betterCode()

CLEAN ARCHITECTURE 2022

ArchUnit

Architektur und Design automatisiert testen

Thomas Much
@thmuch

06.12.2022

ArchUnit

Autor

glücklicher Anwender

zahlreiche „Contributors“

Peter Gafert
@codecholeric
@archtests

Thomas Much
@thmuch
😀

Die Techniker

www.tk.de/IT

Was ist Architektur?

ArchUnit

Entscheidungen, Technologien …
alles „Wichtige“, was schwer zu ändern ist

Struktur und Konventionen innerhalb eines Artefakts (Services)

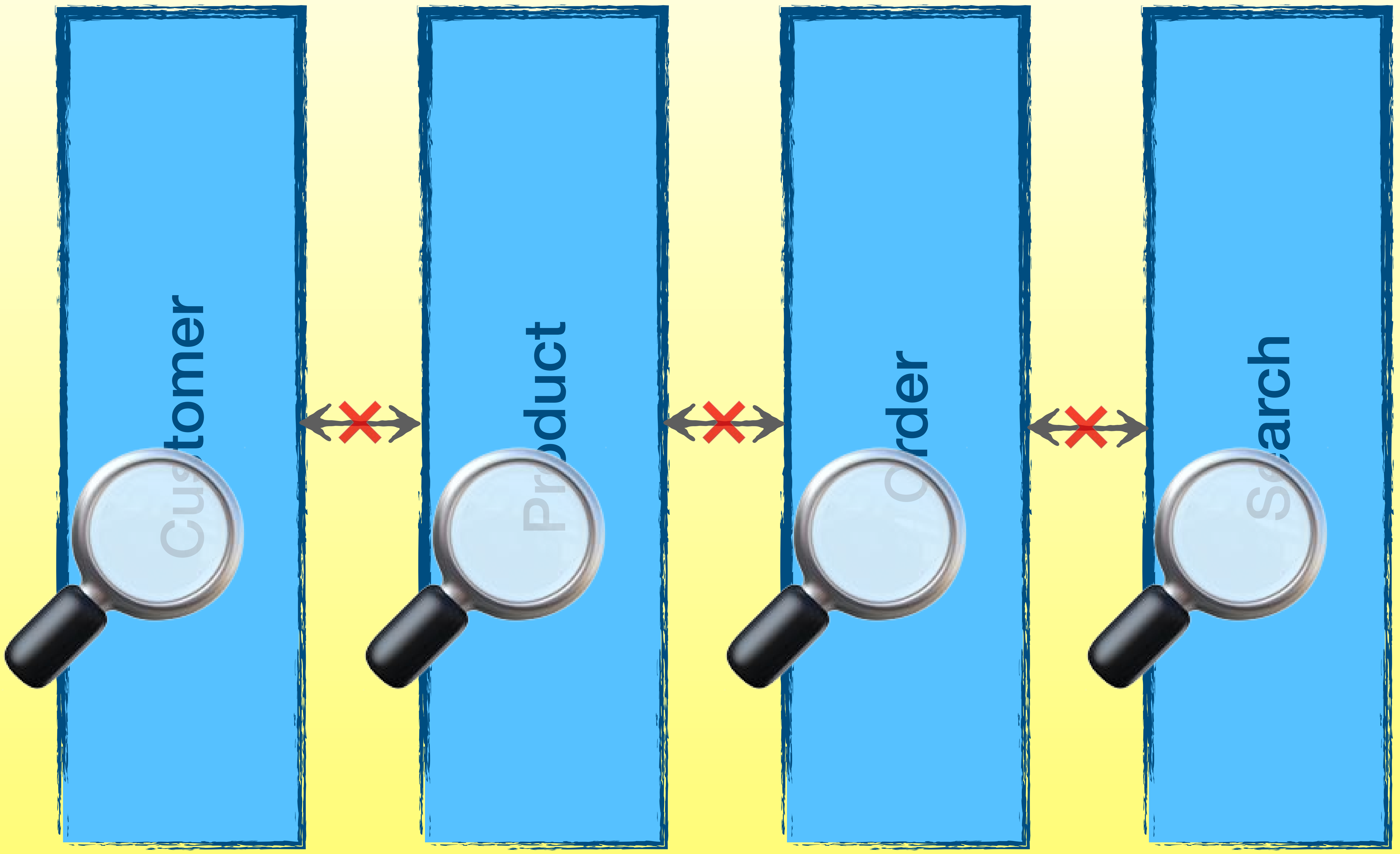
Gemeinsames Verständnis über das System und seine Teile

Kommunikation zwischen den (Teil-)Systemen

…

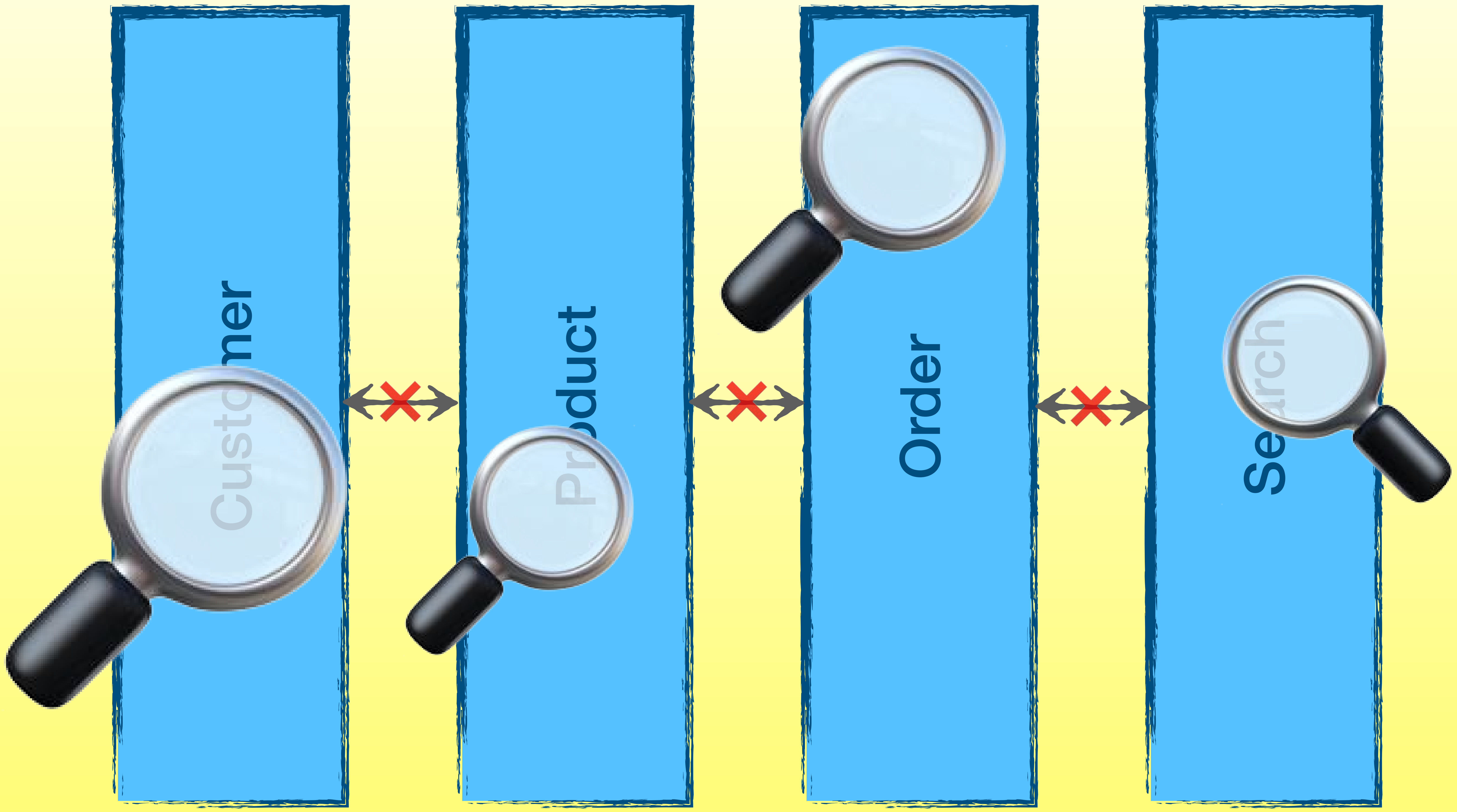
Wo prüfen wir was?

