Orientation in Objects

BRAINTIME

– It's time to rethink –

Datenbank-Design in der schönen neuen Welt

Java User Group Kaiserslautern 2016

Orientation in Objects GmbH

Weinheimer Str. 68
68309 Mannheim

www.oio.de
info@oio.de

Version: 1.2

Ihr Sprecher

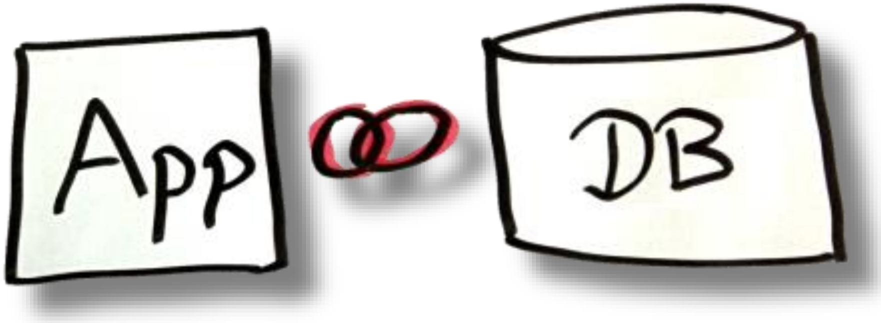
Thorsten Maier

Trainer, Berater, Entwickler

Schwerpunkte
Prozesse
Architektur
Code-Qualität
*eigentlich **kein** „Datenbänker“*

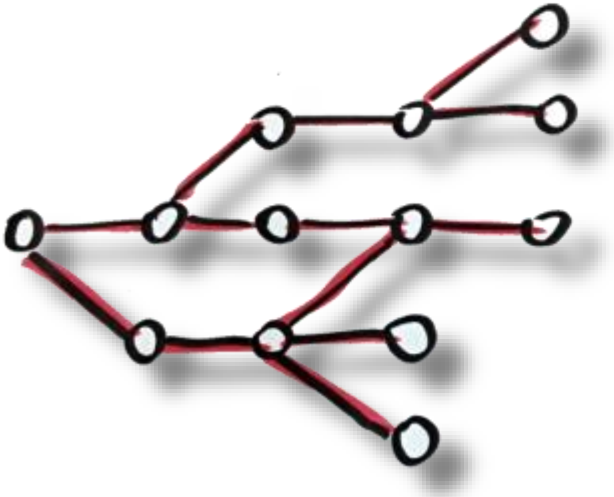
@ThorstenMaier

Um was geht's?



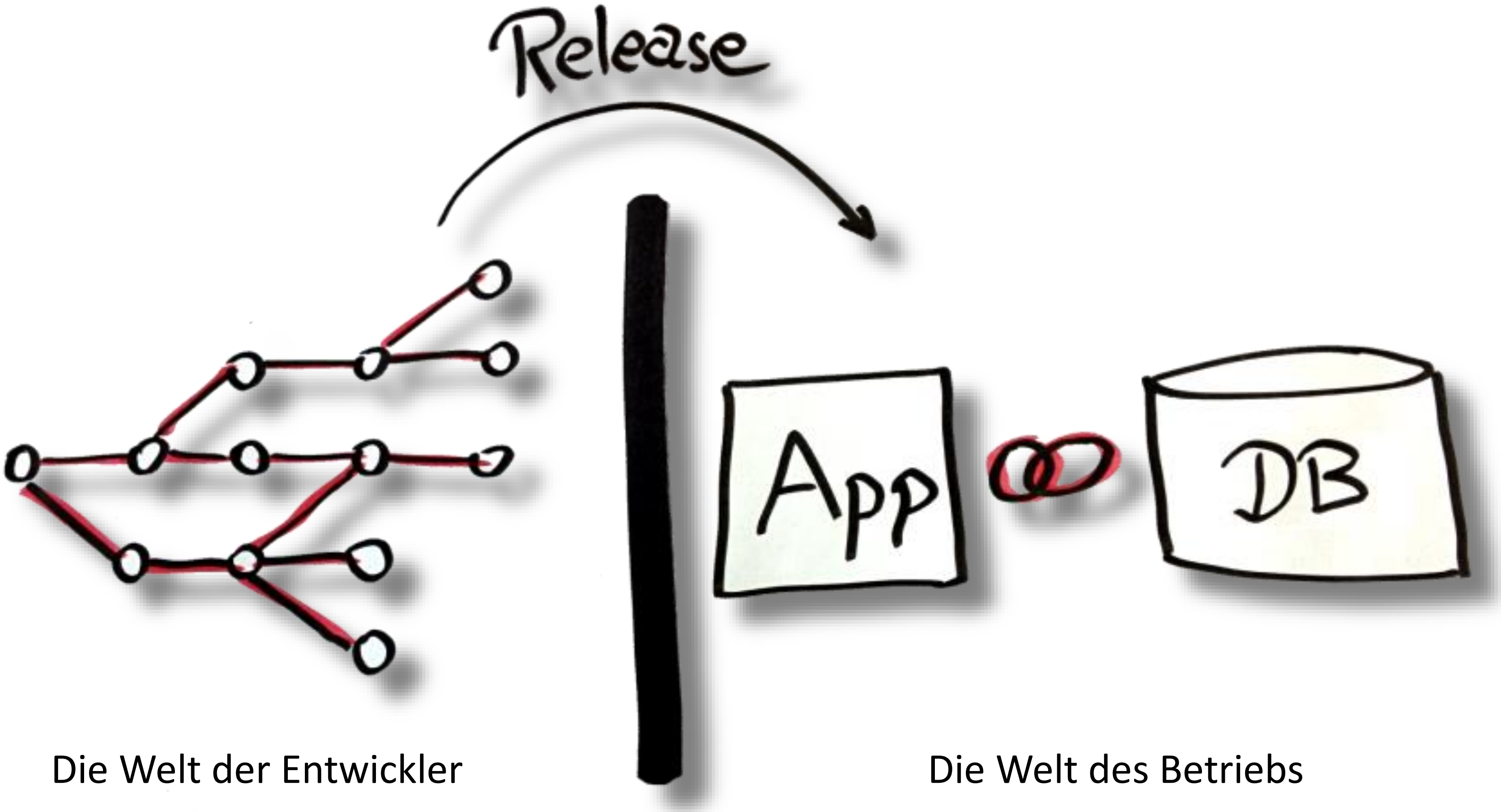
Die Welt des Betriebs

Um was geht's?



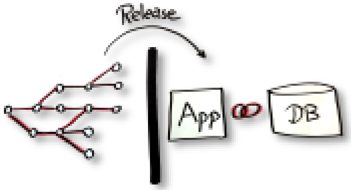
Die Welt der Entwickler

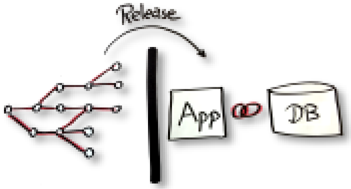
Um was geht's?

