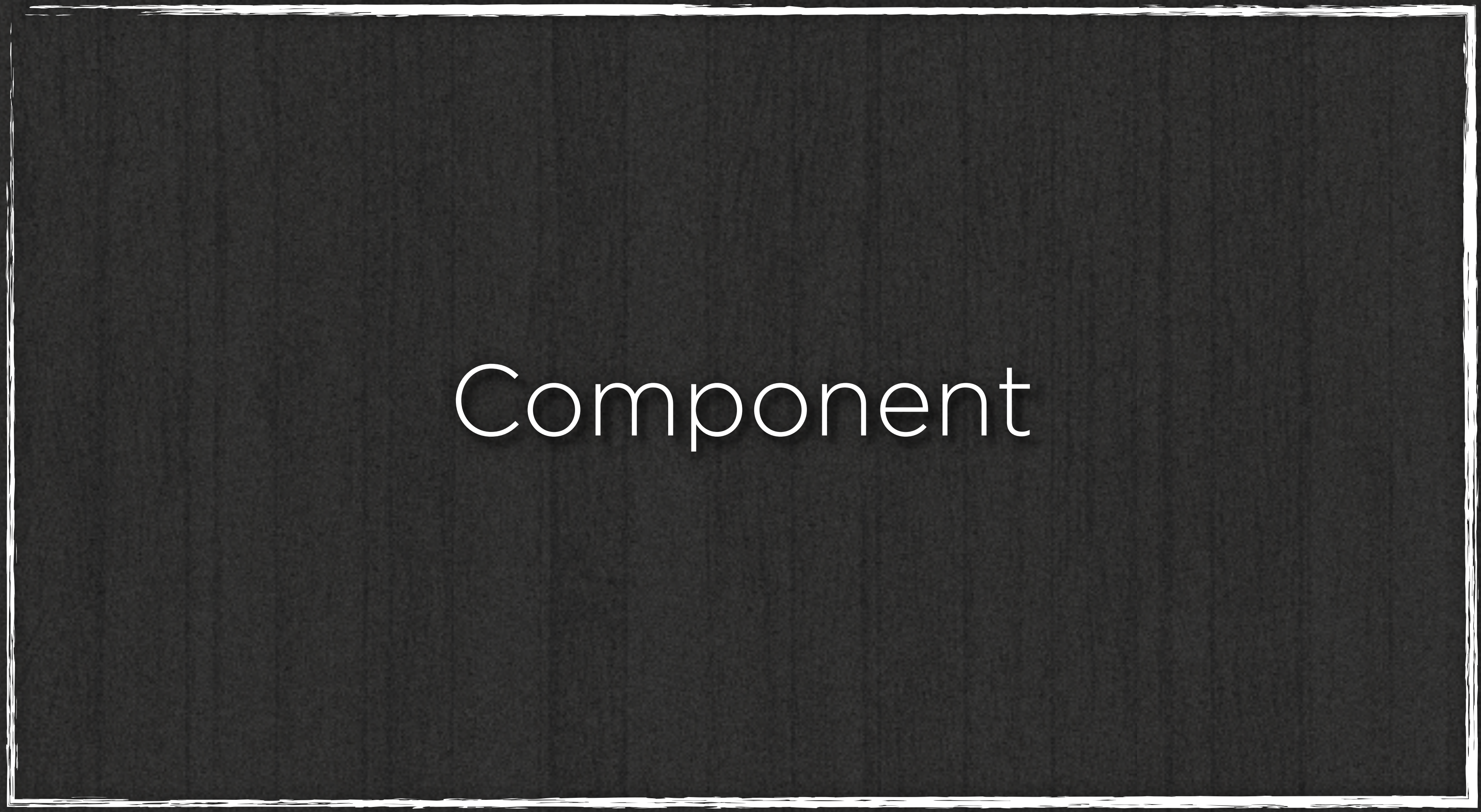
Divide and conquer

Component



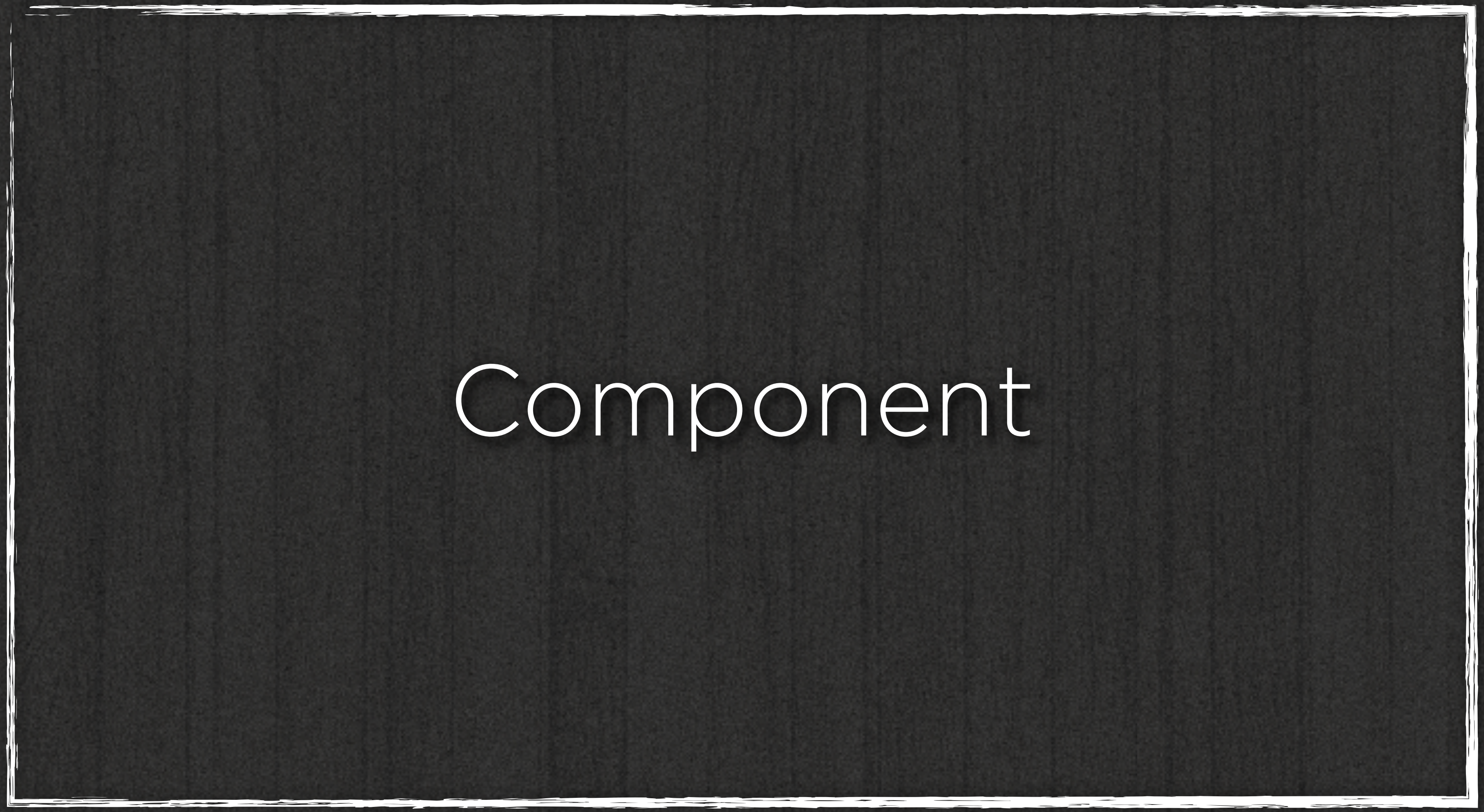
Component

Single unit to understand