Microservices & SCS*



*) „Self-Contained Systems“, siehe z.B. https://scs-architecture.org/

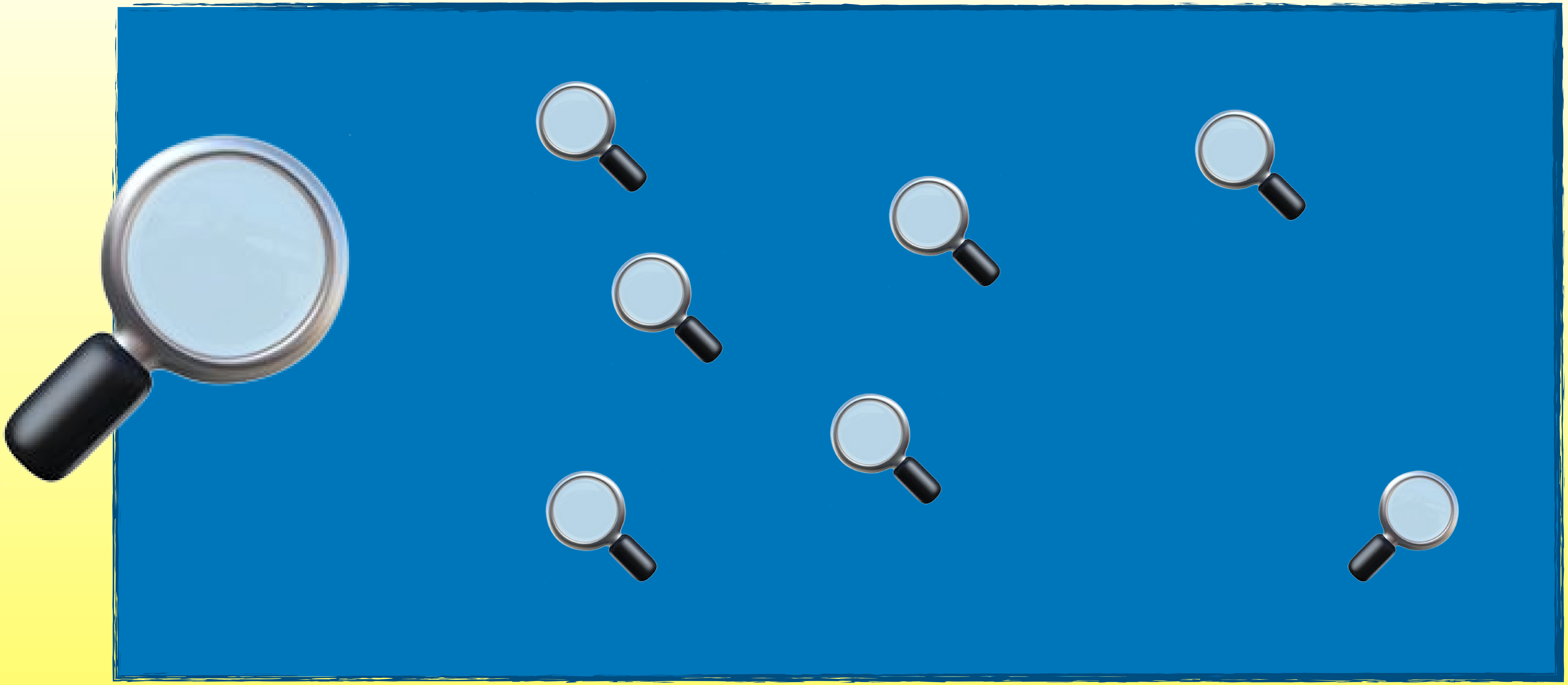
Microservices & SCS*



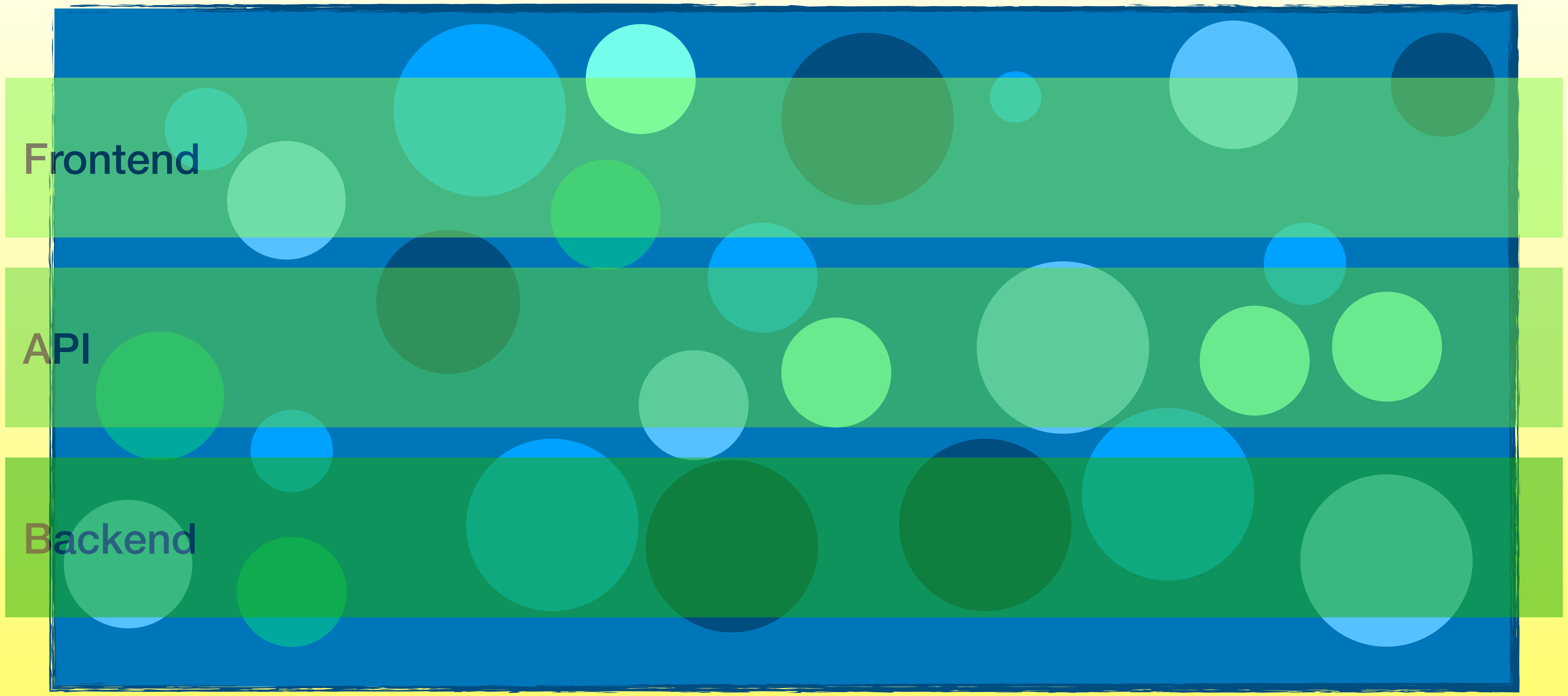
*) „Self-Contained Systems", siehe z.B. https://scs-architecture.org/