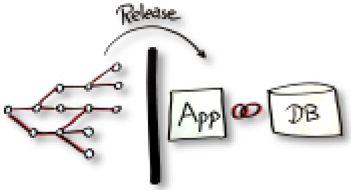


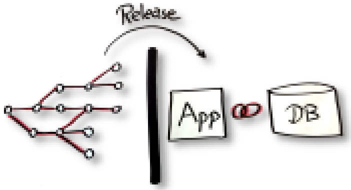
Die Welt der Entwickler

Die Welt des Betriebs



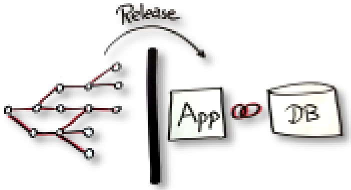


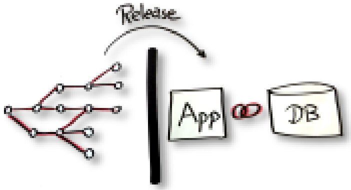


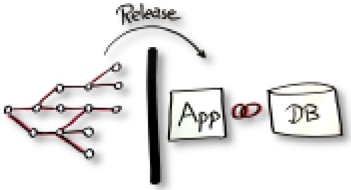


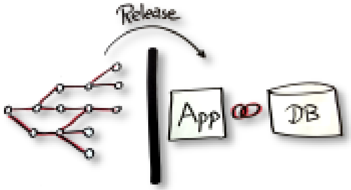
Die schöne neue Welt

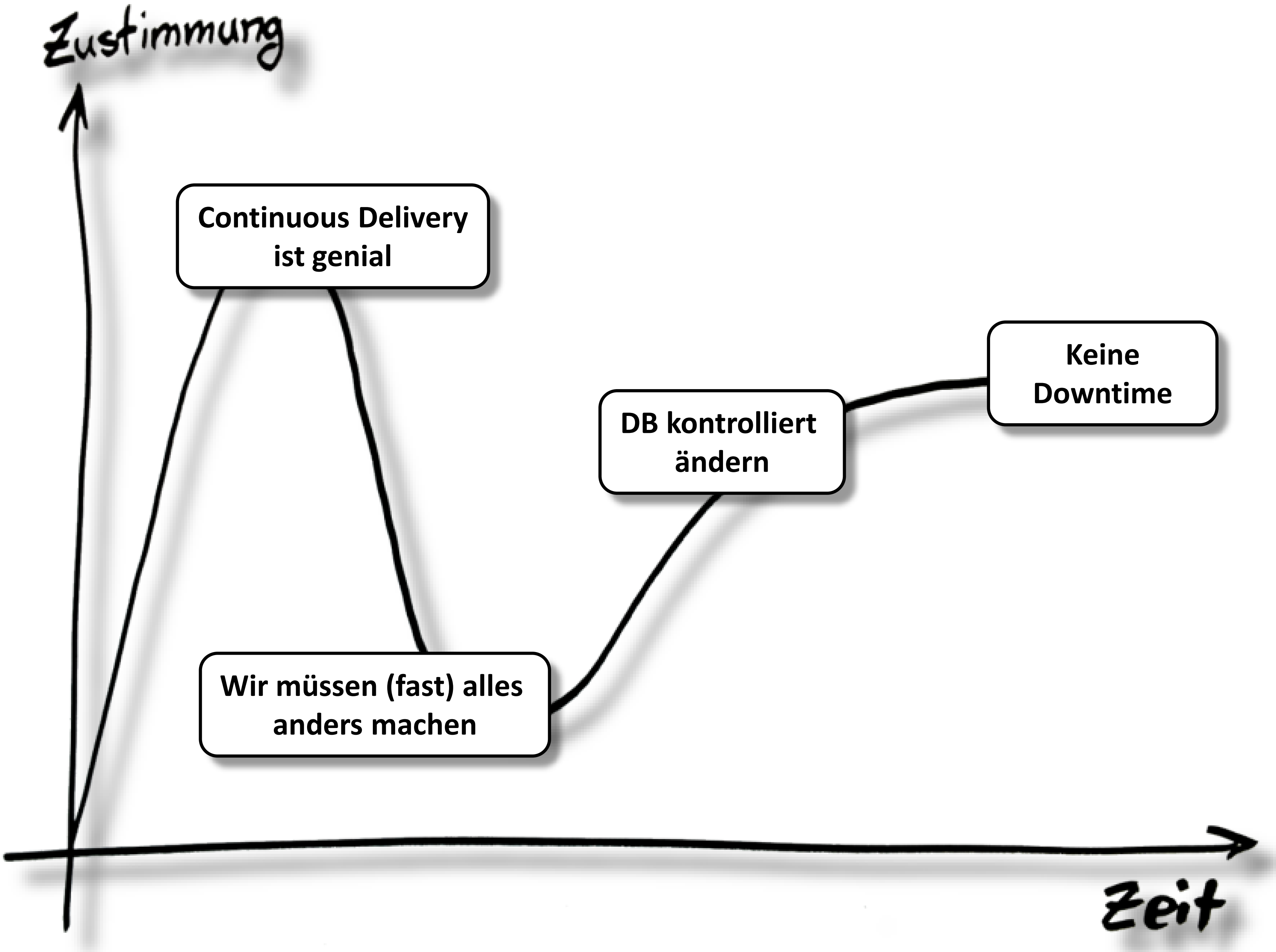
Continuous Delivery











Continuous Delivery ist genial

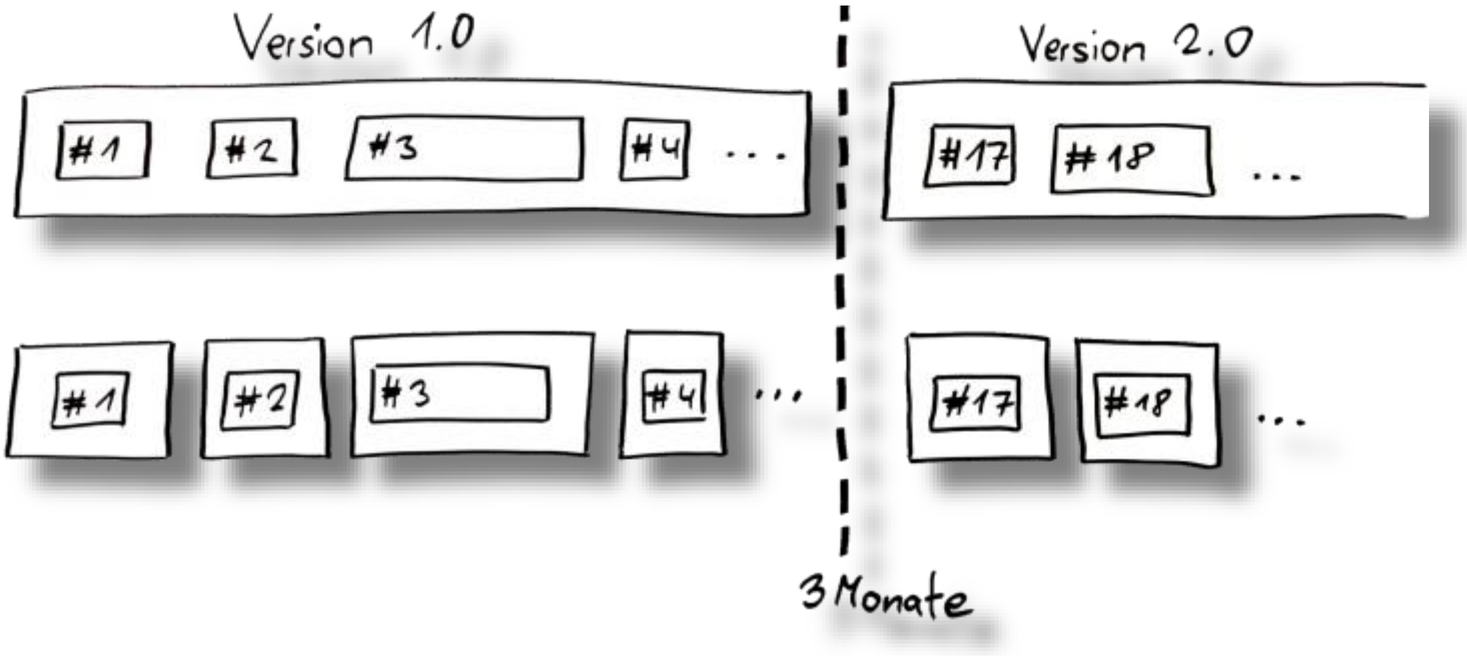
Wir müssen (fast) alles anders machen

DB kontrolliert ändern

Keine Downtime

3 Monats-Releases

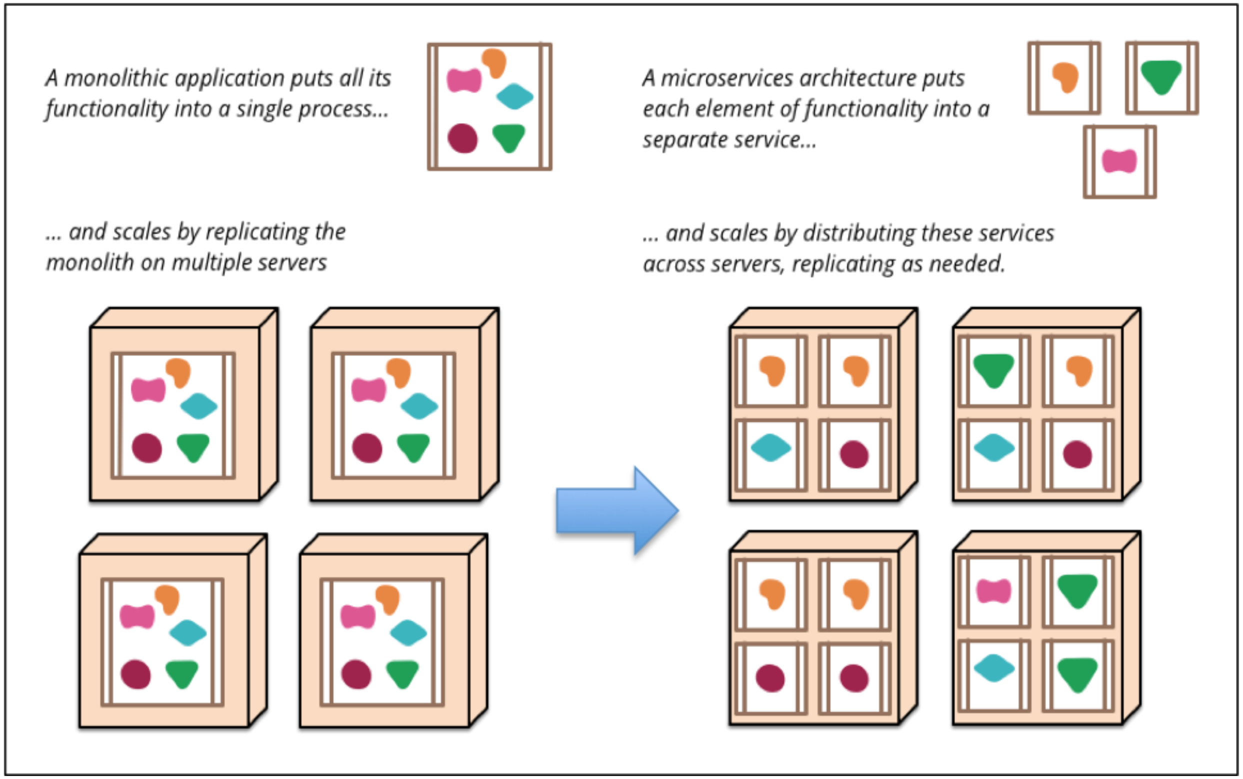
Continuous Delivery



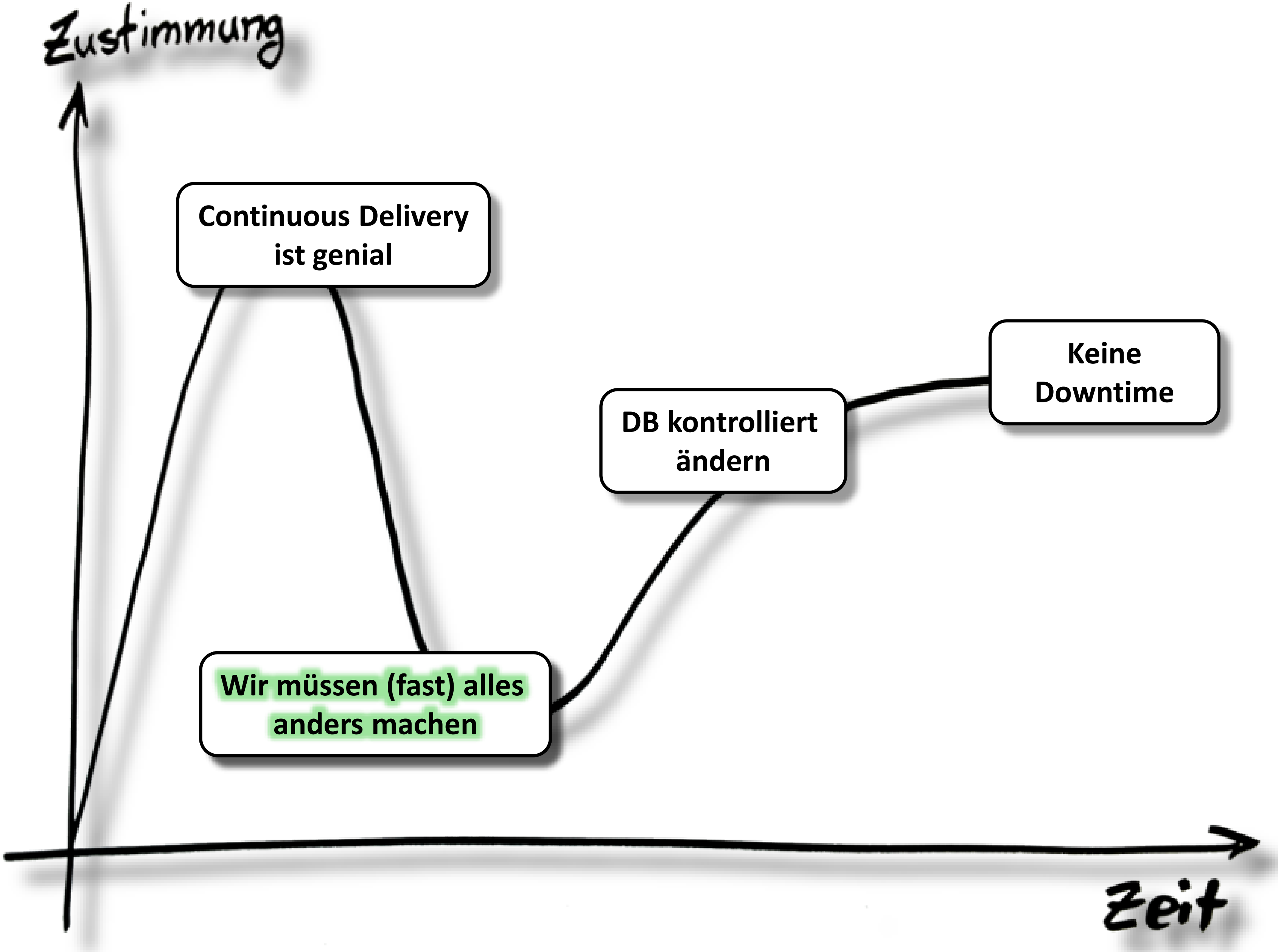
+ Schnellere Reaktionszeiten

+ Weniger Risiko

Microservices


A monolithic application puts all its functionality into a single process...
A microservices architecture puts each element of functionality into a separate service...
... and scales by replicating the monolith on multiple servers
... and scales by distributing these services across servers, replicating as needed.