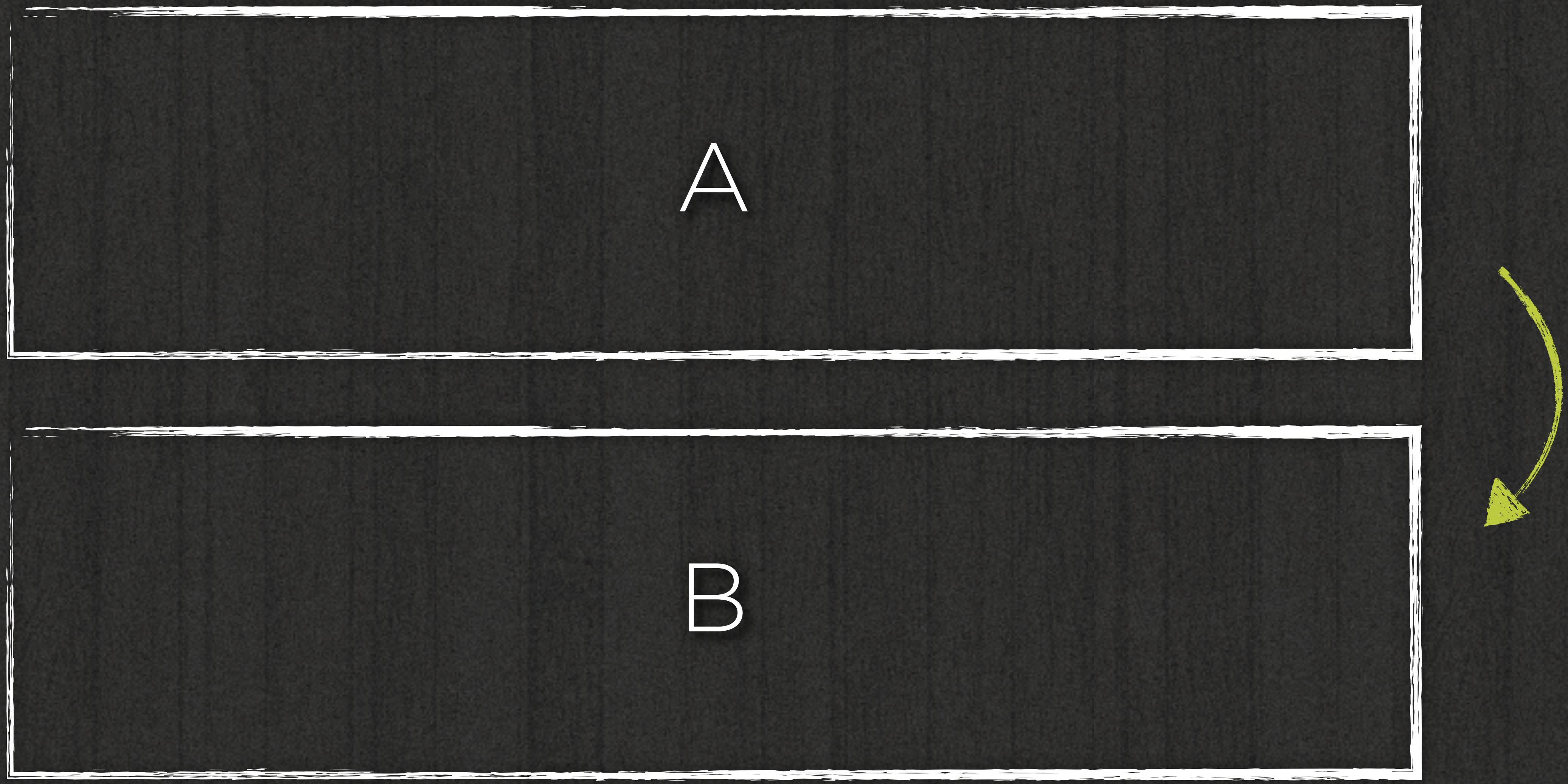
Component

Single unit to change

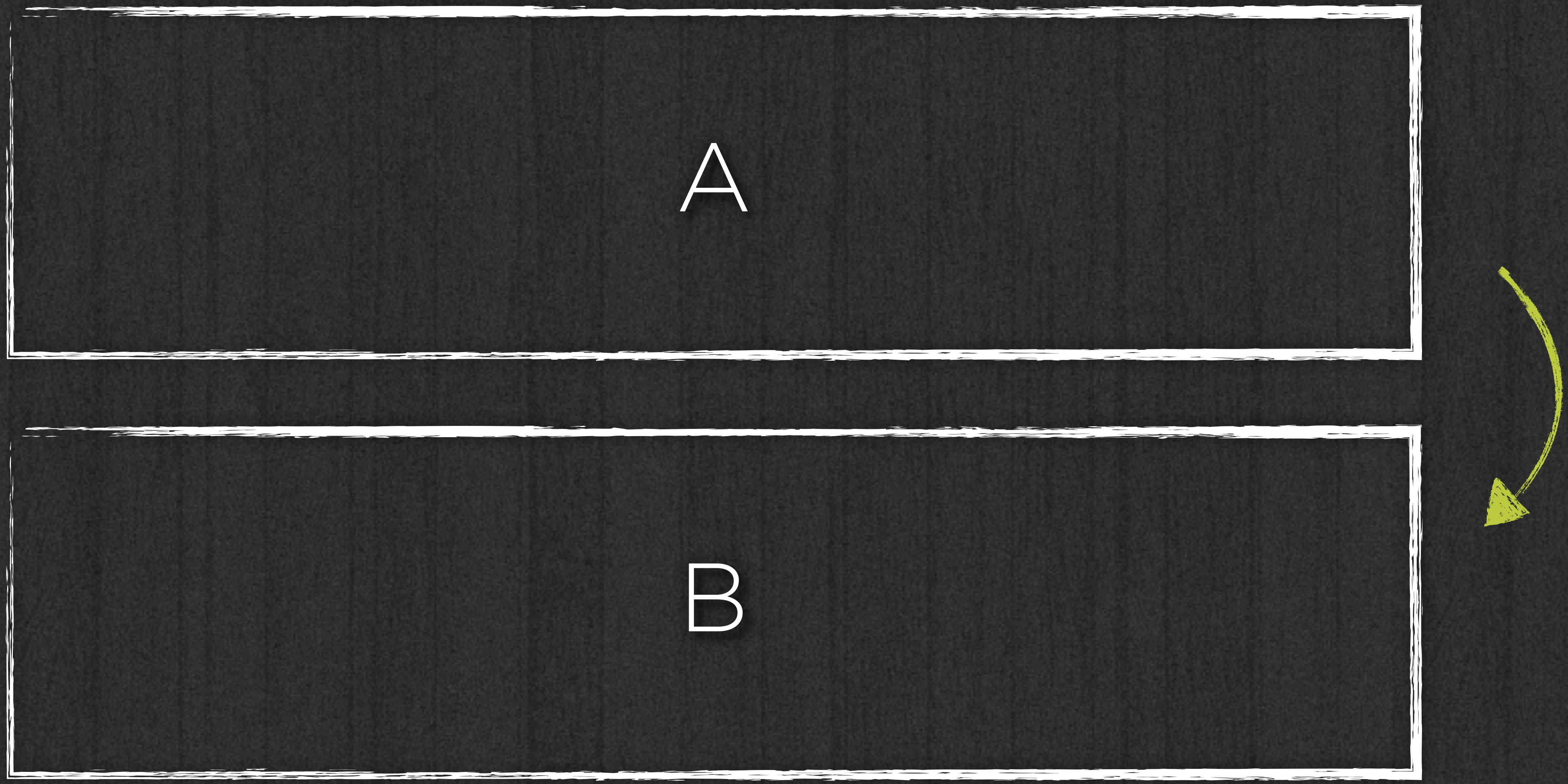


Component

Scope of risk of change