Monolithen

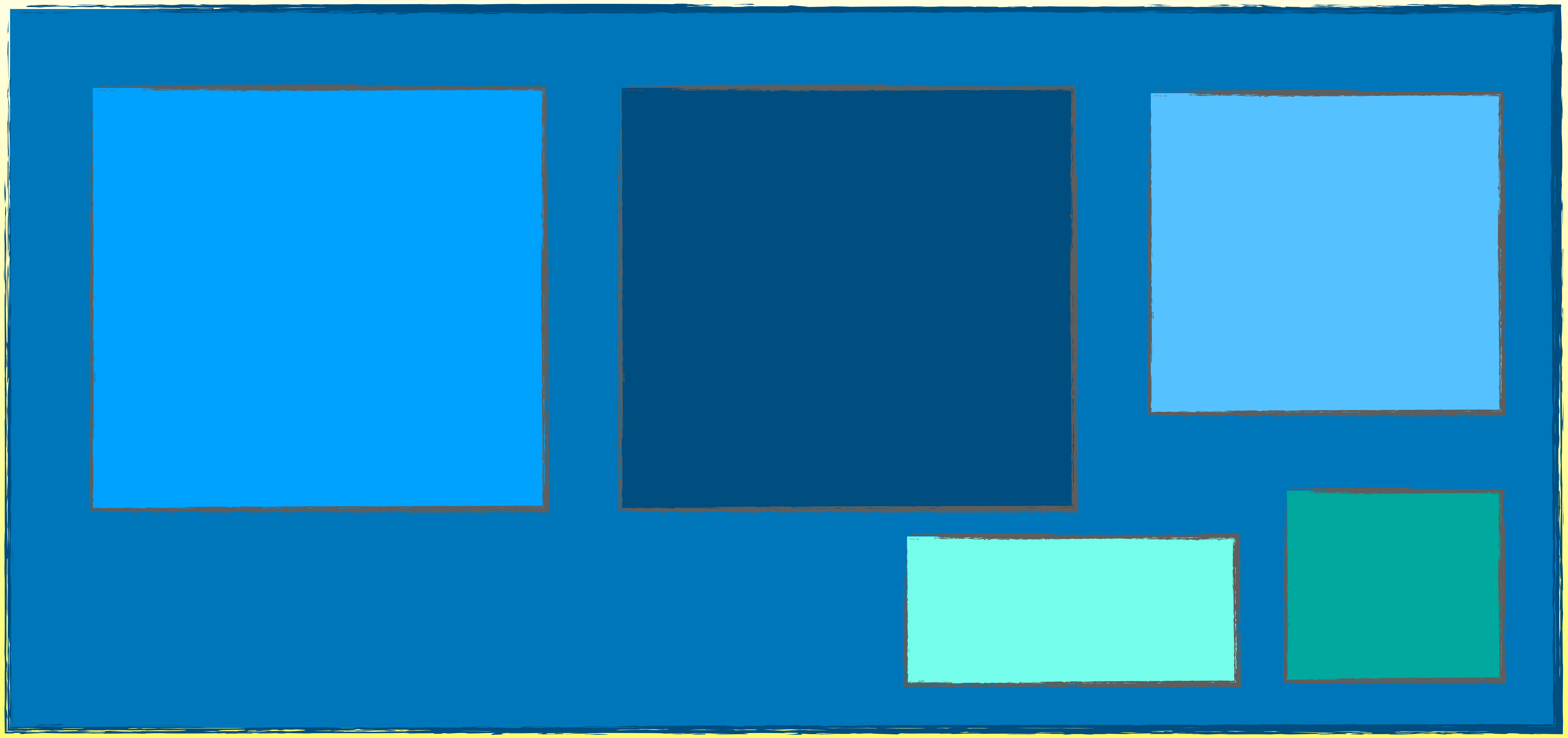
Monolithen



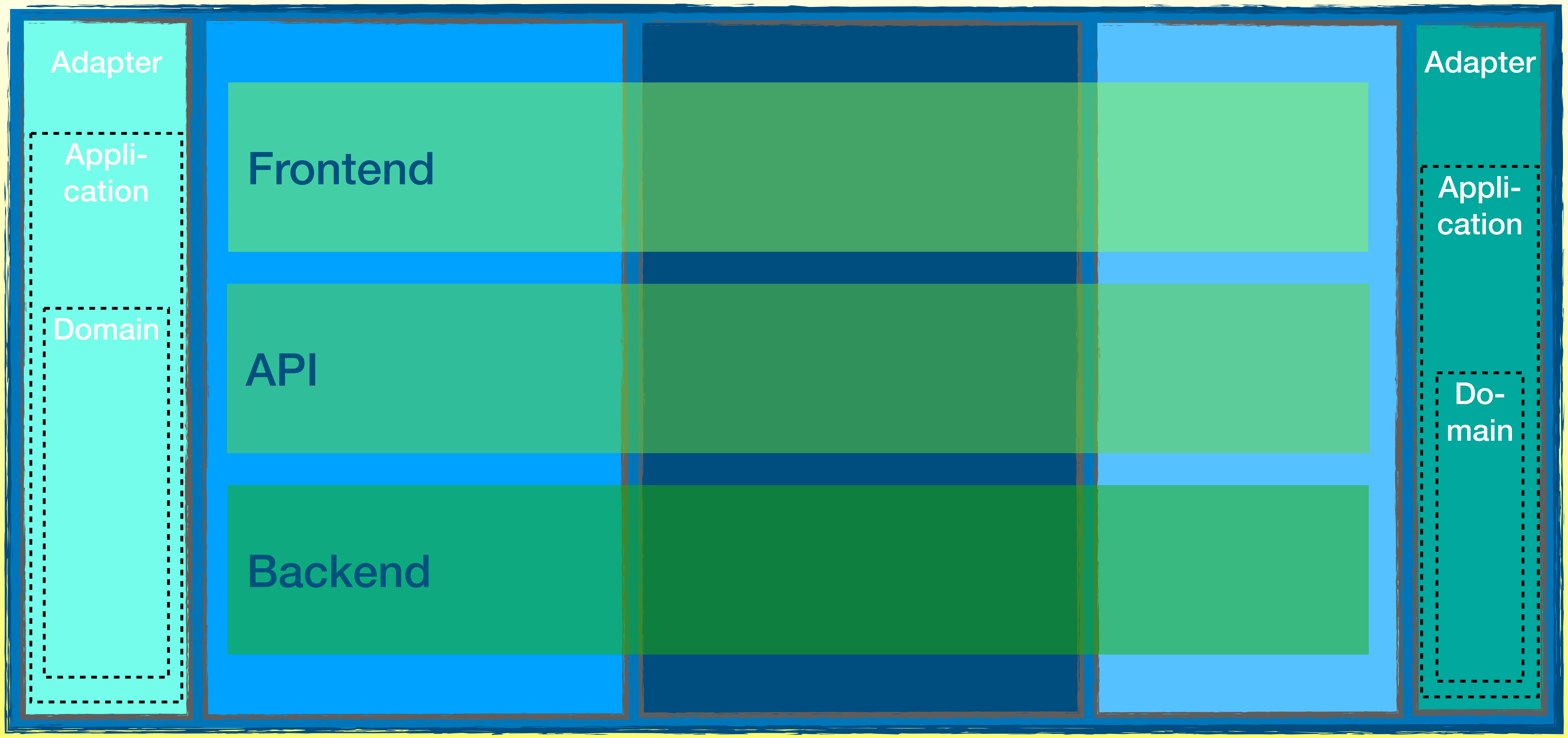
Monolithen



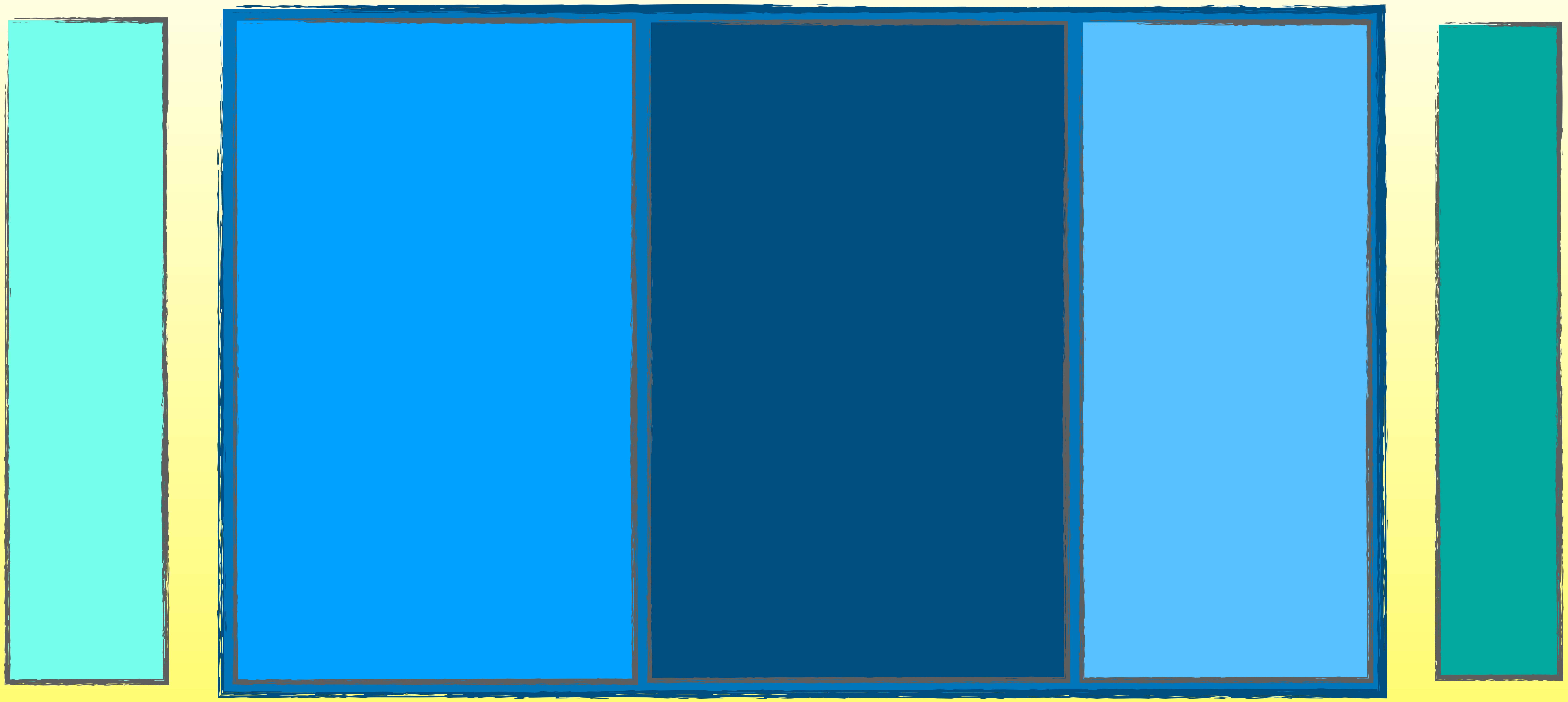
Freundliche Monolithen



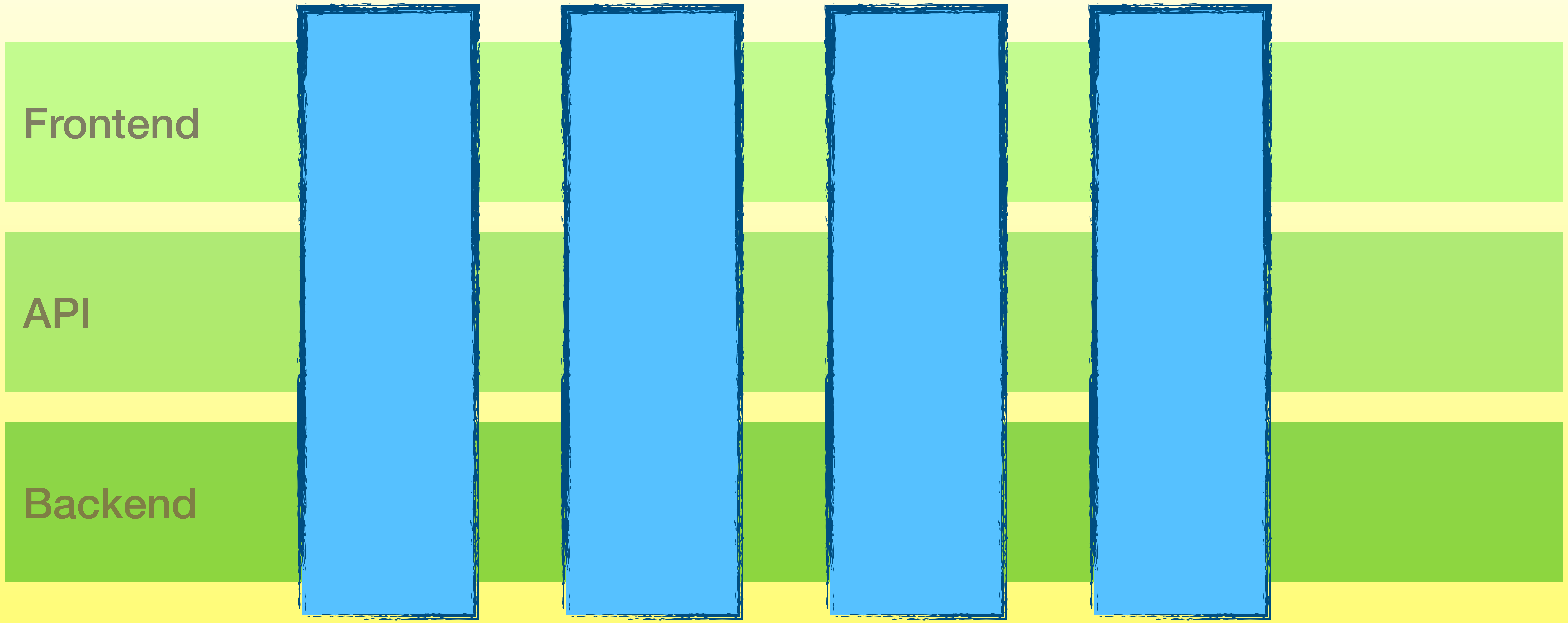
Noch freundlichere Monolithen