http://martinfowler.com/articles/microservices.html



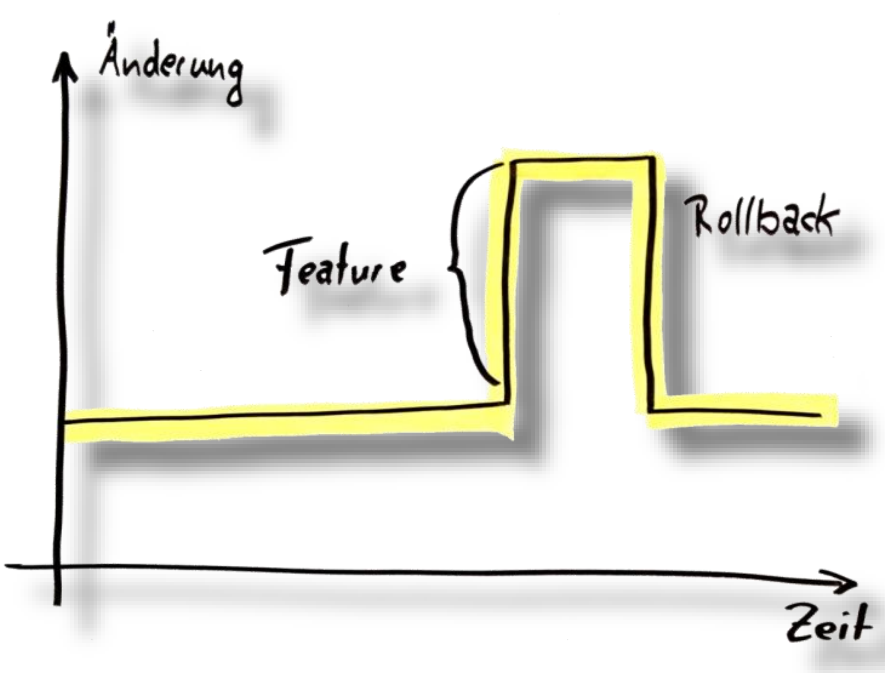
Continuous Delivery ist genial

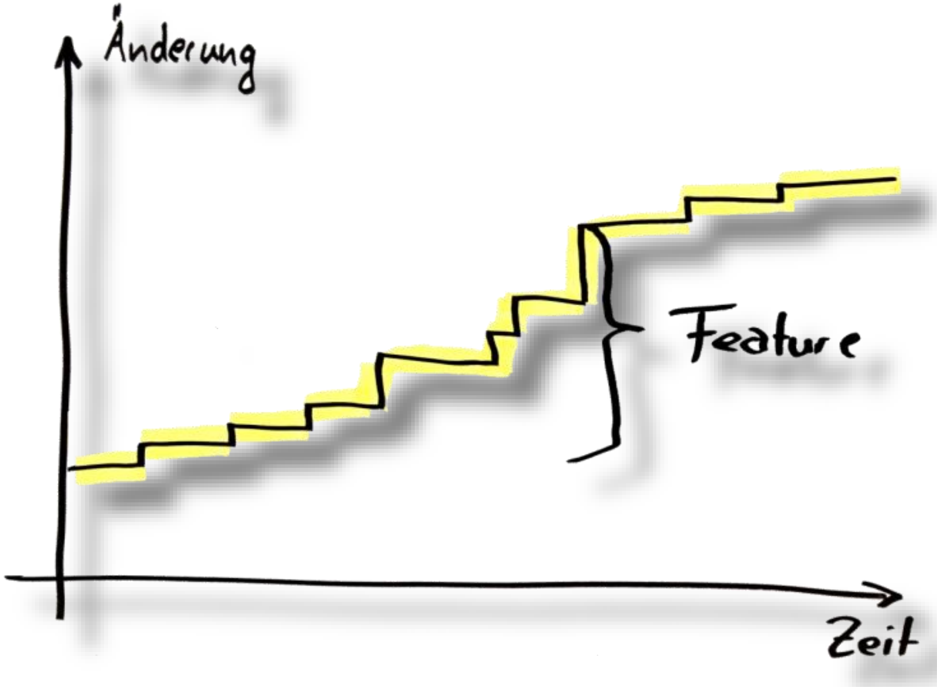
Wir müssen (fast) alles anders machen

DB kontrolliert ändern

Keine Downtime

„Wer schafft mindestens ***1 Produktions-Release pro Tag?*** *“*

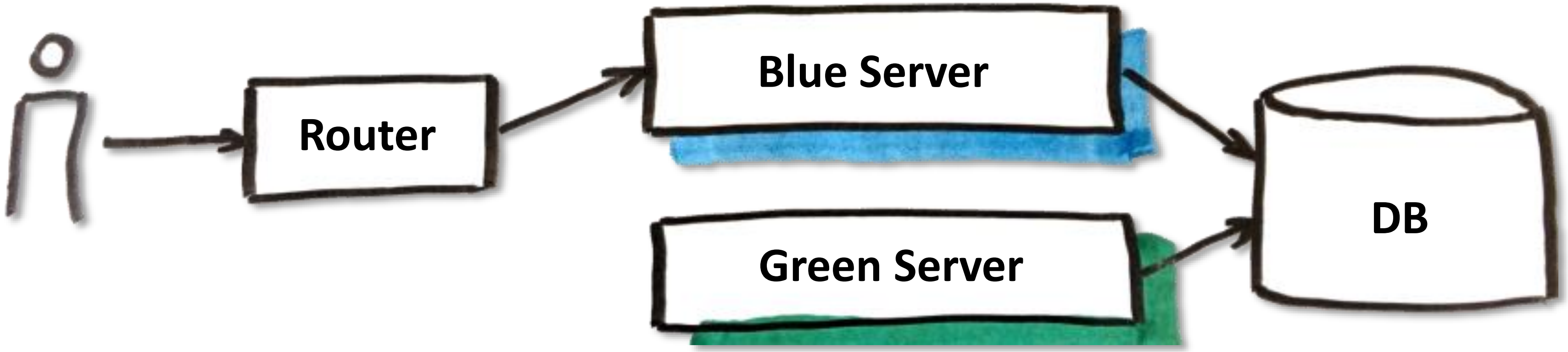




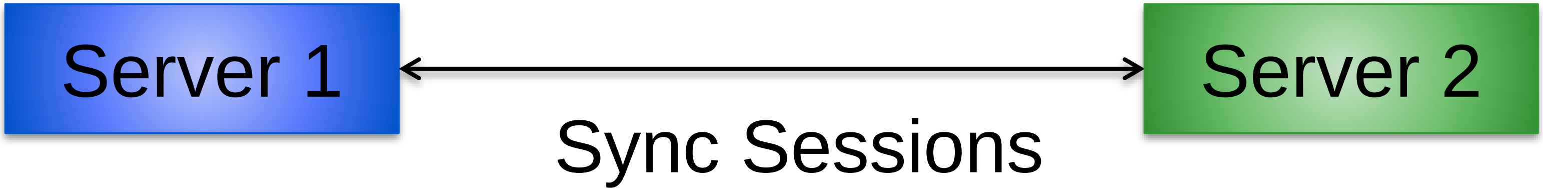
Deploy ≠ Feature Launch

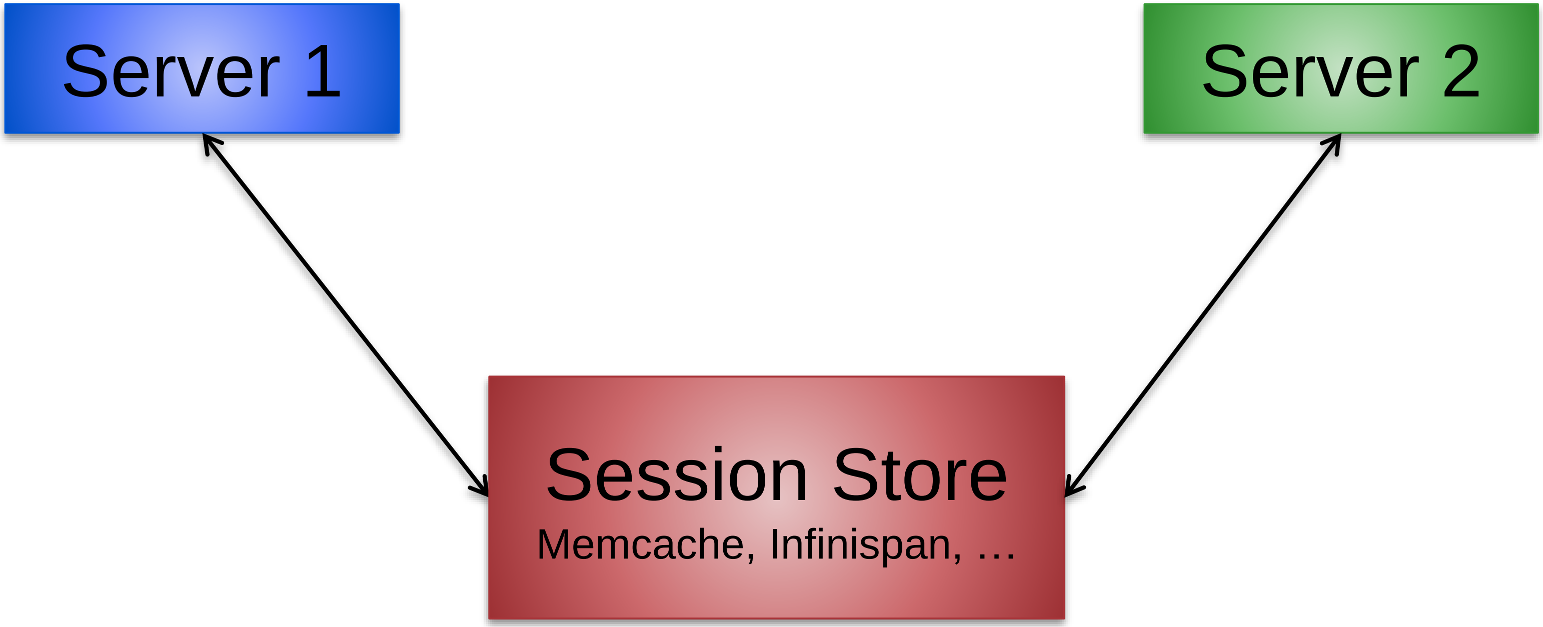
*„Können Sie **ohne Downtime** deployen?“*