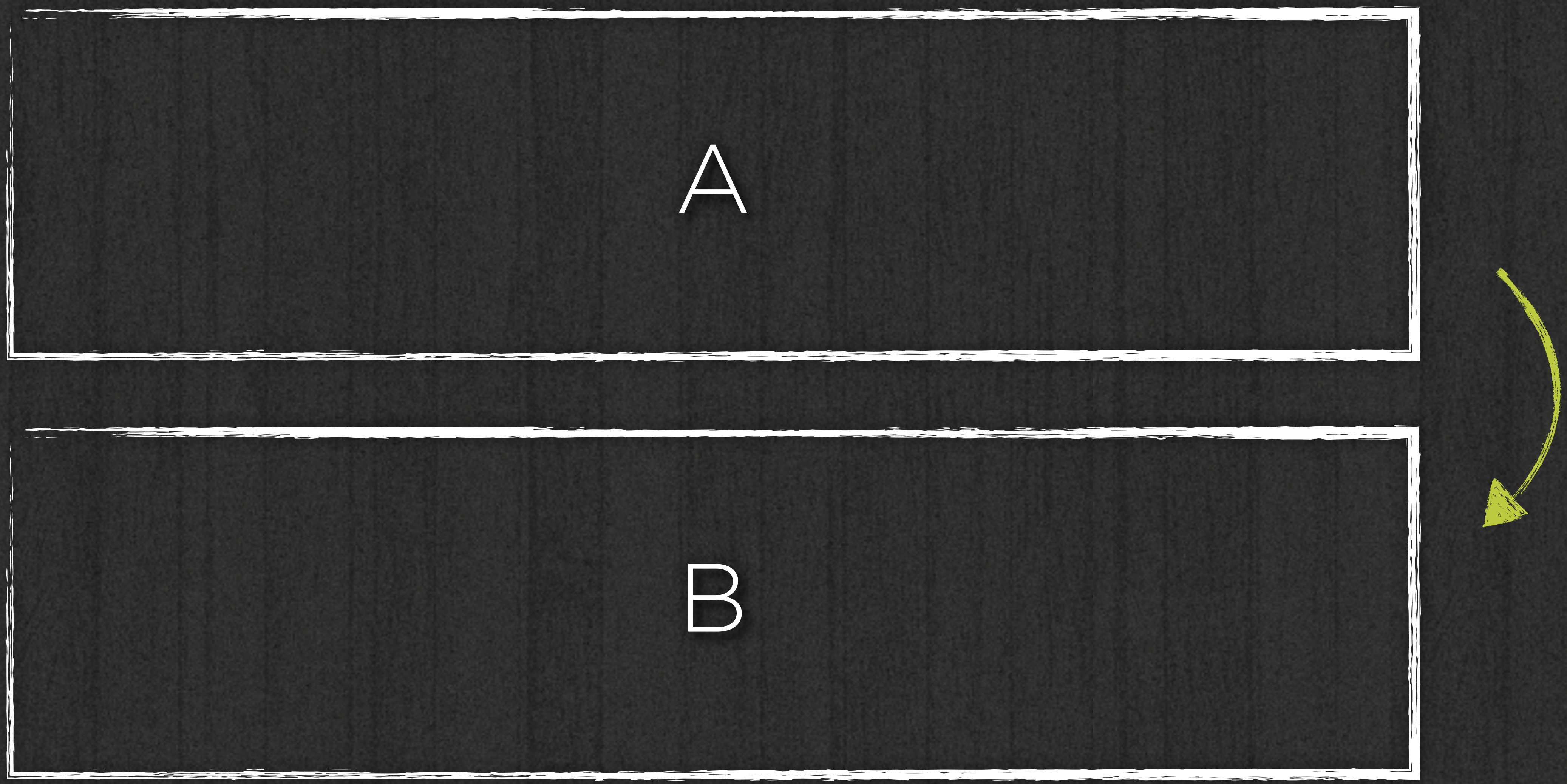
Smaller unit to understand



A

B

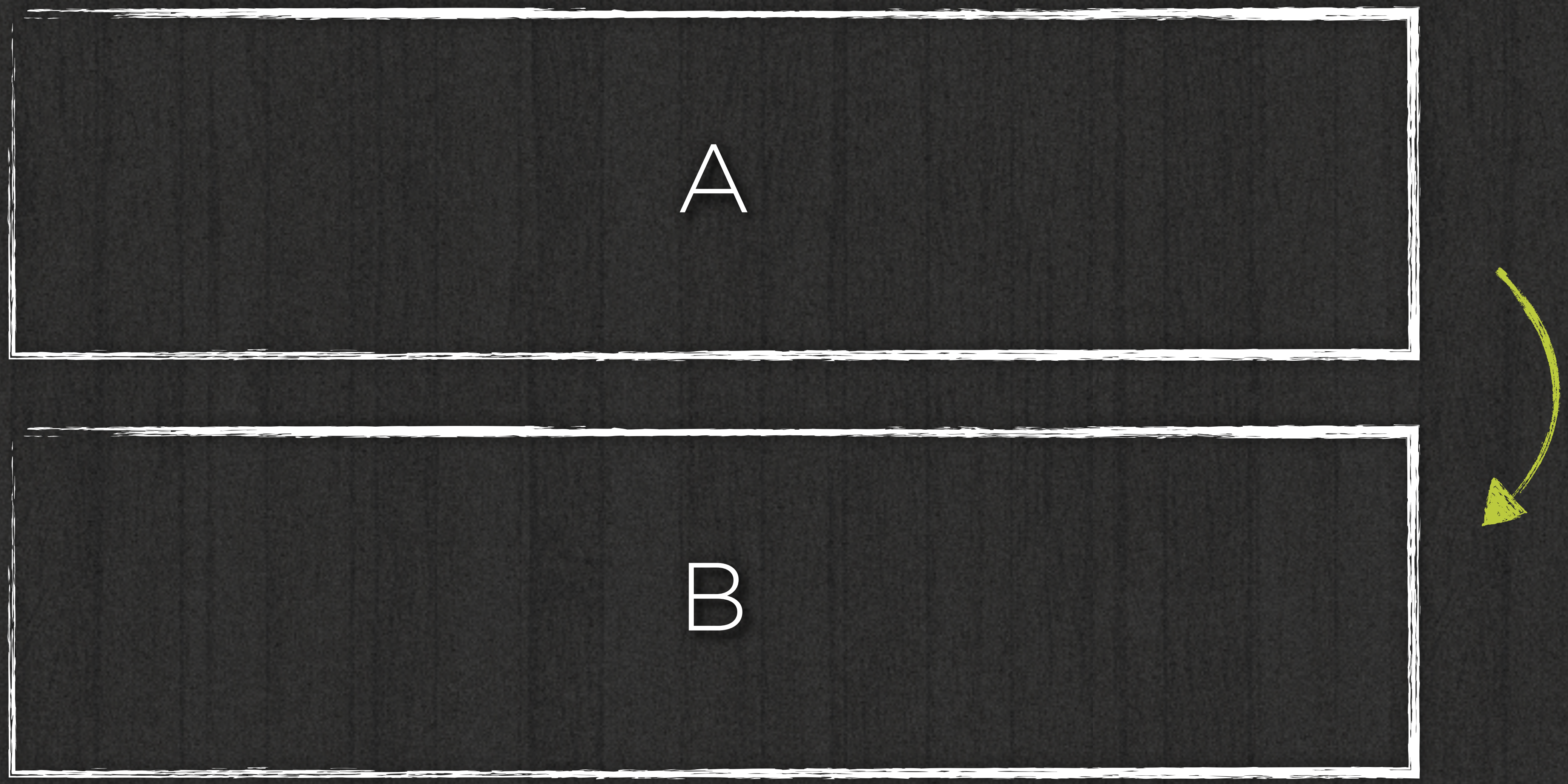
Reduced risk of change



A