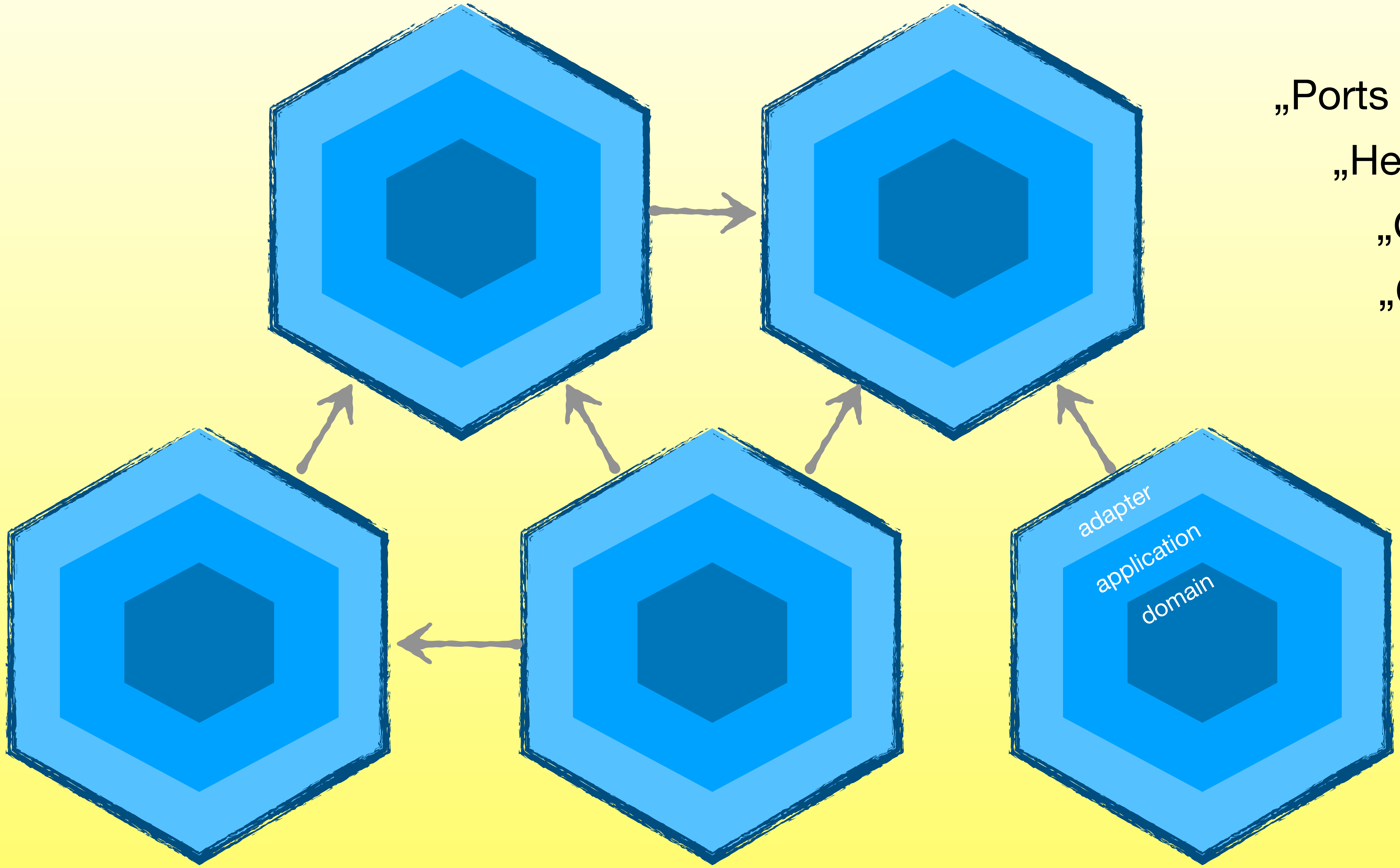
Abgespeckte Monolithen



Was prüfen? Slices & Schichten



Was prüfen? Innen & Außen



„Ports & Adapters“

„Hexagonal“

„Onion“

„Clean“

Architektur testen

Modularisierung
Abhängigkeiten
Kohäsion & Kopplung

Konventionen & Patterns