Blue-Green-Deployment

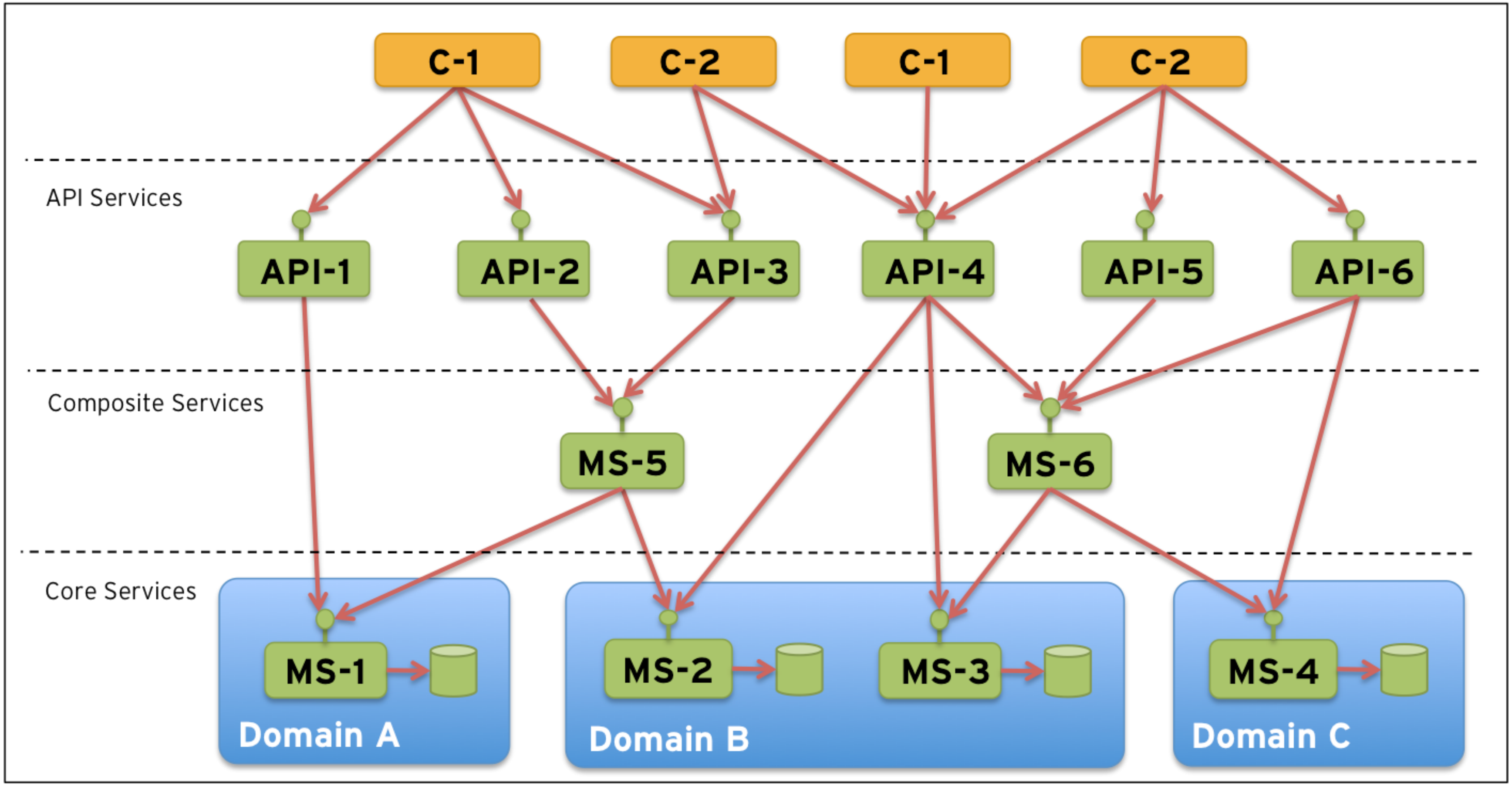


Was passiert mit den ***eingeloggten Usern*** *beim Umschalten zwischen* ***blau*** *und* ***grün****?*


Server 1
Server 2
Sync Sessions


Server 1
Server 2
Session Store
Memcache, Infinispan, …

Nochmal Microservices



http://callistaenterprise.se/blogg/teknik/2015/03/25/an-operations-model-for-microservices/

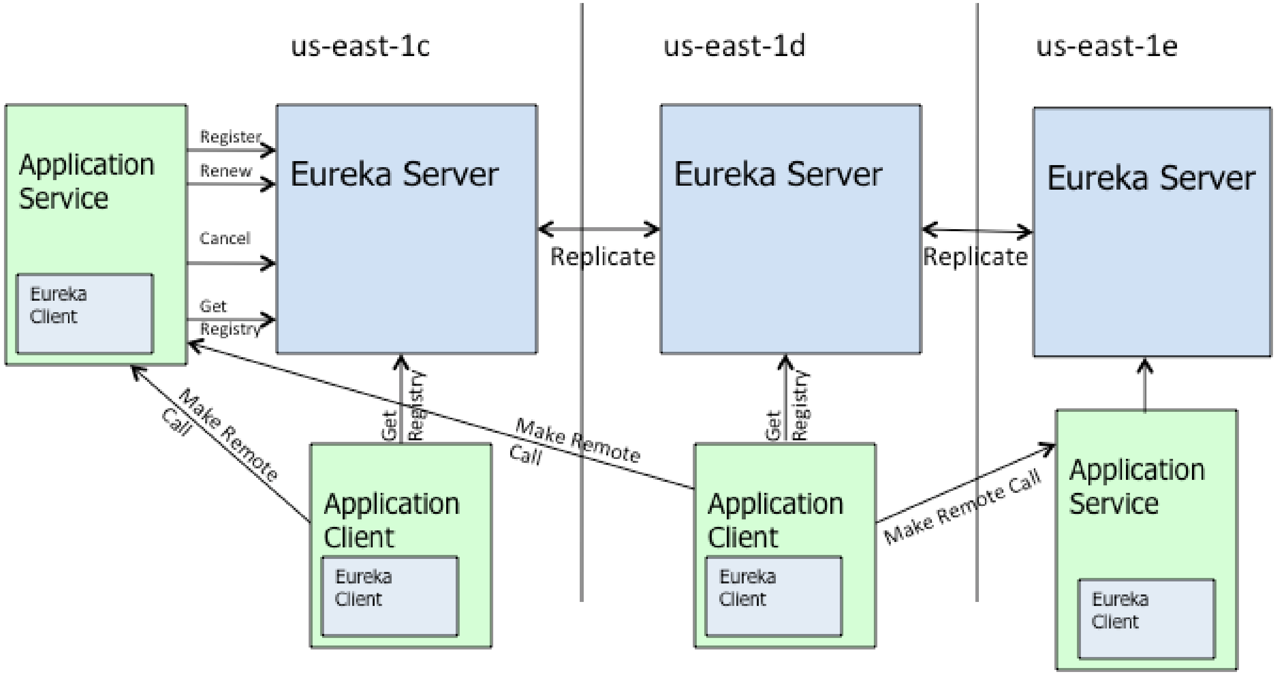
Neue Fragestellungen

- How are all my microservices configured and is it correct?
- What microservices are deployed and where?
- How to keep up with routing information?
- How to prevent chain of failures?
- How to verify that all services are up and running?
- How to track messages that flow between services?
- How to ensure that only the API-services are exposed externally?
- How to secure the API-services?

Neue Dienste

- service discovery
- circuit breaker
- configuration management
- intelligent routing
- micro-proxy
- control bus
- one-time tokens
- global locks
- leadership election
- distributed sessions
- cluster state

Netflix Eureka



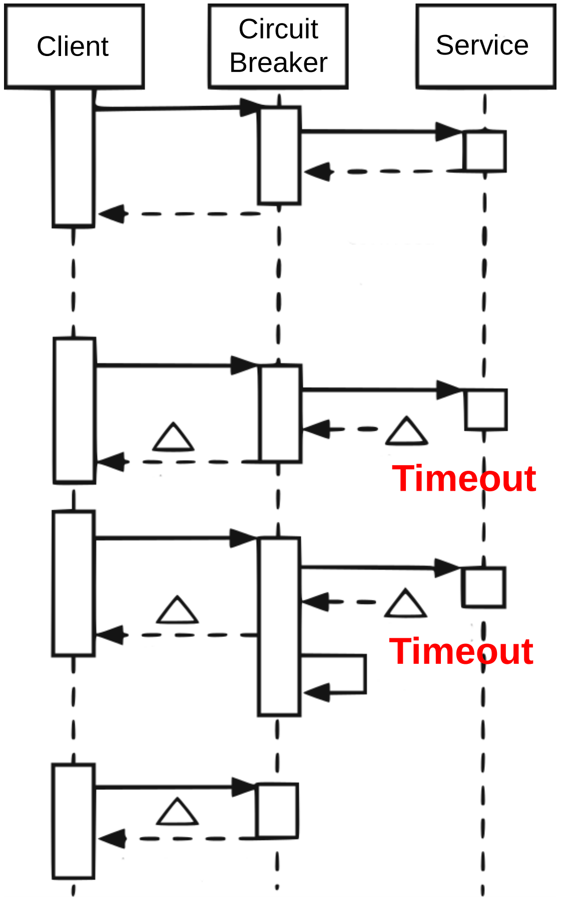
https://github.com/Netflix/eureka/raw/master/images/eureka_architecture.png

Circuit Breaker

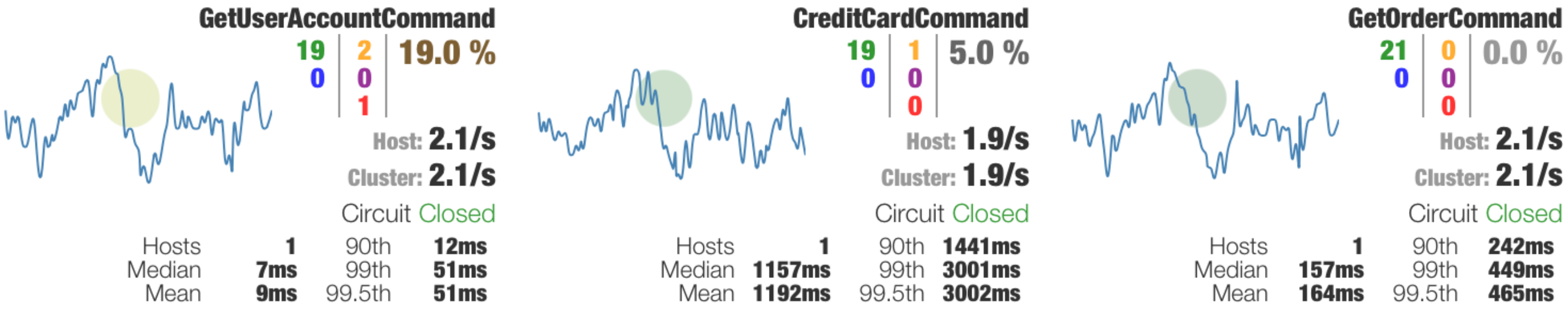
- Der Circuit Breaker (Sicherung) verhindert, dass nicht erreichbare Dienste immer wieder angefragt werden.

- Wird eine maximale Fehleranzahl überschritten, liefert der Circuit Breaker (technisch ein Proxy) sofort einen Fehler und es muss nicht erst ein Timeout abgewartet werden.

- **Hystix** ist eine Circuit Breaker Implementierung von Netflix



Hystrix Dashboard

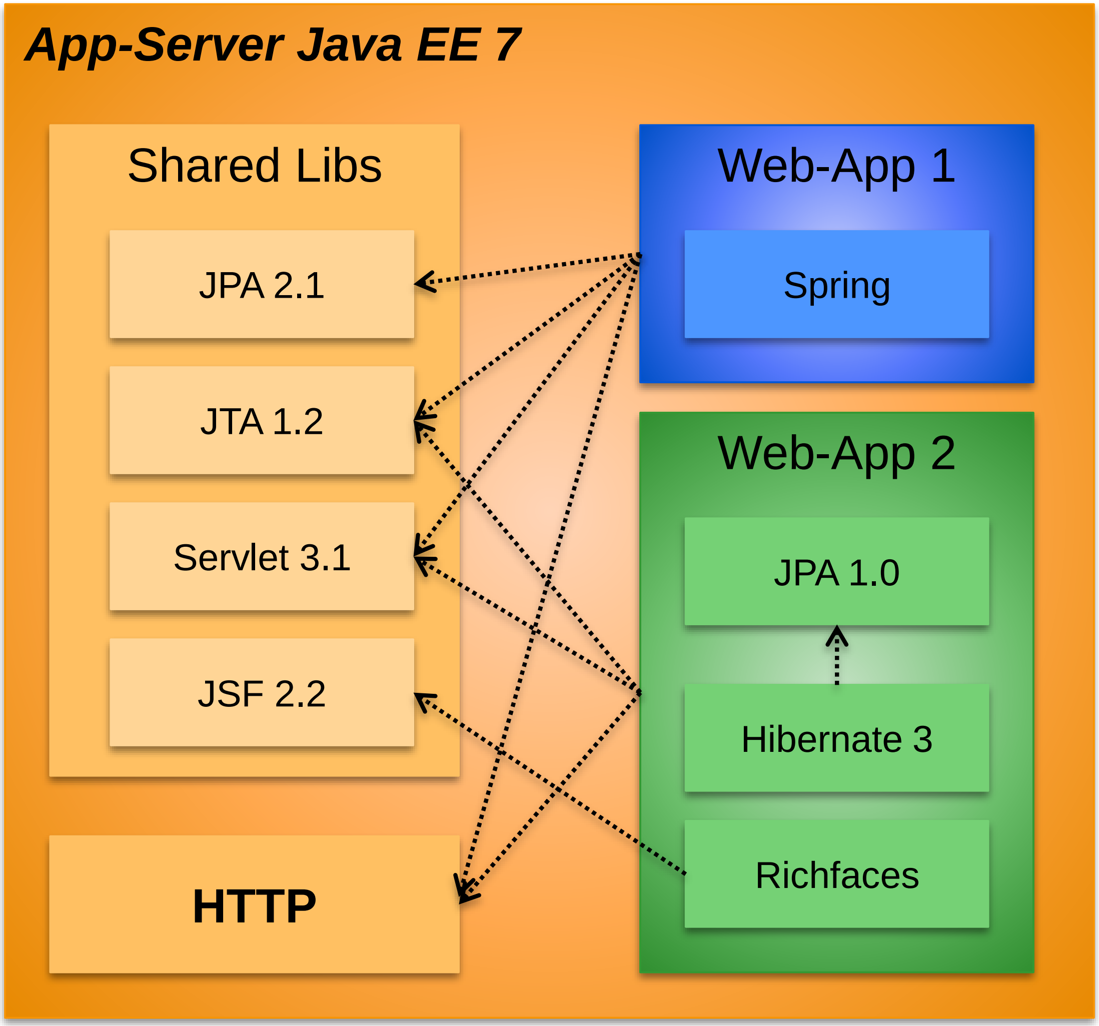


```
GetUserAccountCommand
19 | 2 | 19.0 %
 0 | 0 |
   | 1 |
          Host: 2.1/s
       Cluster: 2.1/s
       Circuit Closed
Hosts      1     90th   12ms
Median   7ms     99th   51ms
Mean     9ms   99.5th   51ms
```