B

Cost of separation

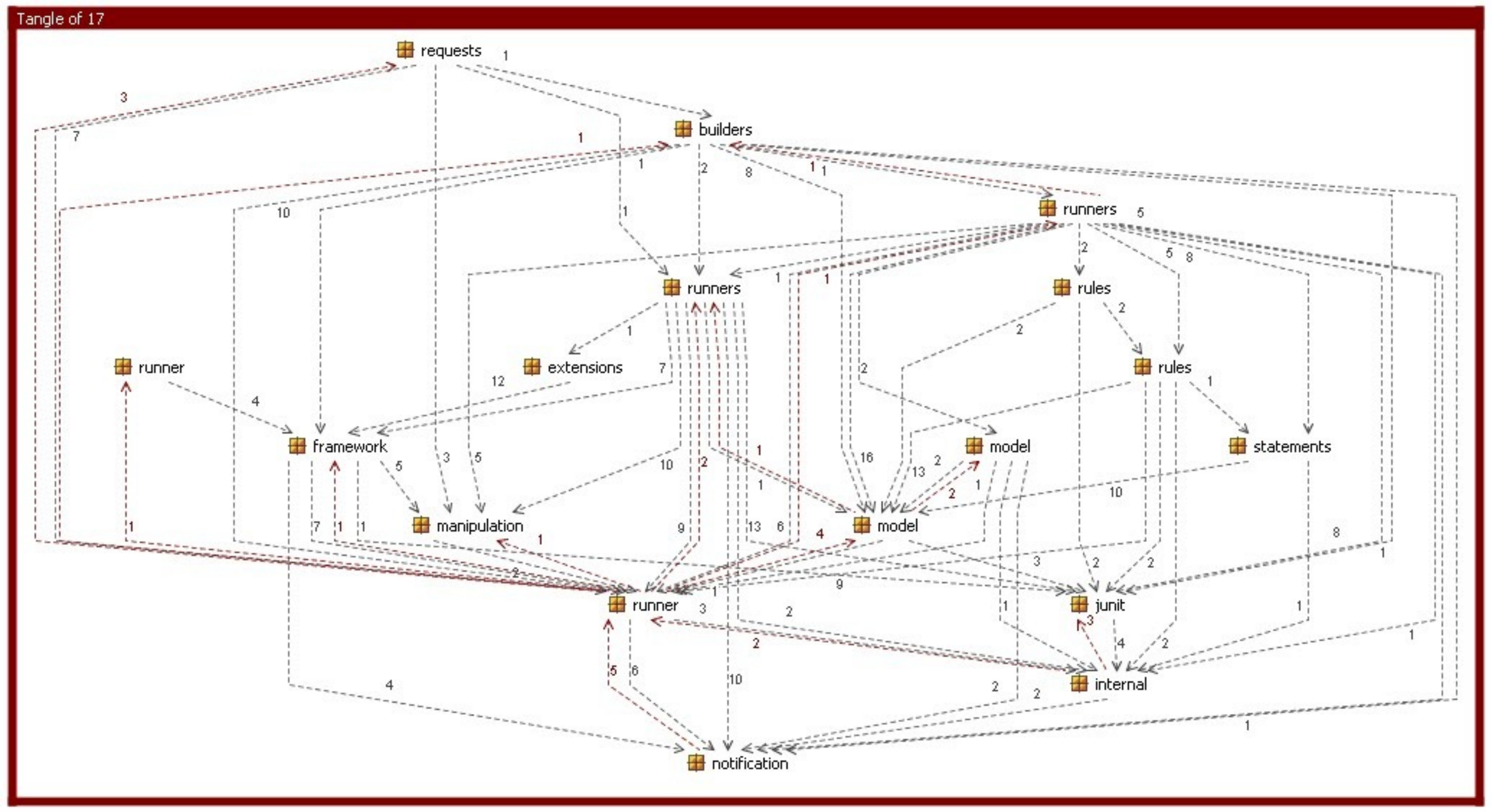


A

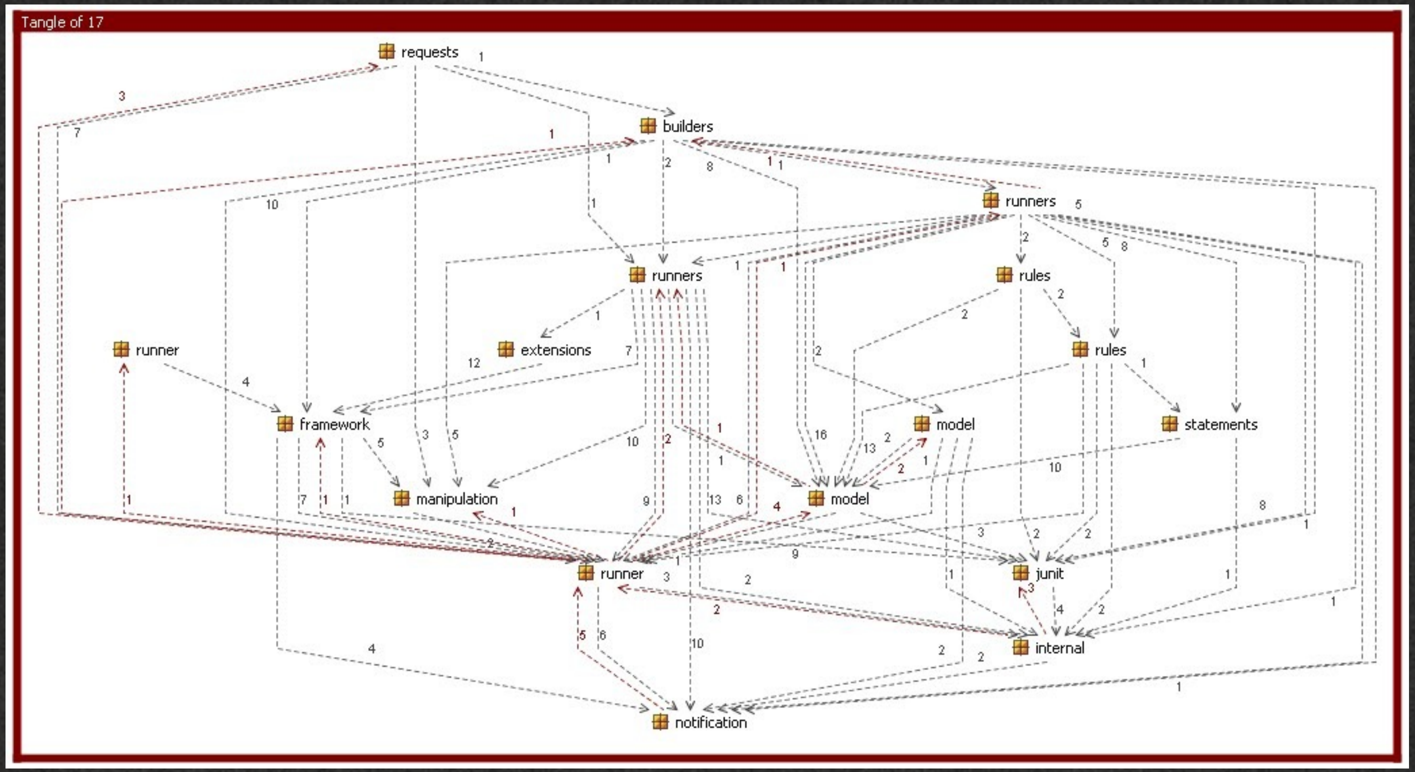
B

Definition and maintenance of dependencies