Mein Weg zu ArchUnit

Oktober 2022:
ArchUnit 1.0 😃

Ende 2017
„ArchUnit 0.4“

Classycle

Checkstyle

möglichst einfach

~~kommerziell~~

ArchUnit

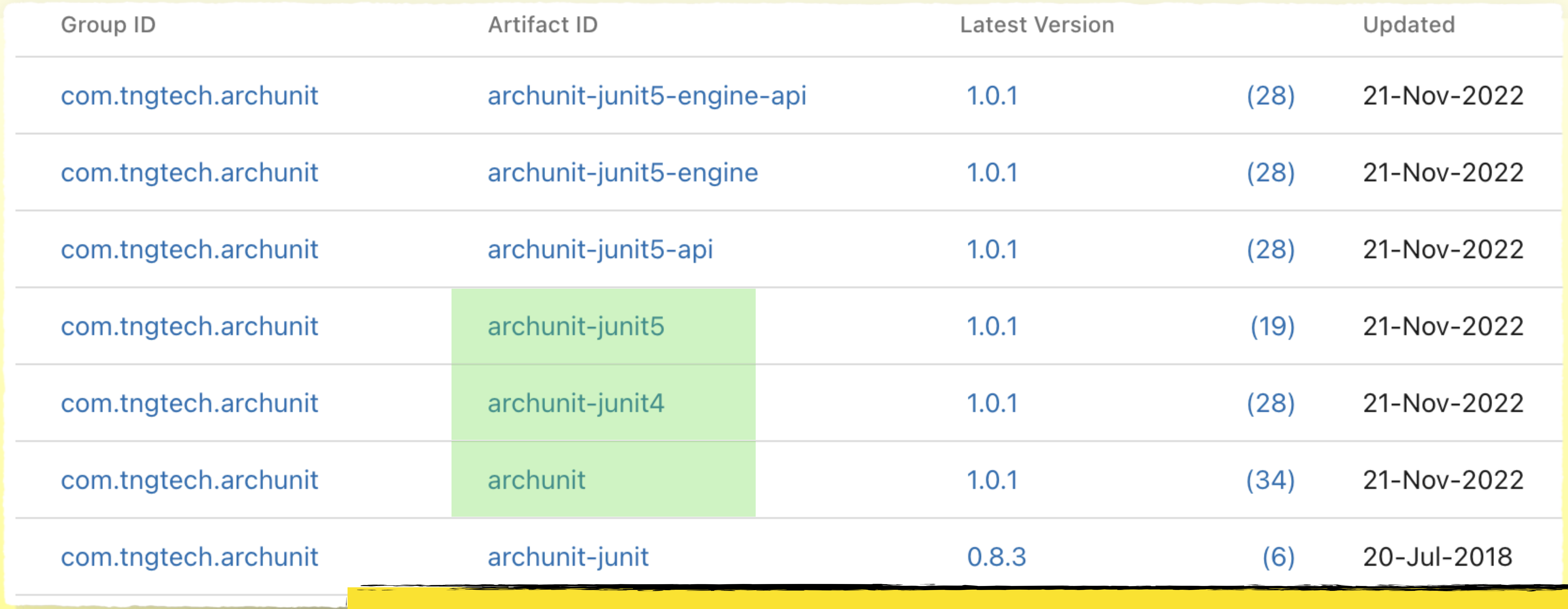
Architektur-Prüfungen als **Unit-Tests**

Normaler **Java-Code**! (oder Kotlin)