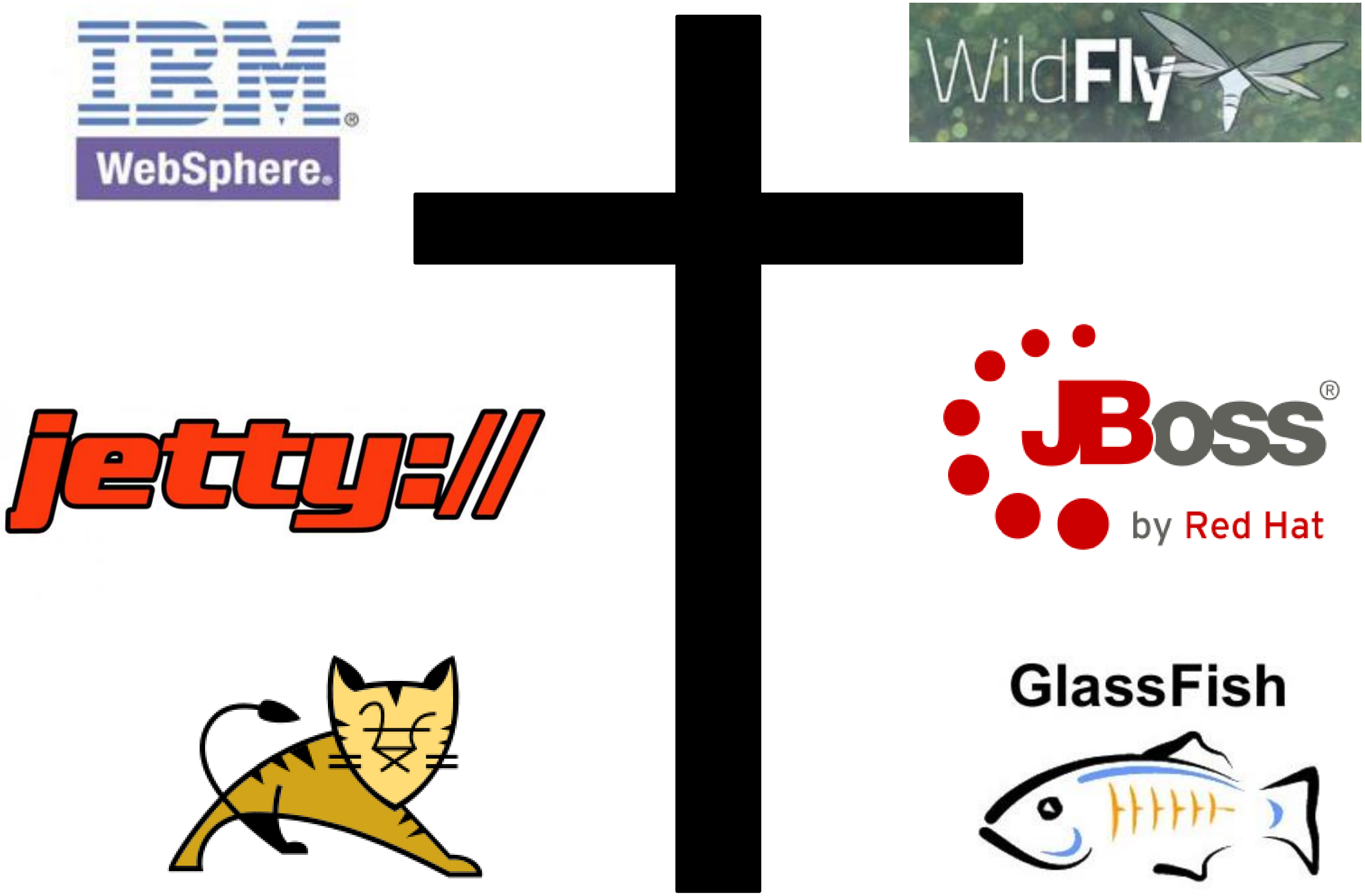
```
CreditCardCommand
19 | 1 | 5.0 %
 0 | 0 |
   | 0 |
          Host: 1.9/s
       Cluster: 1.9/s
       Circuit Closed
Hosts         1     90th   1441ms
Median   1157ms     99th   3001ms
Mean     1192ms   99.5th   3002ms
```

```
GetOrderCommand
21 | 0 | 0.0 %
 0 | 0 |
   | 0 |
          Host: 2.1/s
       Cluster: 2.1/s
       Circuit Closed
Hosts        1     90th   242ms
Median   157ms     99th   449ms
Mean     164ms   99.5th   465ms
```

Abhängigkeiten in App-Servern



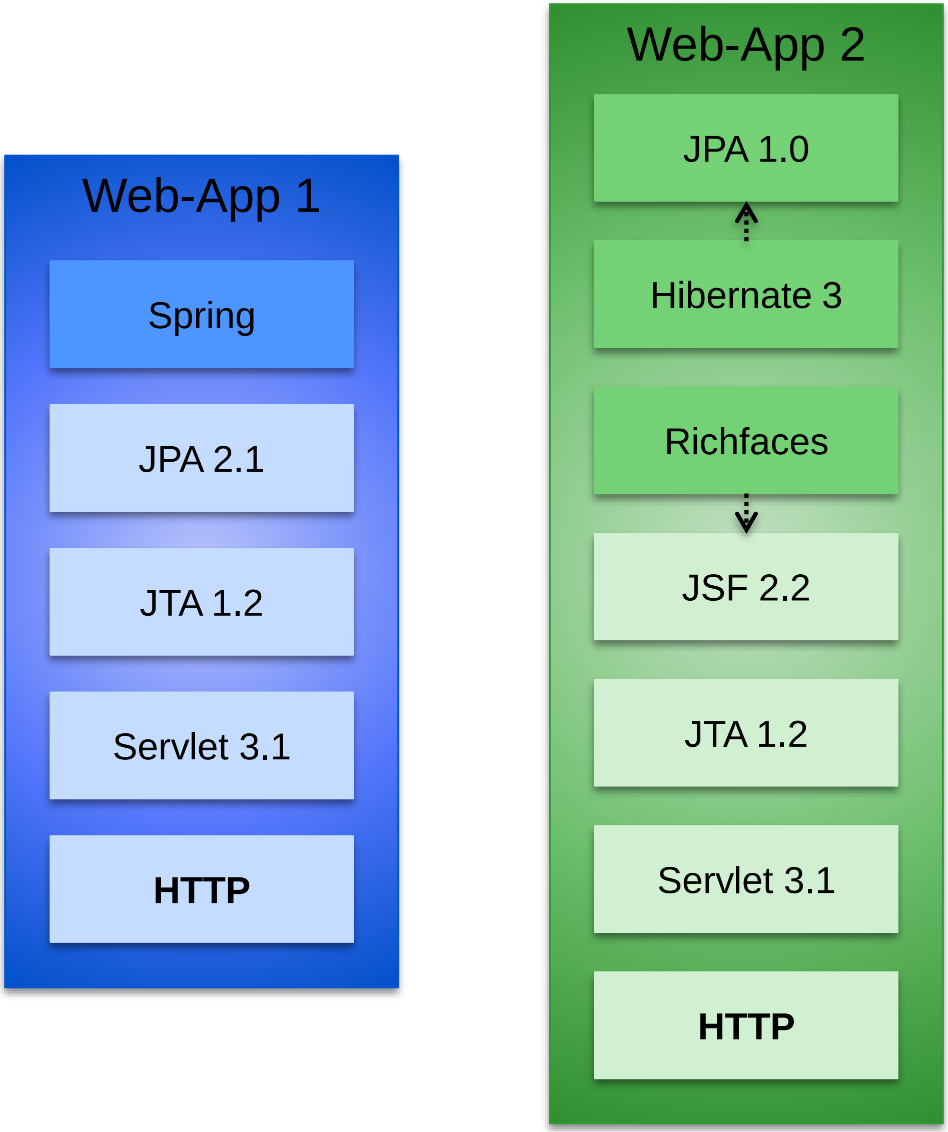
Application-Server sind tot



Was spricht gegen Application-Server?

- Der Betrieb erwartet ein „RPM für Linux“ und kein „WAR / EAR“
  - Spezielles Monitoring für App-Server notwendig
- App-Server für mehrere Apps wird praktisch nicht verwendet
  - eher Clustering von mehreren App-Servern
- App-Server bringt Bibliotheken mit und verhindert somit ein einfaches Update
- App-Server ist ein Problem im Umfeld von Continuous Delivery
  - Deployment-Artefakte können durch die Deployment-Pipe geschoben werden, aber was ist mit der App-Server-Version?