> " Meanwhile, in my friend's project...



How do I add a feature?

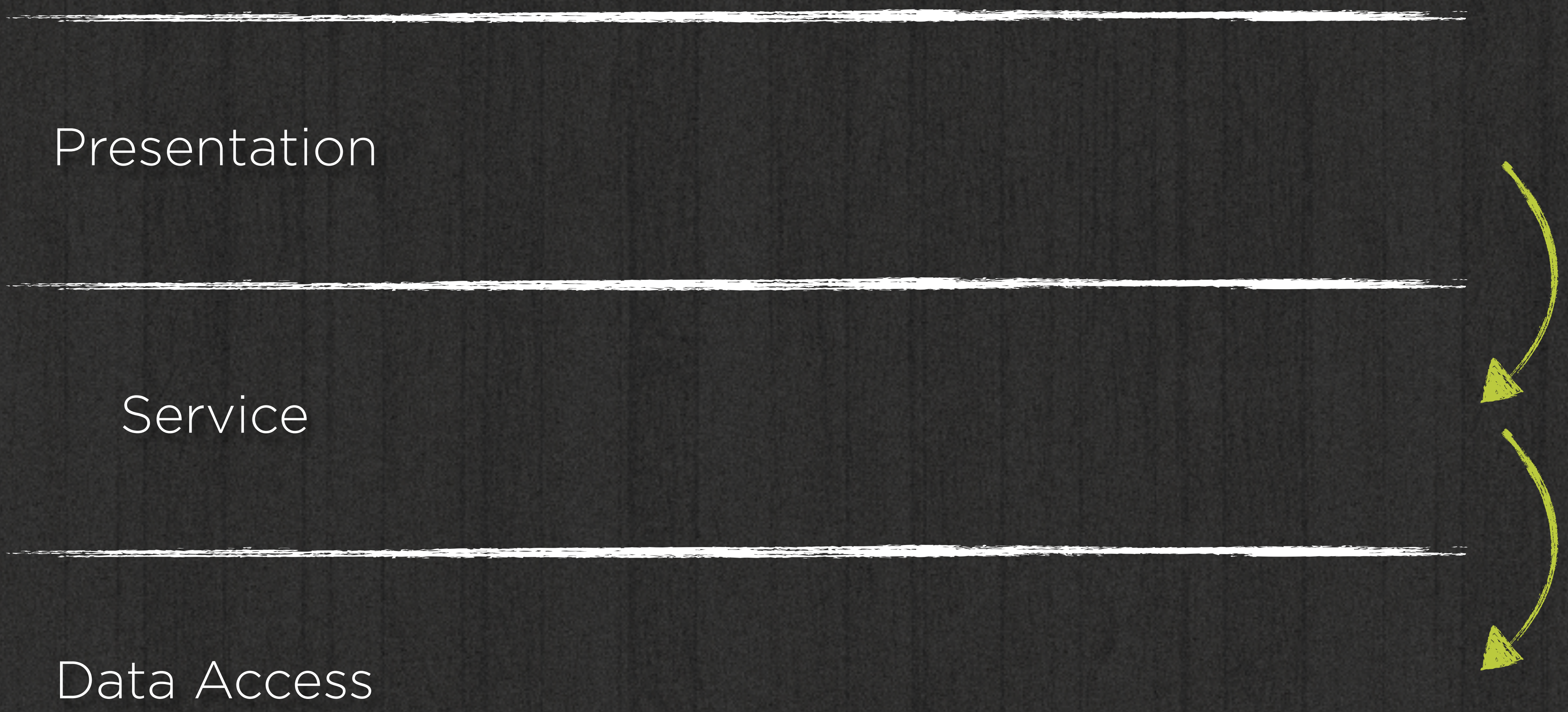


New Feature

How do I add a feature?