Flexibel erweiterbar – auch Design-Prüfungen realisierbar

Prüfung auf **Bytecode**-Ebene

ArchUnit einbinden

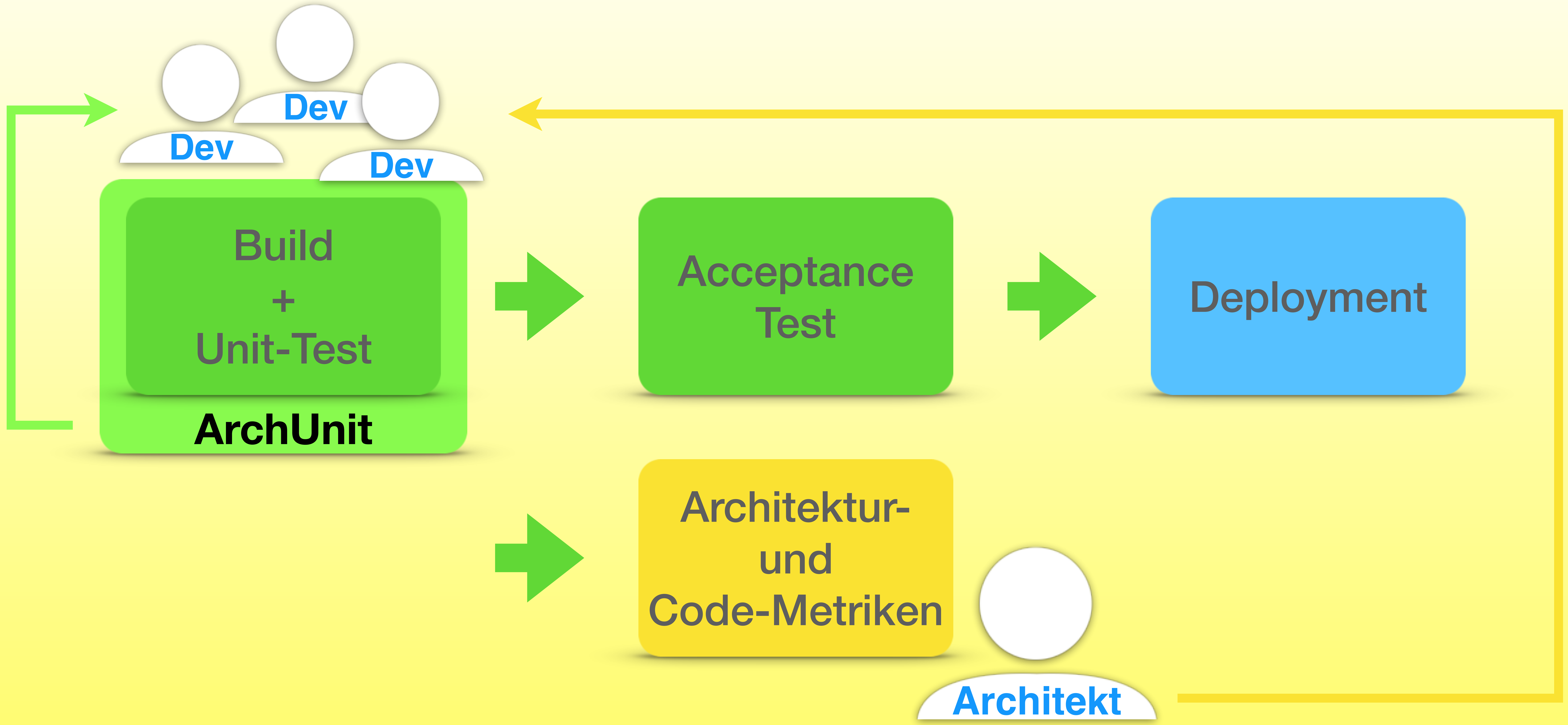


Group ID	Artifact ID	Latest Version		Updated
com.tngtech.archunit	archunit-junit5-engine-api	1.0.1	(28)	21-Nov-2022
com.tngtech.archunit	archunit-junit5-engine	1.0.1	(28)	21-Nov-2022
com.tngtech.archunit	archunit-junit5-api	1.0.1	(28)	21-Nov-2022
com.tngtech.archunit	archunit-junit5	1.0.1	(19)	21-Nov-2022
com.tngtech.archunit	archunit-junit4	1.0.1	(28)	21-Nov-2022
com.tngtech.archunit	archunit	1.0.1	(34)	21-Nov-2022
com.tngtech.archunit	archunit-junit	0.8.3	(6)	20-Jul-2018

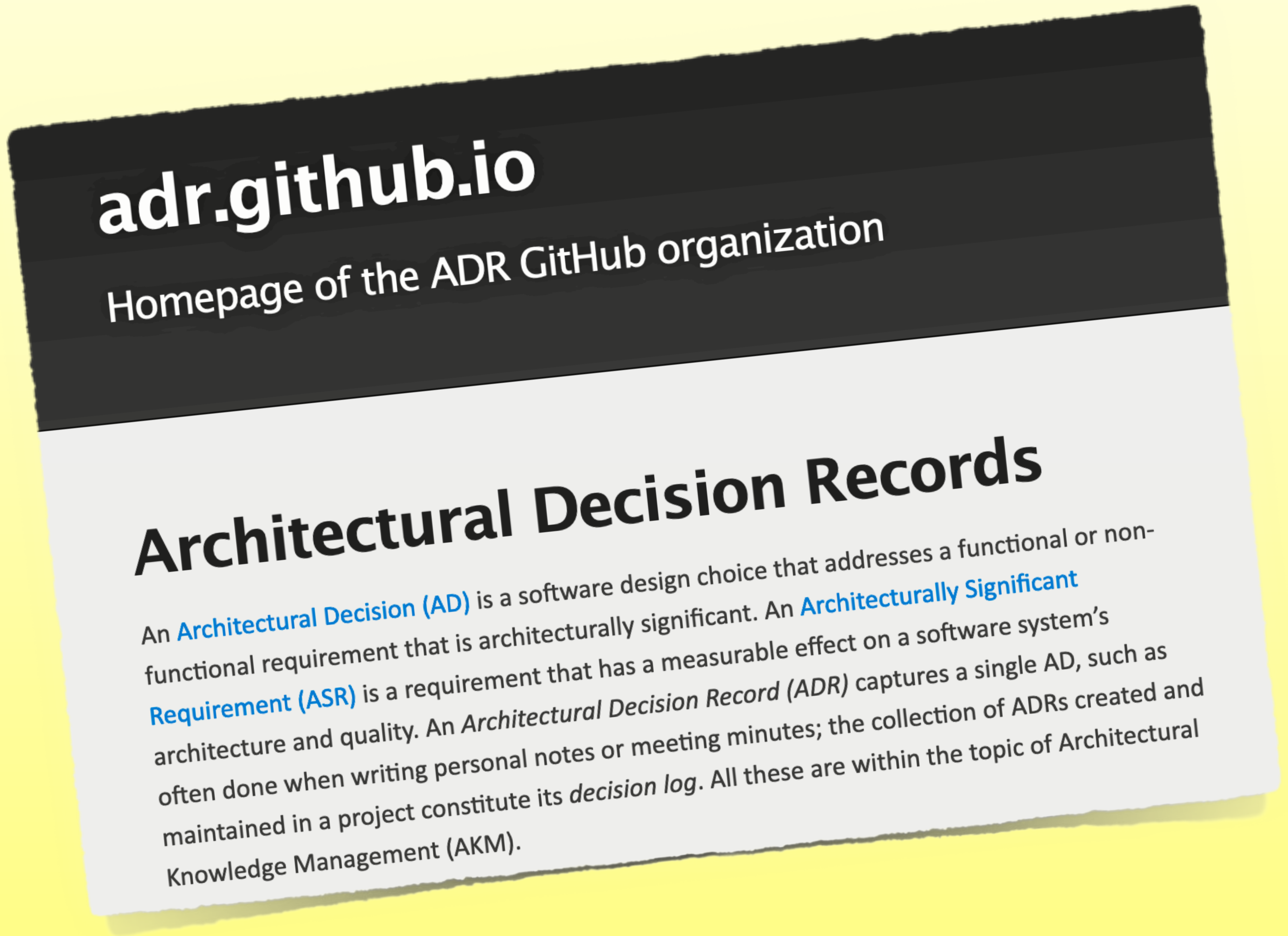
```xml
<dependency>
    <groupId>com.tngtech.archunit</groupId>
    <artifactId>archunit-junit5</artifactId>
    <version>1.0.1</version>
    <scope>test</scope>
</dependency>
```

Live-Demo

Warum ArchUnit?