Abhängigkeiten OHNE App-Server



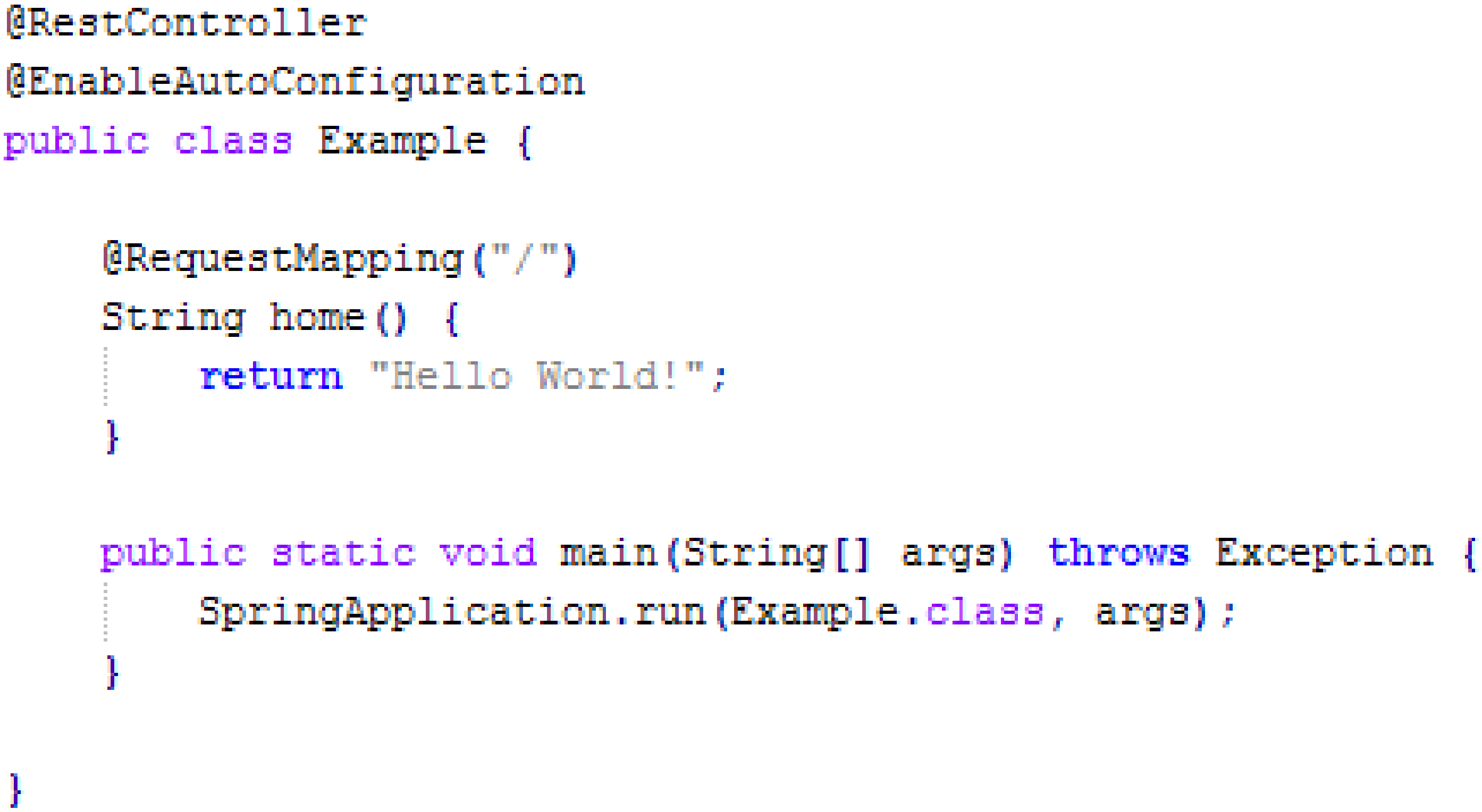
Sind Application-Server tot?

```
com.example:myproject:jar:0.0.1-SNAPSHOT
\- org.springframework.boot:spring-boot-starter-web:jar:1.1.1.BUILD-SNAPSHOT:compile
   +- org.springframework.boot:spring-boot-starter:jar:1.1.1.BUILD-SNAPSHOT:compile
   |  +- org.springframework.boot:spring-boot:jar:1.1.1.BUILD-SNAPSHOT:compile
   |  +- org.springframework.boot:spring-boot-autoconfigure:jar:1.1.1.BUILD-SNAPSHOT:compile
   |  +- org.springframework.boot:spring-boot-starter-logging:jar:1.1.1.BUILD-SNAPSHOT:compile
   |  |  +- org.slf4j:jcl-over-slf4j:jar:1.7.7:compile
   |  |  |  \- org.slf4j:slf4j-api:jar:1.7.7:compile
   |  |  +- org.slf4j:jul-to-slf4j:jar:1.7.7:compile
   |  |  +- org.slf4j:log4j-over-slf4j:jar:1.7.7:compile
   |  |  \- ch.qos.logback:logback-classic:jar:1.1.2:compile
   |  |     \- ch.qos.logback:logback-core:jar:1.1.2:compile
   |  \- org.yaml:snakeyaml:jar:1.13:runtime
   +- org.springframework.boot:spring-boot-starter-tomcat:jar:1.1.1.BUILD-SNAPSHOT:compile
   |  +- org.apache.tomcat.embed:tomcat-embed-core:jar:7.0.54:compile
   |  +- org.apache.tomcat.embed:tomcat-embed-el:jar:7.0.54:compile
   |  \- org.apache.tomcat.embed:tomcat-embed-logging-juli:jar:7.0.54:compile
   +- com.fasterxml.jackson.core:jackson-databind:jar:2.3.3:compile
   |  +- com.fasterxml.jackson.core:jackson-annotations:jar:2.3.3:compile
   |  \- com.fasterxml.jackson.core:jackson-core:jar:2.3.3:compile
   +- org.hibernate:hibernate-validator:jar:5.0.3.Final:compile
   |  +- javax.validation:validation-api:jar:1.1.0.Final:compile
   |  +- org.jboss.logging:jboss-logging:jar:3.1.1.GA:compile
   |  \- com.fasterxml:classmate:jar:1.0.0:compile
   +- org.springframework:spring-core:jar:4.0.5.RELEASE:compile
   +- org.springframework:spring-web:jar:4.0.5.RELEASE:compile
   |  +- org.springframework:spring-aop:jar:4.0.5.RELEASE:compile
   |  |  \- aopalliance:aopalliance:jar:1.0:compile
   |  +- org.springframework:spring-beans:jar:4.0.5.RELEASE:compile
   |  \- org.springframework:spring-context:jar:4.0.5.RELEASE:compile
   \- org.springframework:spring-webmvc:jar:4.0.5.RELEASE:compile
      \- org.springframework:spring-expression:jar:4.0.5.RELEASE:compile
```

Spring Boot – Starter POMs

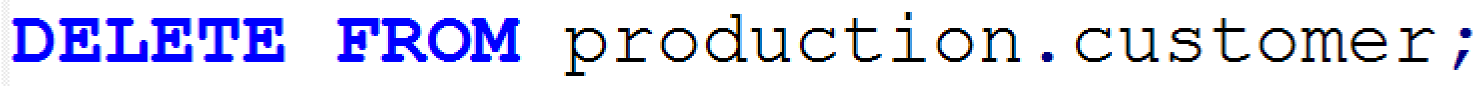
- spring-boot-starter-actuator
- spring-boot-starter-amqp
- spring-boot-starter-aop
- spring-boot-starter-batch
- spring-boot-starter-data-elasticsearch
- spring-boot-starter-data-gemfire
- spring-boot-starter-data-jpa
- spring-boot-starter-data-mongodb
- spring-boot-starter-data-rest
- spring-boot-starter-data-solr
- spring-boot-starter-freemarker
- spring-boot-starter-groovy-templates
- spring-boot-starter-hornetq
- spring-boot-starter-integration
- spring-boot-starter-jdbc
- spring-boot-starter-jetty
- spring-boot-starter-log4j
- spring-boot-starter-logging
- spring-boot-starter-mobile
- spring-boot-starter-parent
- spring-boot-starter-redis
- spring-boot-starter-remote-shell
- spring-boot-starter-security
- spring-boot-starter-social-facebook
- spring-boot-starter-social-linkedin
- spring-boot-starter-social-twitter
- spring-boot-starter-test
- spring-boot-starter-thymeleaf
- spring-boot-starter-tomcat
- spring-boot-starter-velocity
- spring-boot-starter-web
- spring-boot-starter-websocket
- spring-boot-starter-ws