© Eberhard Wolff

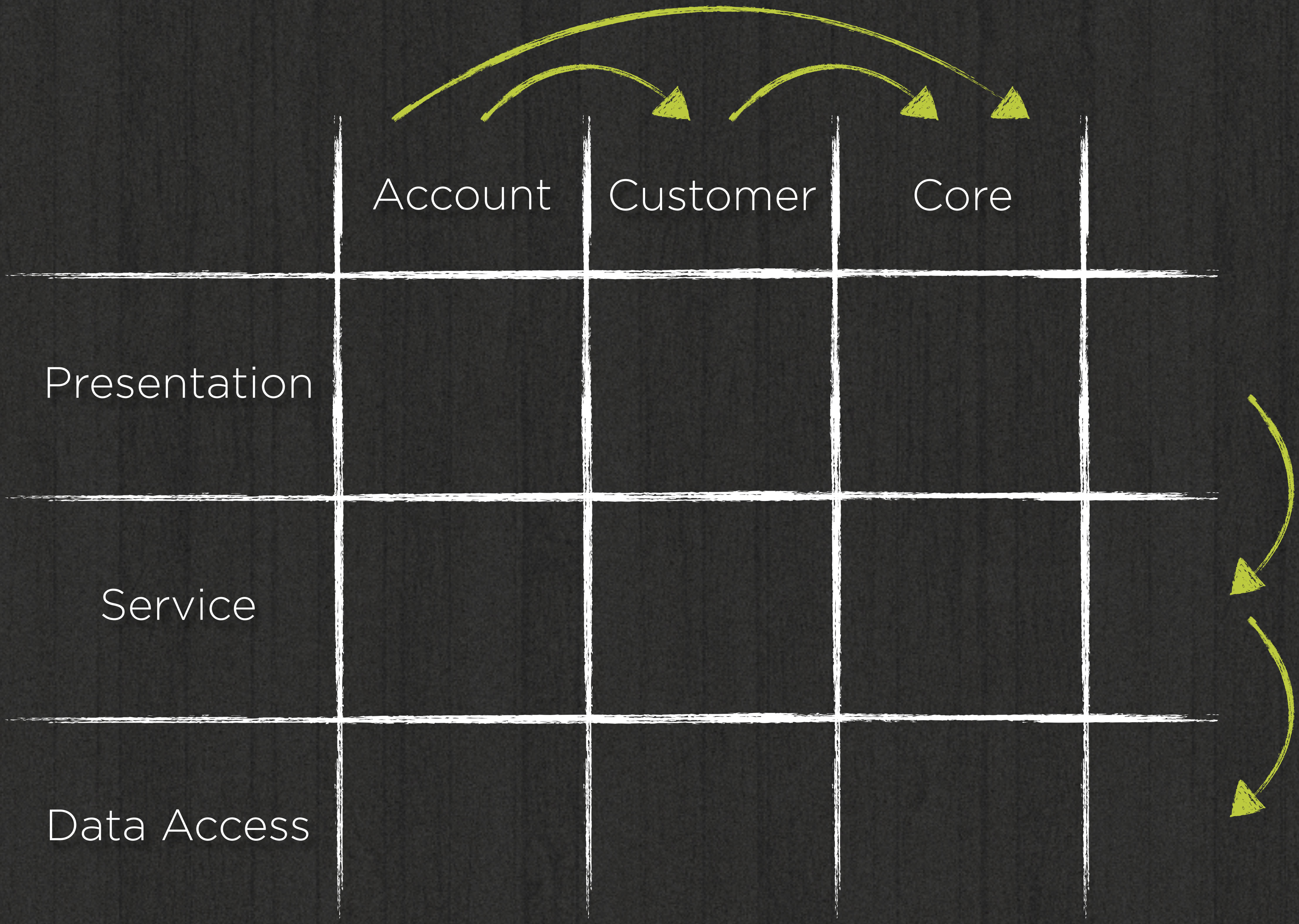
Of layers and slices…



Presentation

Service

Data Access



	Account	Customer	Core
Presentation			
Service			
Data Access			

Layers

Well understood

Known to developers

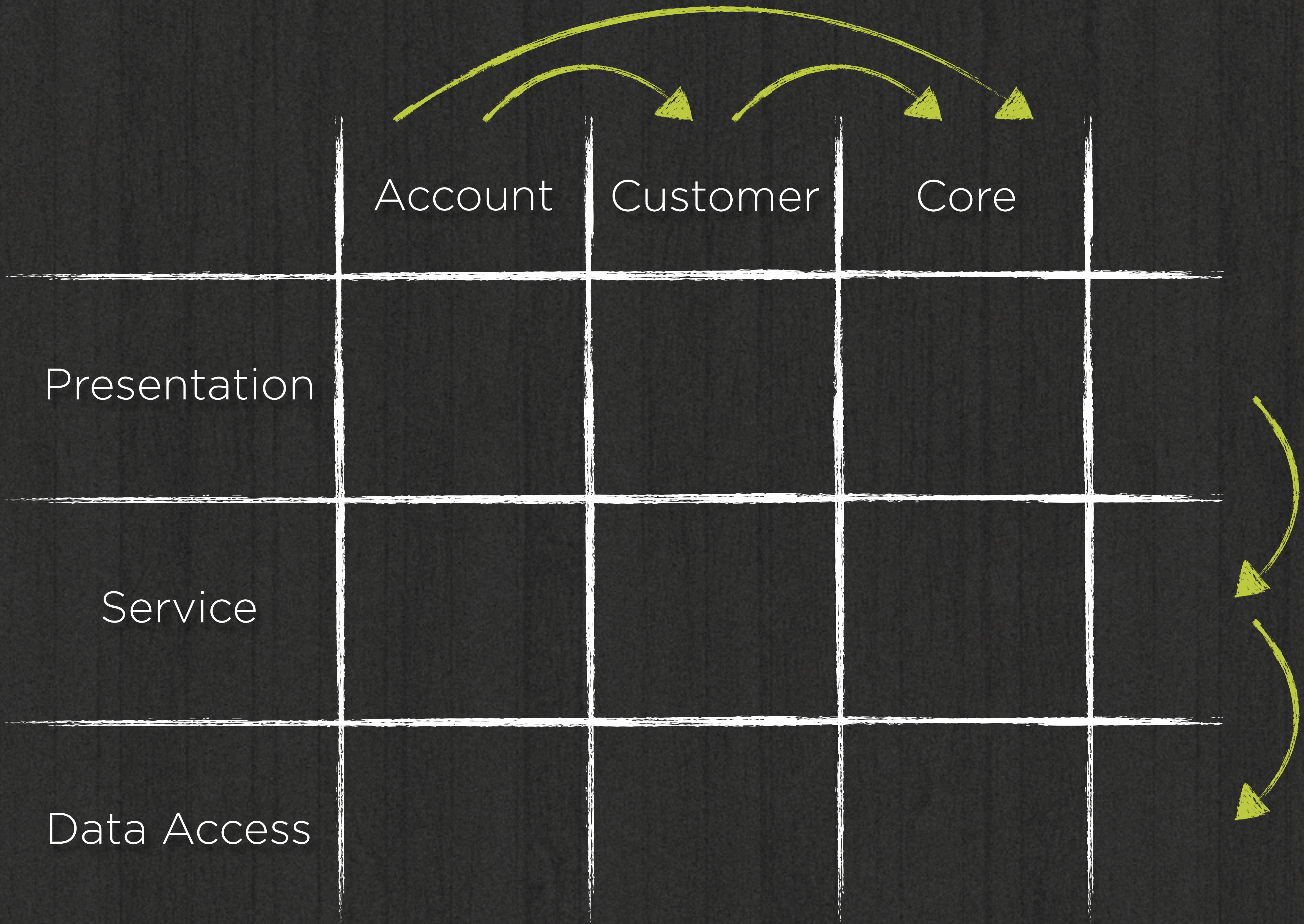
Less important to business

Slices

Hardly understood

New to developers

Key for business requirements



	Account	Customer	Core
Presentation			
Service			
Data Access			

"How to implement an architecture inside a codebase?