Architektur-Entscheidungen als Team

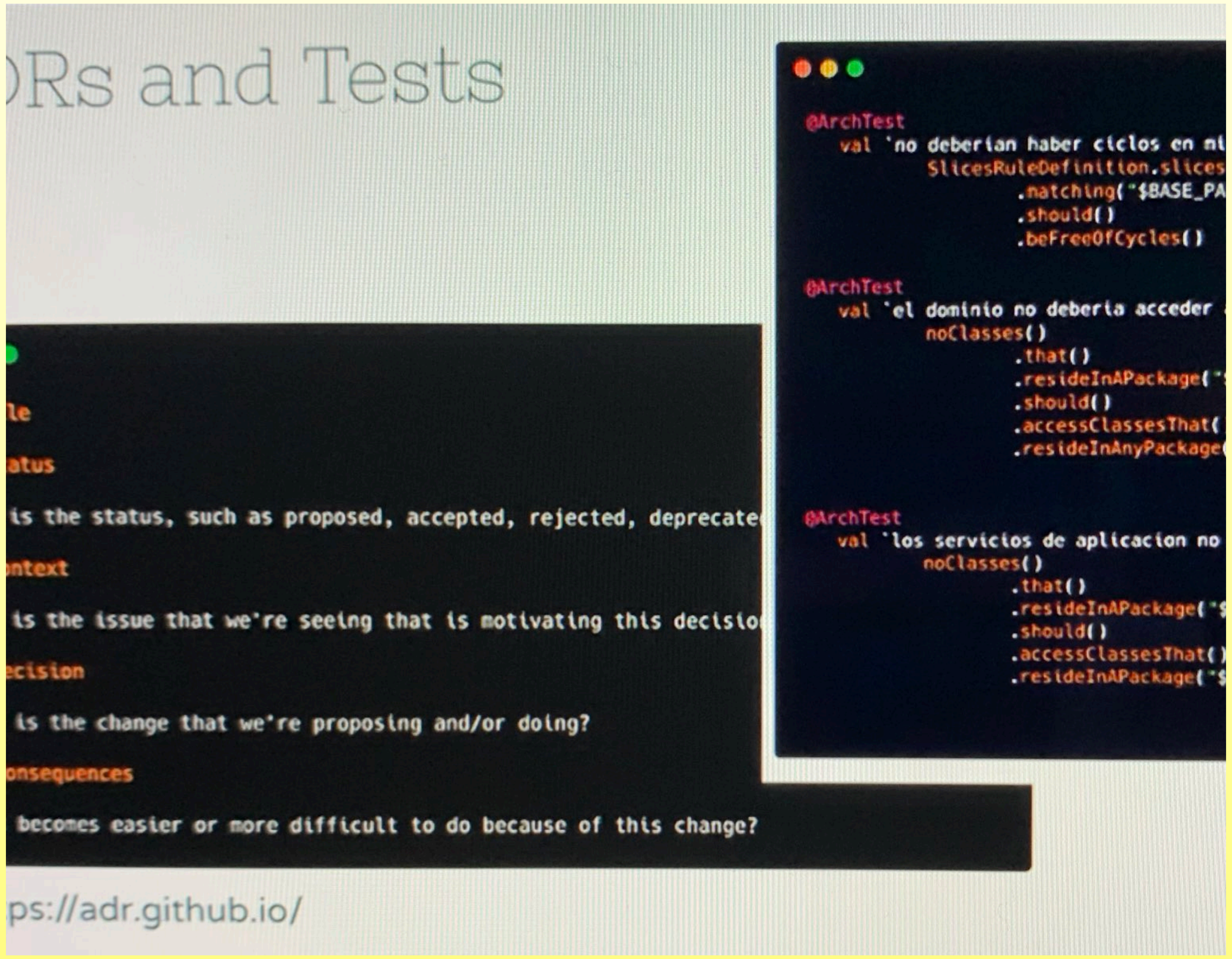


adr.github.io

Homepage of the ADR GitHub organization

Architectural Decision Records

An Architectural Decision (AD) is a software design choice that addresses a functional or non-functional requirement that is architecturally significant. An Architecturally Significant Requirement (ASR) is a requirement that has a measurable effect on a software system's architecture and quality. An *Architectural Decision Record (ADR)* captures a single AD, such as often done when writing personal notes or meeting minutes; the collection of ADRs created and maintained in a project constitute its *decision log*. All these are within the topic of Architectural Knowledge Management (AKM).



Rs and Tests

```
le
atus
 is the status, such as proposed, accepted, rejected, deprecate
ntext
 is the issue that we're seeing that is motivating this decisio
ecision
 is the change that we're proposing and/or doing?
onsequences
 becomes easier or more difficult to do because of this change?
```

```
@ArchTest
   val `no deberian haber ciclos en mi
            SlicesRuleDefinition.slices
                   .matching("$BASE_PA
                   .should()
                   .beFreeOfCycles()

@ArchTest
   val `el dominio no deberia acceder
            noClasses()
                   .that()
                   .resideInAPackage("
                   .should()
                   .accessClassesThat(
                   .resideInAnyPackage(

@ArchTest
   val `los servicios de aplicacion no
            noClasses()
                   .that()
                   .resideInAPackage("$
                   .should()
                   .accessClassesThat()
                   .resideInAPackage("$
```

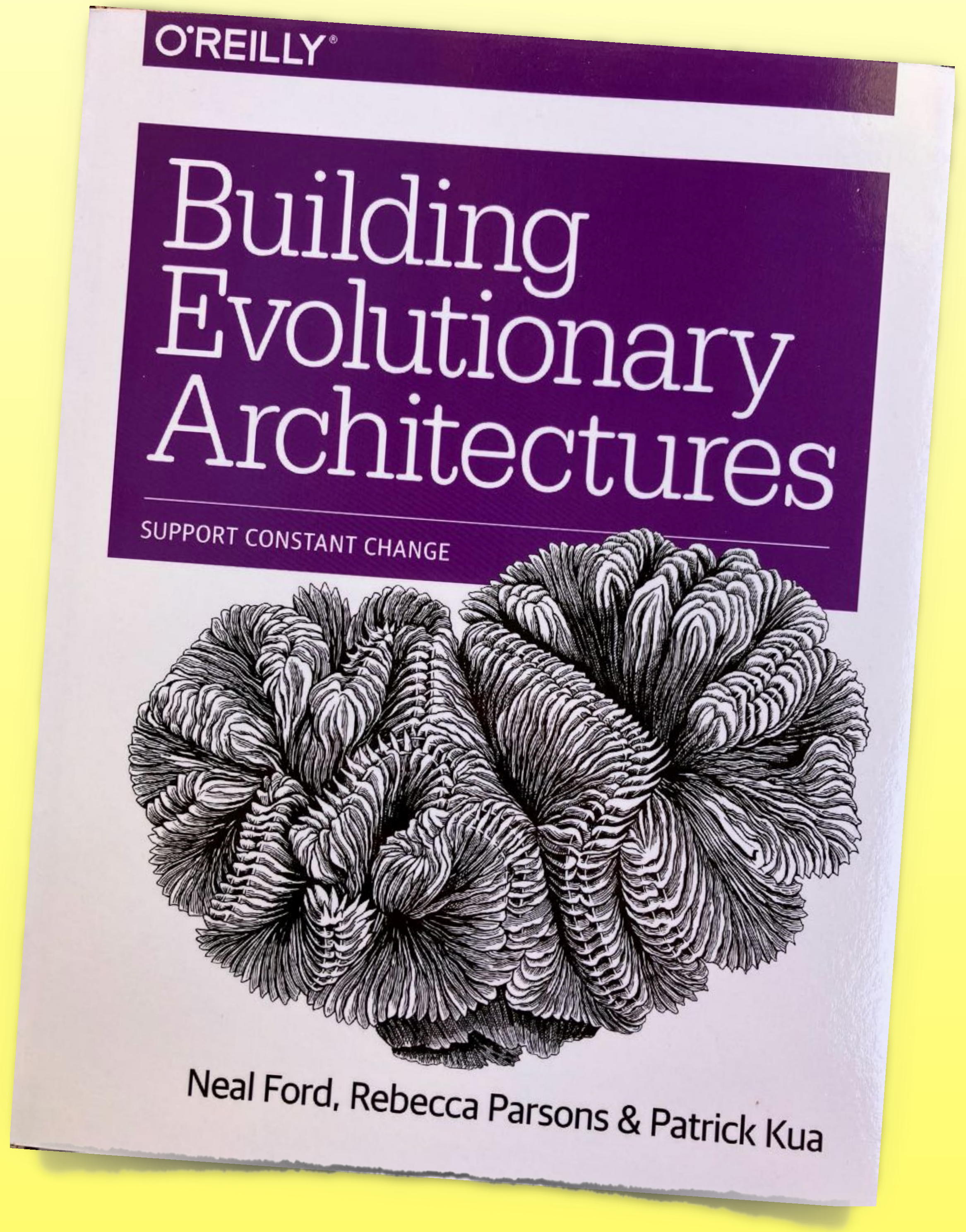
ps://adr.github.io/

@JavitaLaso beim @ddd_eu: https://twitter.com/gtielsch/status/1357716900084674562

Evolutionäre Architektur

Was ist eine wichtige Architektur-Eigenschaft?
Änderbarkeit!

ArchUnit-Prüfregeln als
Fitness Functions
für eine evolutionäre Architektur


O'REILLY
Building Evolutionary Architectures
SUPPORT CONSTANT CHANGE
Neal Ford, Rebecca Parsons & Patrick Kua