Spring Boot – „Hello World“-Class

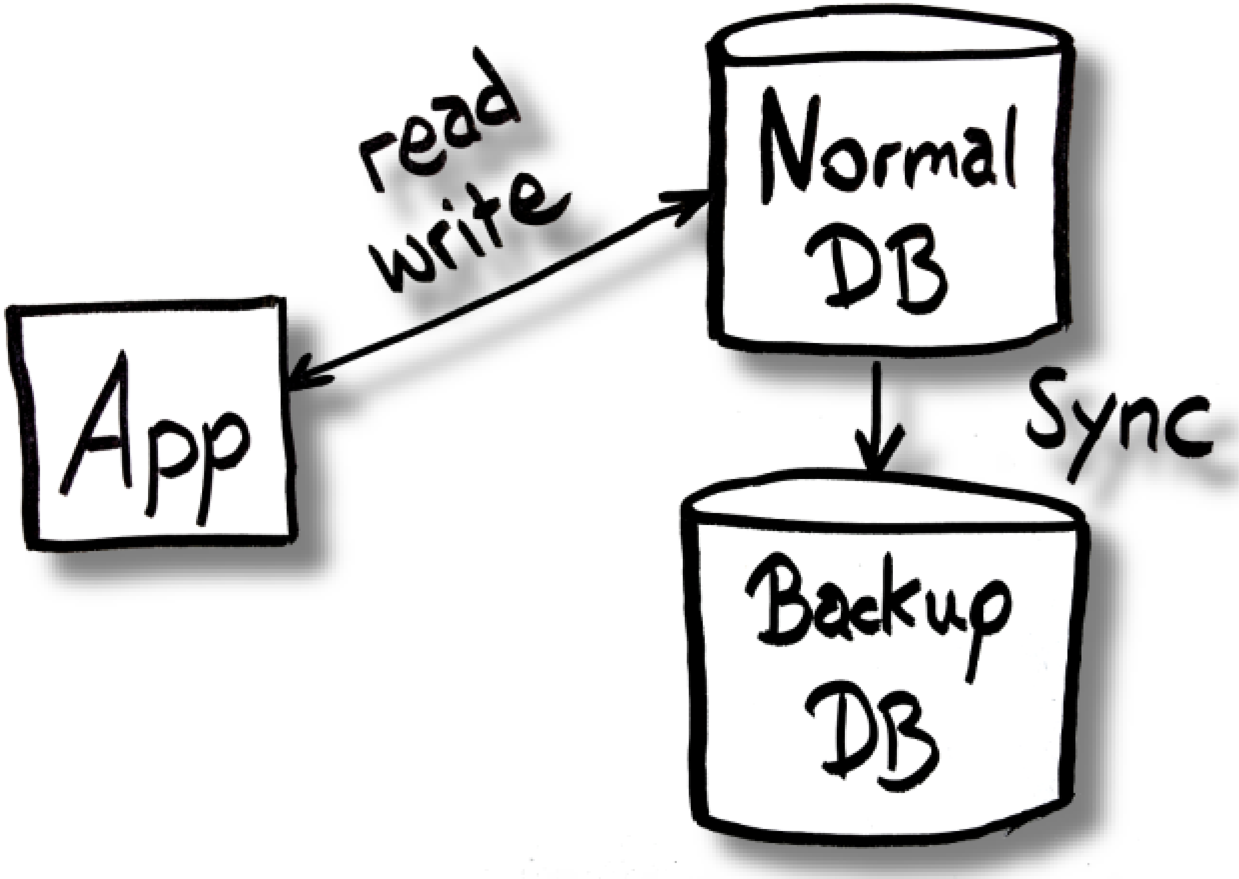


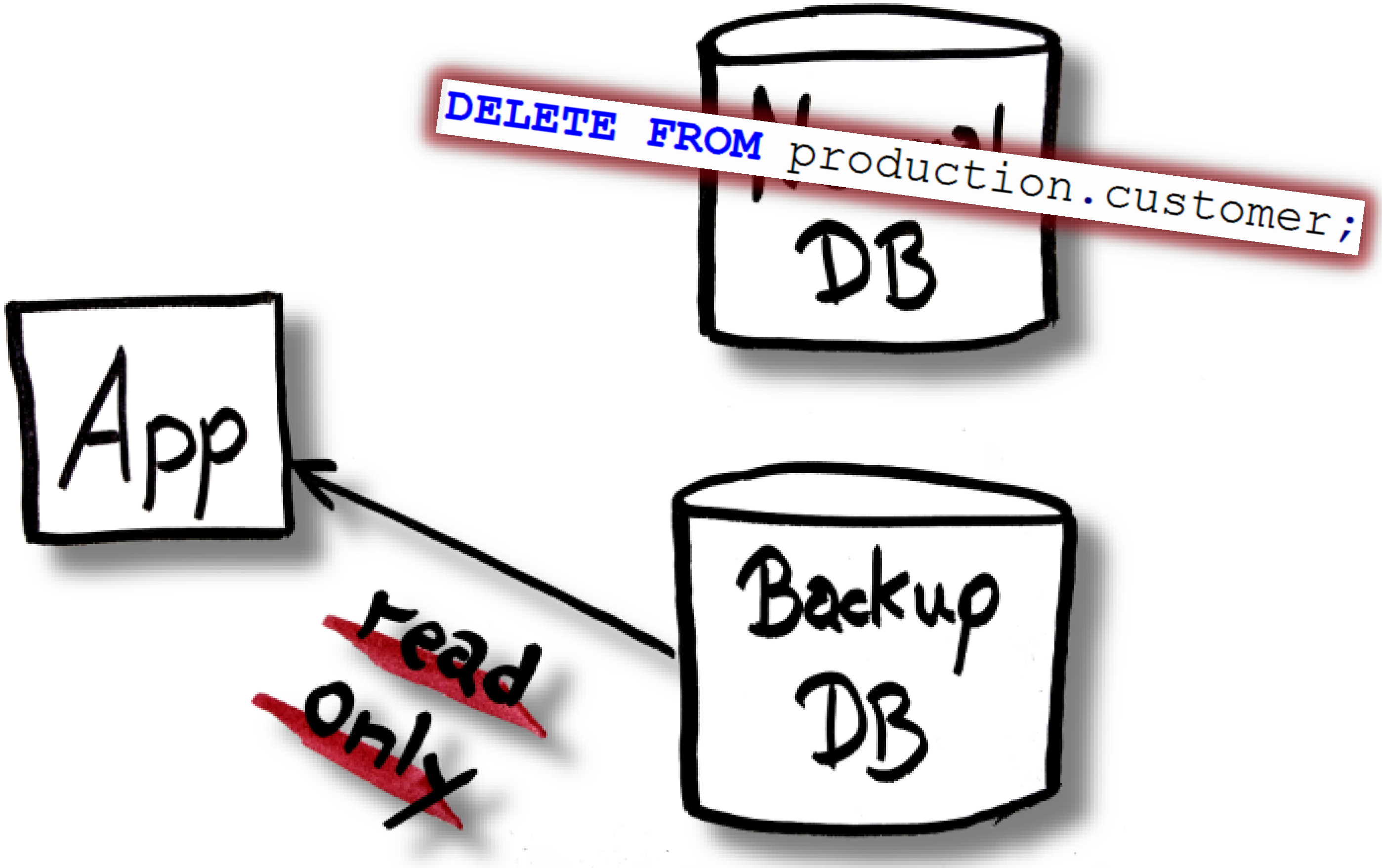
```
@RestController
@EnableAutoConfiguration
public class Example {

    @RequestMapping("/")
    String home() {
        return "Hello World!";
    }

    public static void main(String[] args) throws Exception {
        SpringApplication.run(Example.class, args);
    }

}
```

„Wie sieht bei 30 Deploys pro Tag die Rollback-Strategie aus?



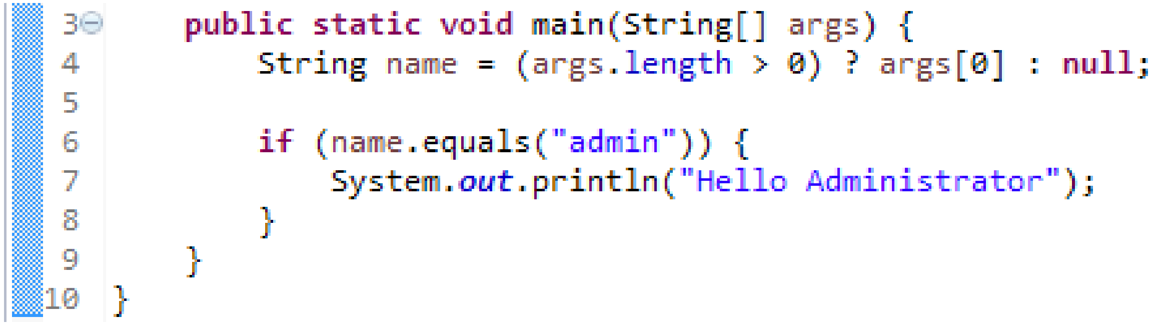
```
DELETE FROM production.customer;
```


read
write
Normal
DB
Sync
App
Backup
DB

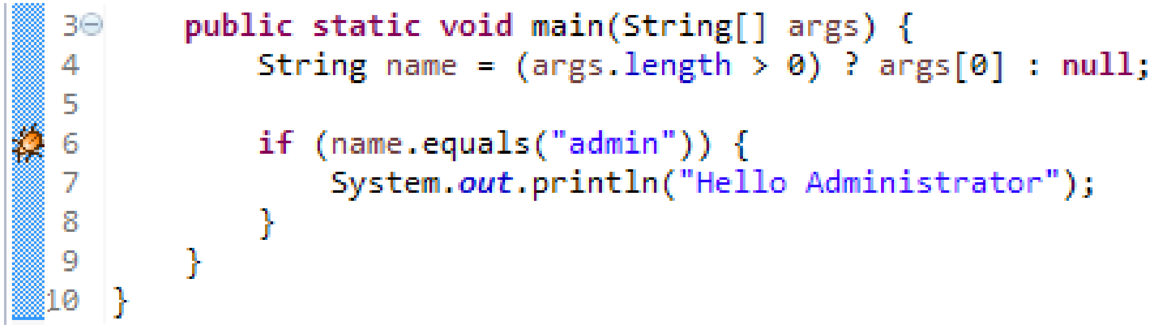


```
DELETE FROM production.customer;
```

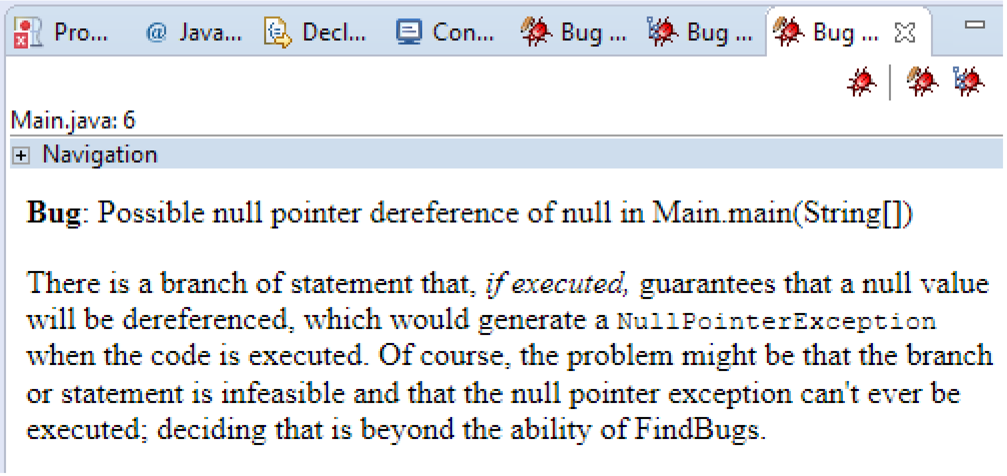
“Chuck Norris does not deploy,
he develops on the production environment.”



```java
    public static void main(String[] args) {
        String name = (args.length > 0) ? args[0] : null;

        if (name.equals("admin")) {
            System.out.println("Hello Administrator");
        }
    }
}
```



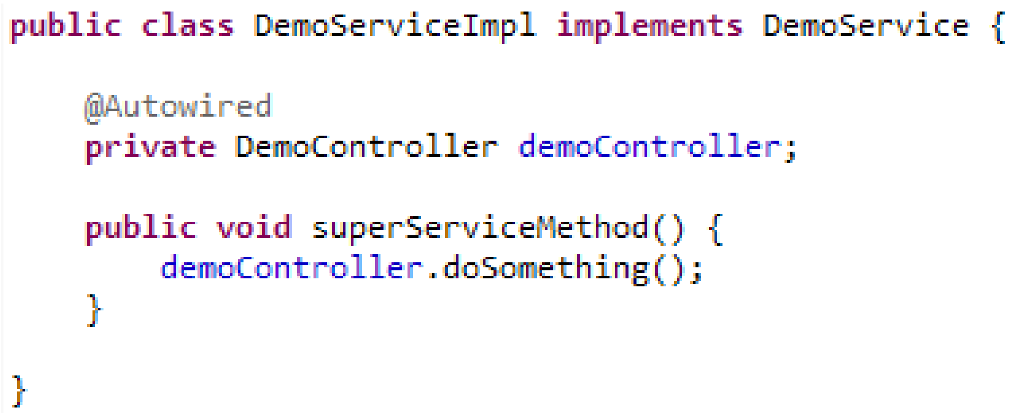
```
 3      public static void main(String[] args) {
 4          String name = (args.length > 0) ? args[0] : null;
 5
 6          if (name.equals("admin")) {
 7              System.out.println("Hello Administrator");
 8          }
 9      }
10  }
```



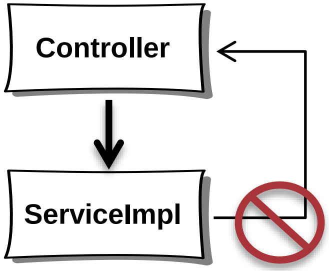
Main.java: 6

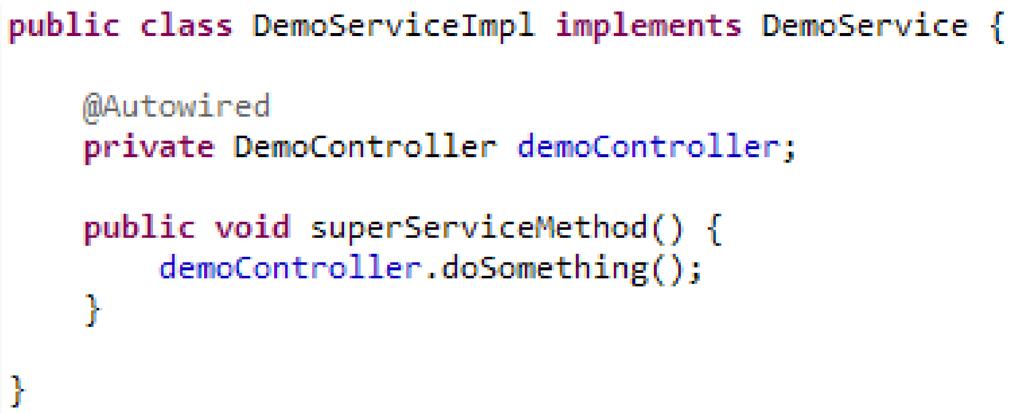
Bug: Possible null pointer dereference of null in Main.main(String[])

There is a branch of statement that, *if executed,* guarantees that a null value will be dereferenced, which would generate a `NullPointerException` when the code is executed. Of course, the problem might be that the branch or statement is infeasible and that the null pointer exception can't ever be executed; deciding that is beyond the ability of FindBugs.