Architecture VS. Codebase

" How to implement an architecture inside a codebase?

"How to ~~implement~~ an architecture inside a codebase?

"How to maintain an architecture inside a codebase?

Code analysis

JDepend

Sonar

Sonargraph

Formerly known as SonarJ

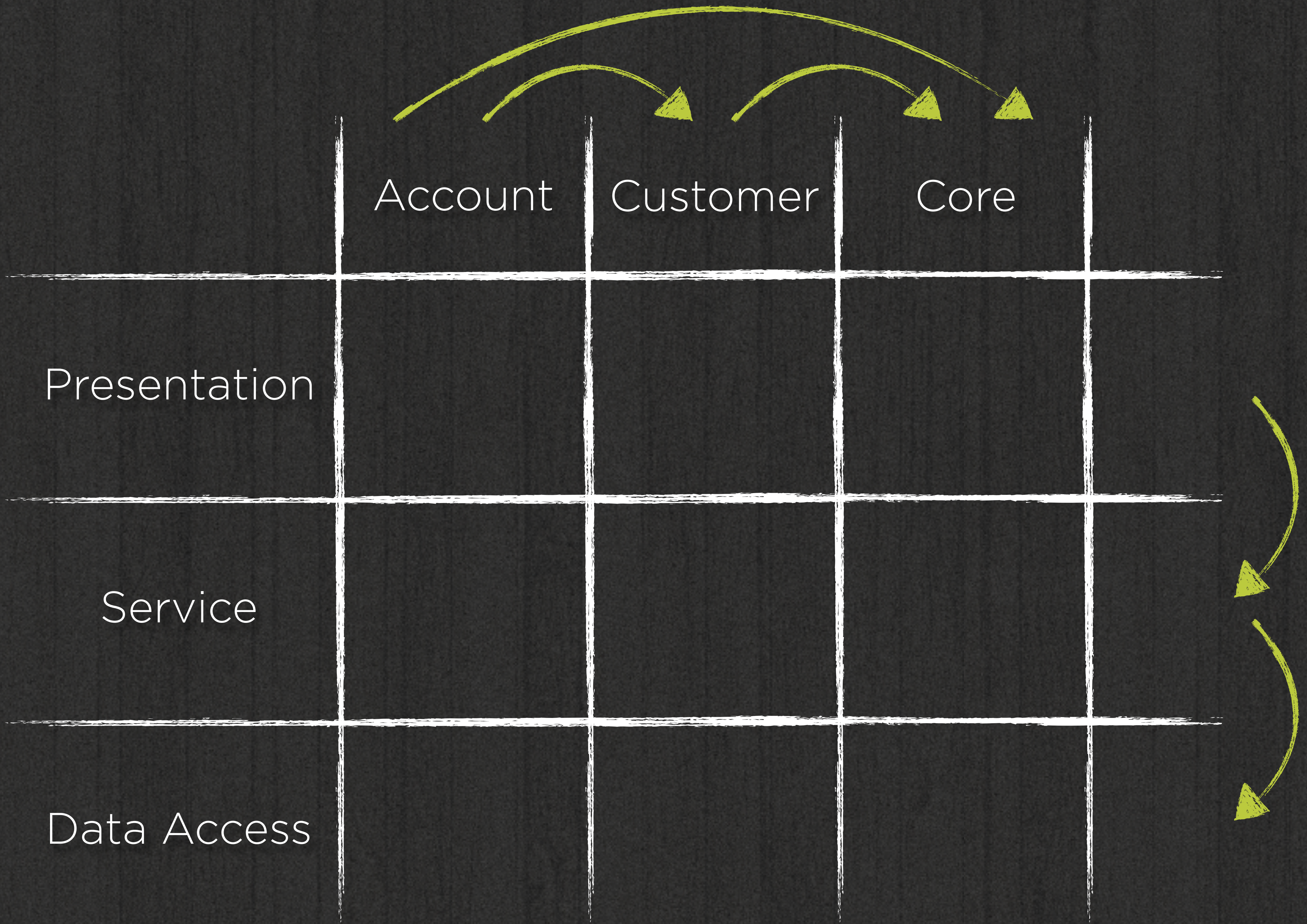
Demo

jQAssistant

http://github.com/buschmais/jqassistant

A plain Java based approach

> ” How far can we get with **plain Java** **means** only?



	Account	Customer	Core
Presentation			
Service			
Data Access			

Packages

```
….layer.slice
….slice.layer
….slice
```

….web.core

….service.core

….repository.core

```
….core.web
….core.service
….core.repository
```

```
….core
….customer
….account
```

" Why the f#$k should I even care?

"Does it make a difference?

Dependency management

" You only need to manage, what you can refer to...