Alternativen & Ergänzungen

jQAssistant

structure101

sonarqube

checkstyle

SONARGRAPH

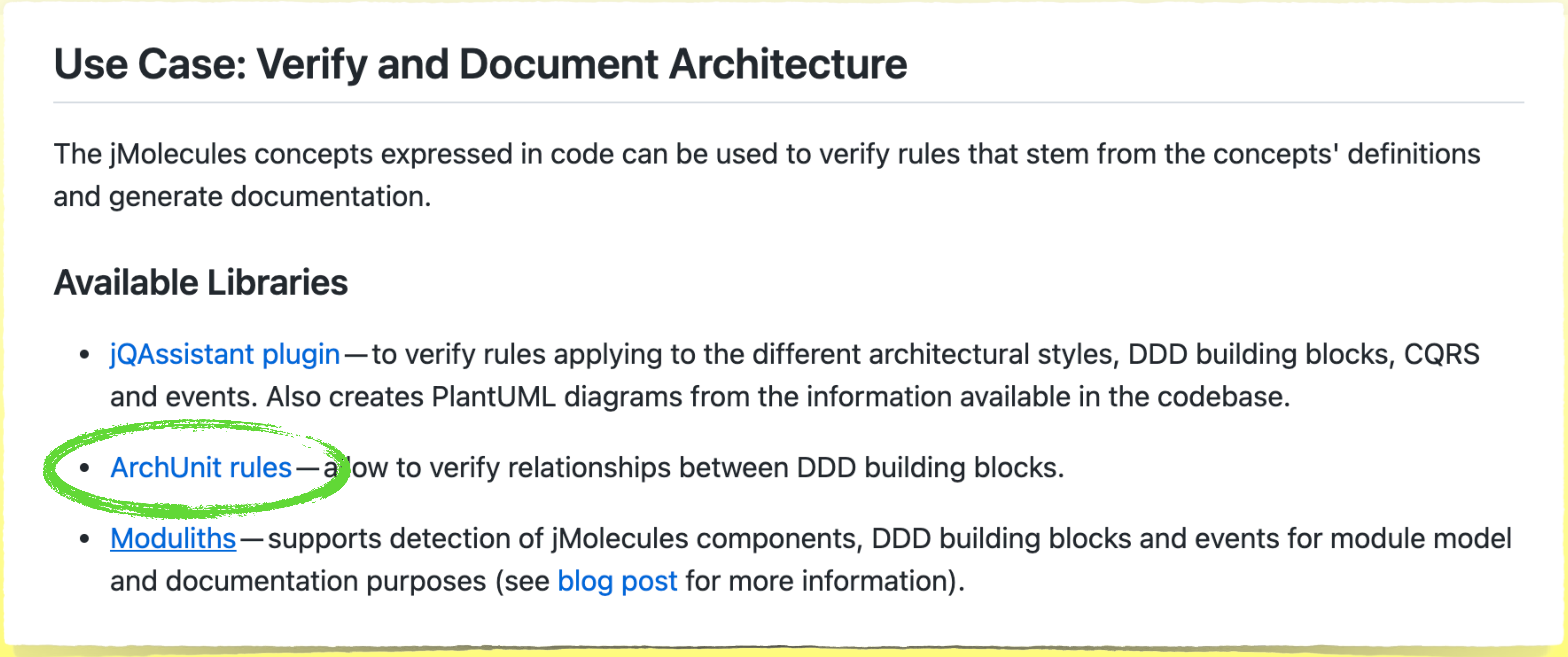
u.v.a.m.

Spring Moduliths

The verification as well as the underlying analysis of the application module model are implemented by using ArchUnit. It will reject cyclic dependencies between application modules, access to types considered internal (as per the definition above), and, optionally, allow only references to modules explicitly allow-listed by using `@ApplicationModule(allowedDependencies = …)` on the application modules `package-info.java`. For more information on how to define application module boundaries and allowed dependencies between them in the link, see the reference documentation.

https://spring.io/blog/2022/10/21/introducing-spring-modulith

jMolecules



Use Case: Verify and Document Architecture

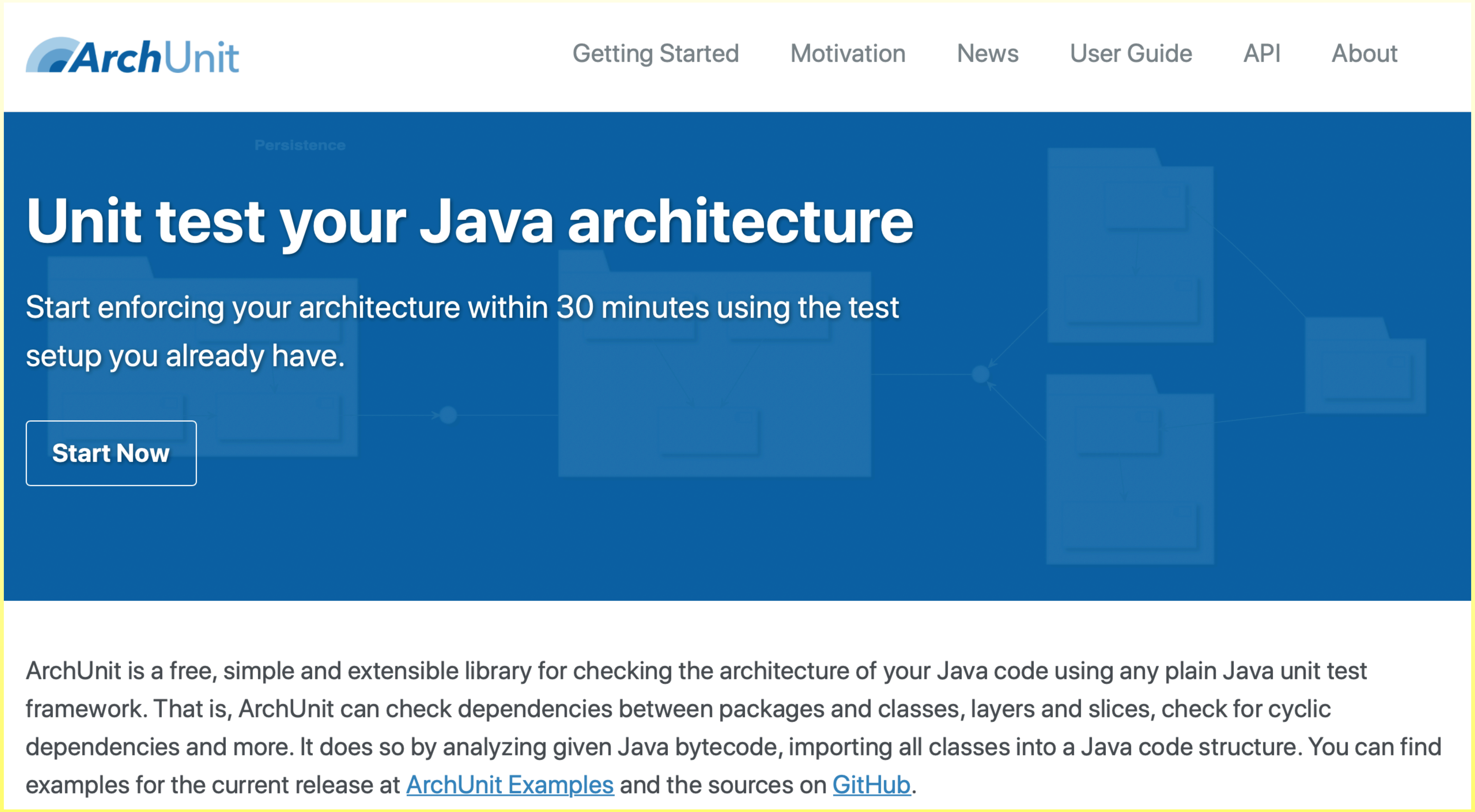
The jMolecules concepts expressed in code can be used to verify rules that stem from the concepts' definitions and generate documentation.

Available Libraries

- jQAssistant plugin — to verify rules applying to the different architectural styles, DDD building blocks, CQRS and events. Also creates PlantUML diagrams from the information available in the codebase.
- ArchUnit rules — allow to verify relationships between DDD building blocks.
- Moduliths — supports detection of jMolecules components, DDD building blocks and events for module model and documentation purposes (see blog post for more information).

https://github.com/xmolecules/jmolecules

www.archunit.org



Unit test your Java architecture

Start enforcing your architecture within 30 minutes using the test setup you already have.

ArchUnit is a free, simple and extensible library for checking the architecture of your Java code using any plain Java unit test framework. That is, ArchUnit can check dependencies between packages and classes, layers and slices, check for cyclic dependencies and more. It does so by analyzing given Java bytecode, importing all classes into a Java code structure. You can find examples for the current release at ArchUnit Examples and the sources on GitHub.

ArchUnit

*https://github.com/**TNG**/ArchUnit-Examples*

*https://github.com/**thmuch**/archunit-demos*

ArchUnit

Schichten

Architektur

Design

Slices

Konventionen

Vertikalen

Fragen?

Monolithen

Module

SCS

Abhängigkeiten

Microservices

@thmuch

Kopplung

Kohäsion

betterCode()

CLEAN ARCHITECTURE 2022

ArchUnit

Danke!

☺️

Thomas Much
@thmuch

Die Techniker

www.tk.de/IT