Layers first

Leaks slice internals

Lower layers visible to everyone

Slices first/only

Start with package per slice

Expose interfaces and domain types

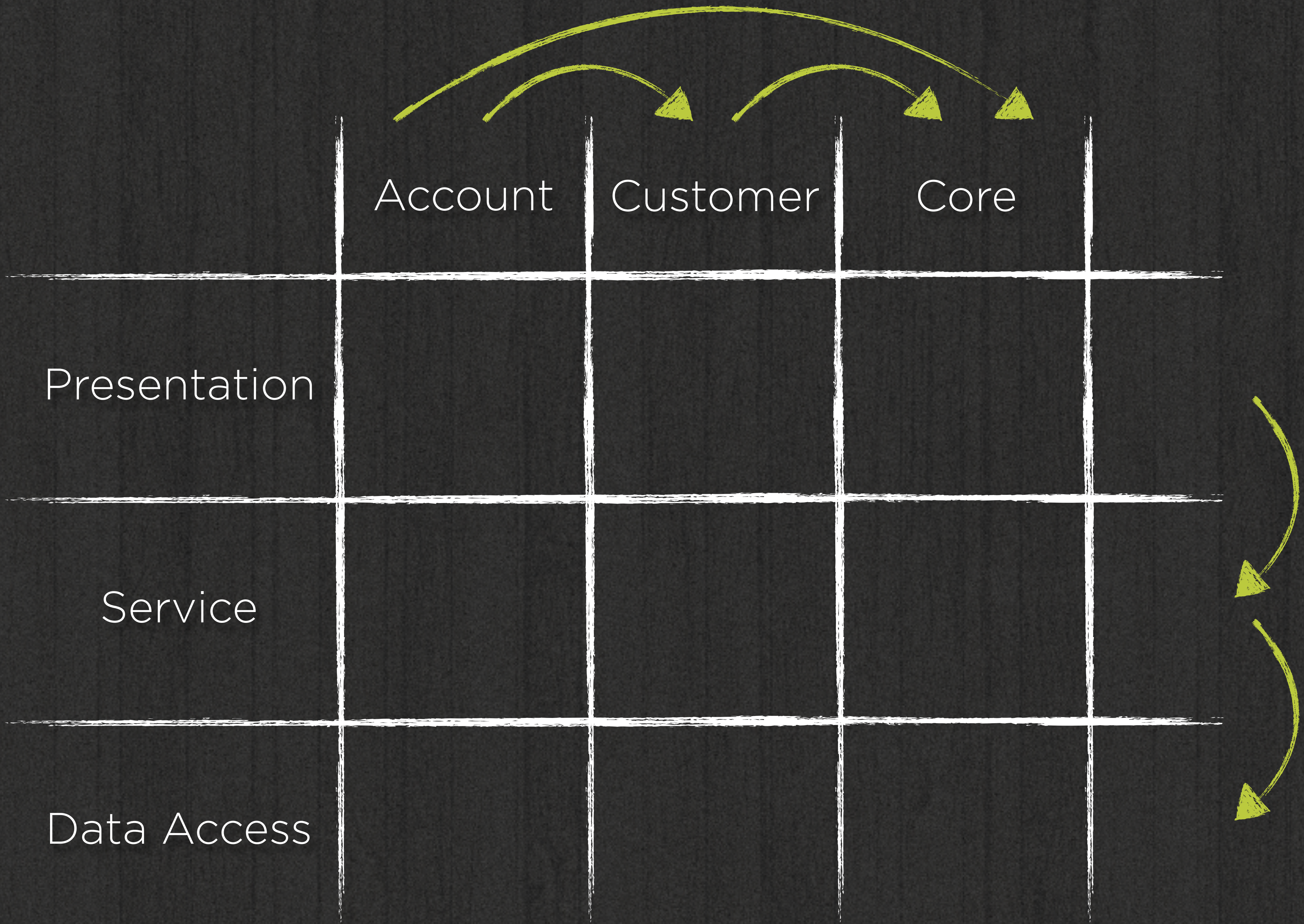
Keep implementations private

Slices first/only

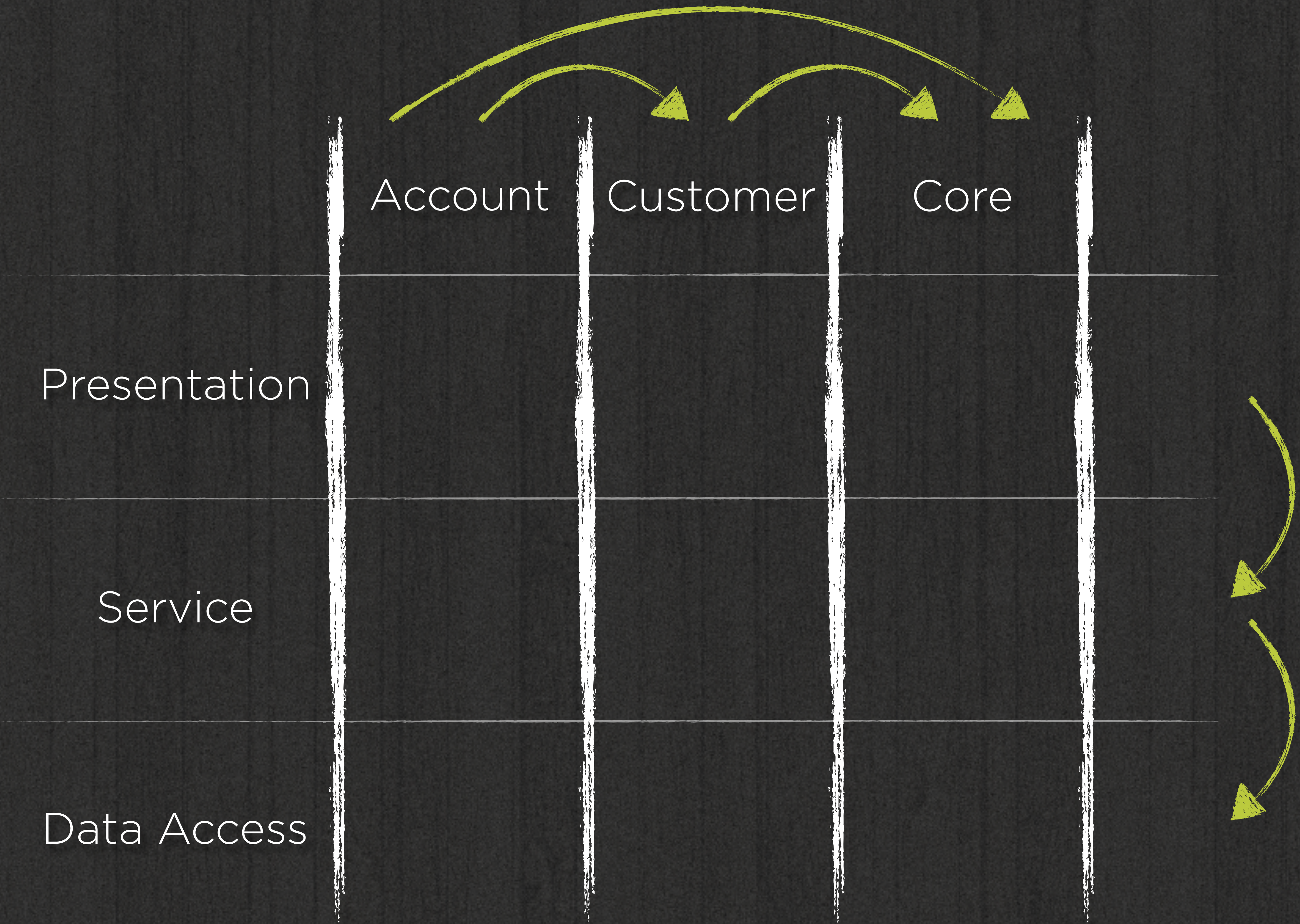
Encapsulates business module

Internals understood anyway

> "Start with **less** packages and the **least visibility** possible...



	Account	Customer	Core
Presentation			
Service			
Data Access			



	Account	Customer	Core
Presentation			
Service			
Data Access			

Demo

Take-aways

Know your dependencies

On every granularity

Start as strict as possible

Get lenient where necessary

Resources

Spring Data JPA @ GitHub

Sonargraph

jQAssistant

Blogpost

Sample Code

http://github.com/olivergierke/whoops-architecture

Oliver Gierke

Spring Data Project Lead

ogierke@pivotal.io

olivergierke

www.olivergierke.de

Thanks & Credits

Eoin Woods - Talk @ InfoQ

Stefan Tilkov - Talk @ InfoQ

Eberhard Wolff - Slides @ Slideshare