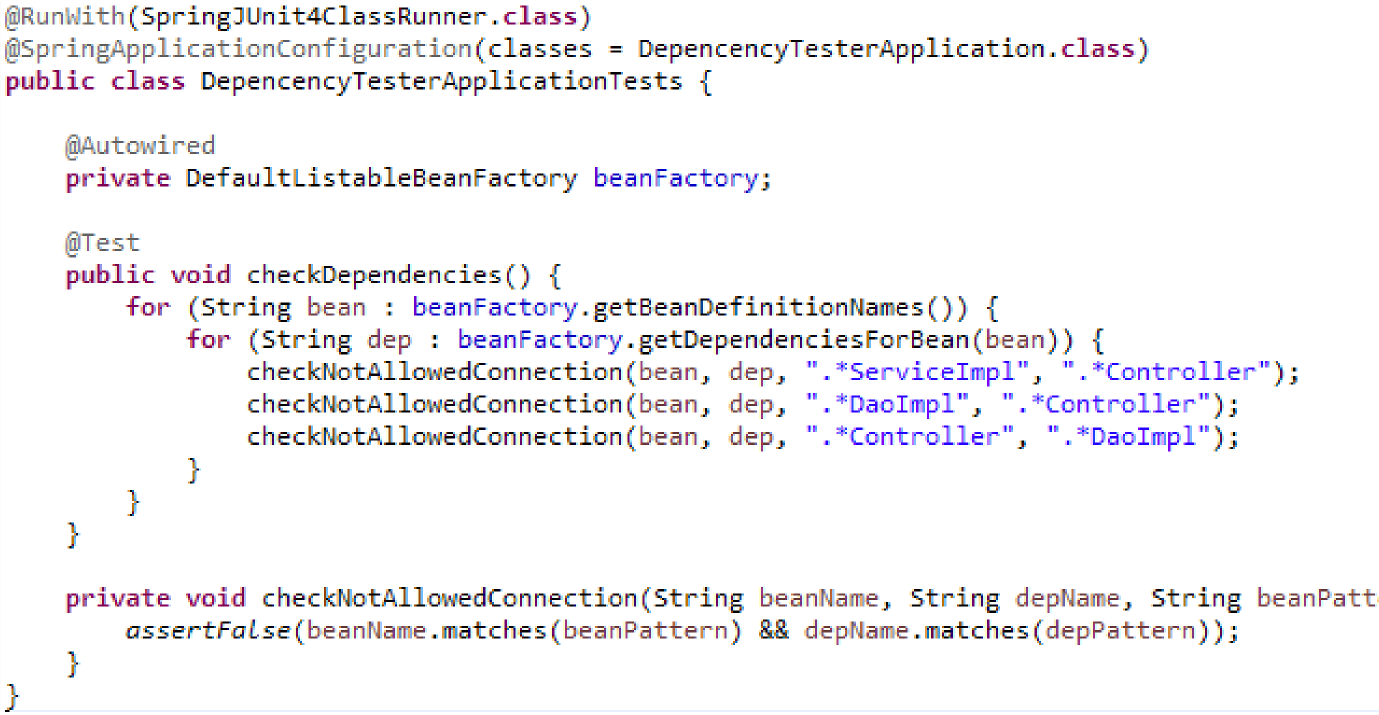


```java
public class DemoServiceImpl implements DemoService {

    @Autowired
    private DemoController demoController;

    public void superServiceMethod() {
        demoController.doSomething();
    }

}
```

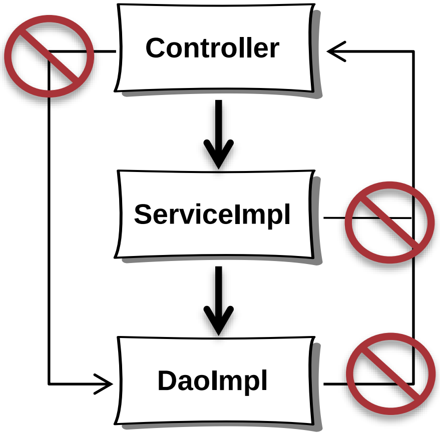




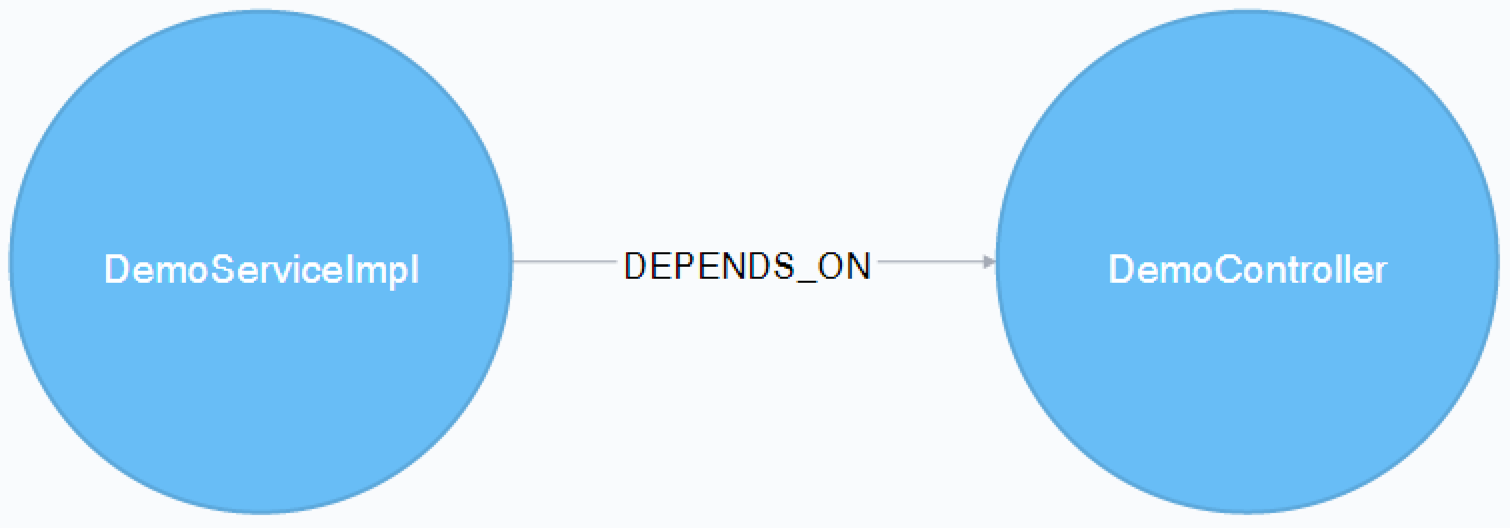
```
public class DemoServiceImpl implements DemoService {

    @Autowired
    private DemoController demoController;

    public void superServiceMethod() {
        demoController.doSomething();
    }

}
```

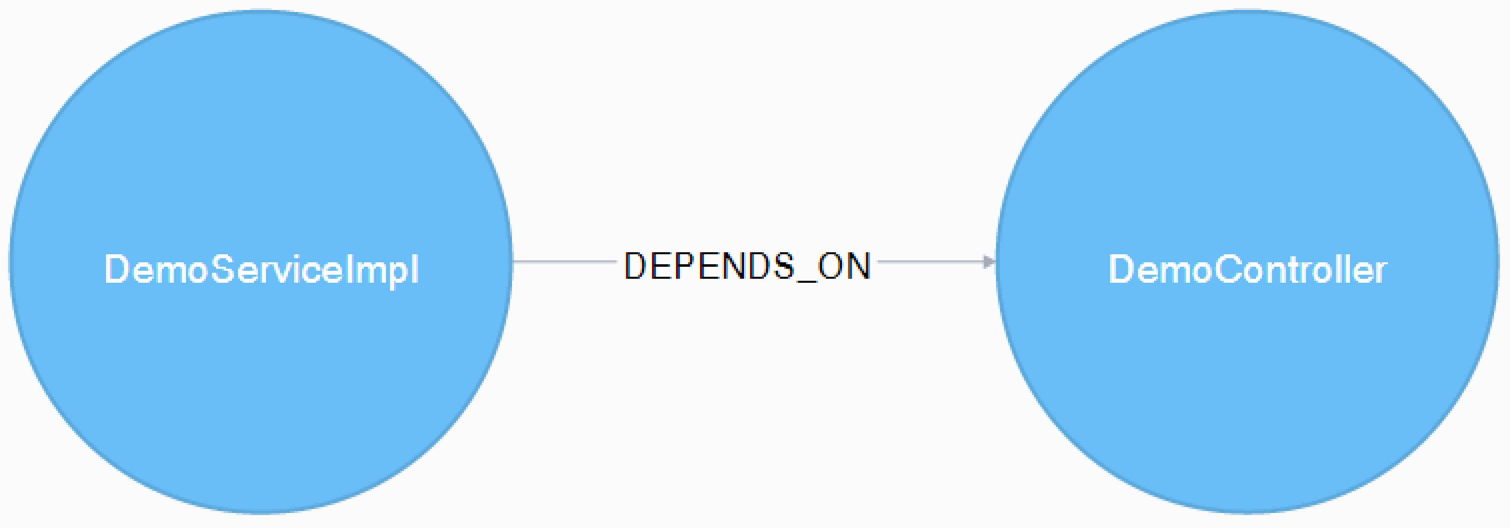


```
@RunWith(SpringJUnit4ClassRunner.class)
@SpringApplicationConfiguration(classes = DepencencyTesterApplication.class)
public class DepencencyTesterApplicationTests {

    @Autowired
    private DefaultListableBeanFactory beanFactory;

    @Test
    public void checkDependencies() {
        for (String bean : beanFactory.getBeanDefinitionNames()) {
            for (String dep : beanFactory.getDependenciesForBean(bean)) {
                checkNotAllowedConnection(bean, dep, ".*ServiceImpl", ".*Controller");
                checkNotAllowedConnection(bean, dep, ".*DaoImpl", ".*Controller");
                checkNotAllowedConnection(bean, dep, ".*Controller", ".*DaoImpl");
            }
        }
    }

    private void checkNotAllowedConnection(String beanName, String depName, String beanPattern, String depPattern) {
        assertFalse(beanName.matches(beanPattern) && depName.matches(depPattern));
    }
}
```

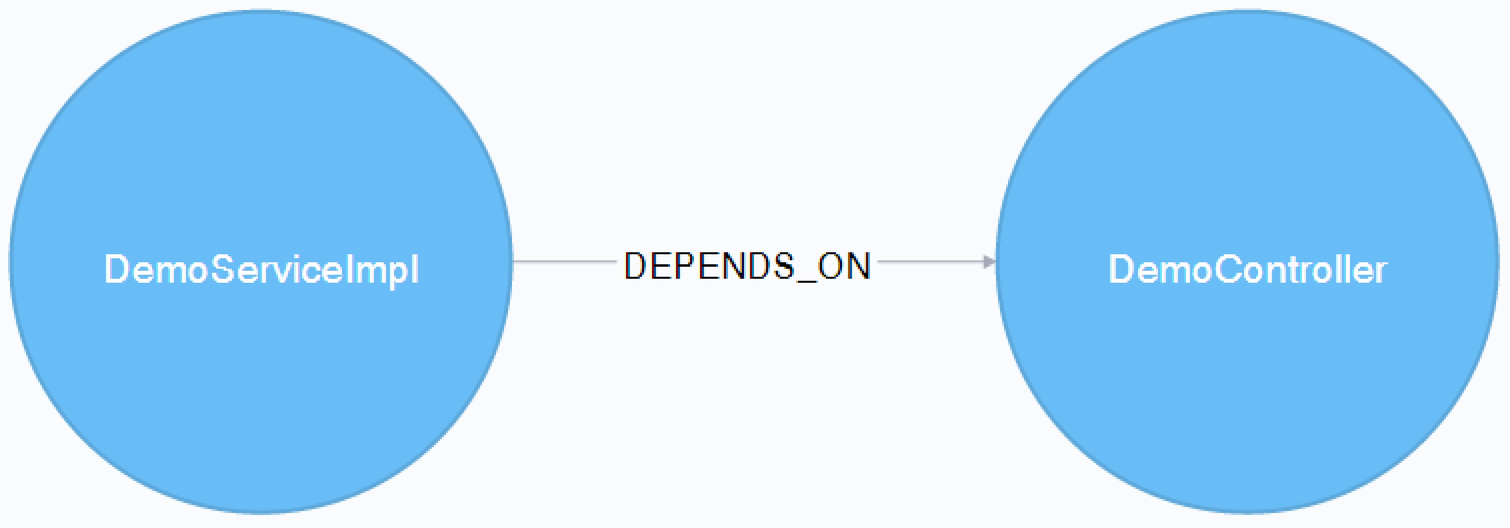


https://github.com/thorstenmaier/architecture-layer-check/

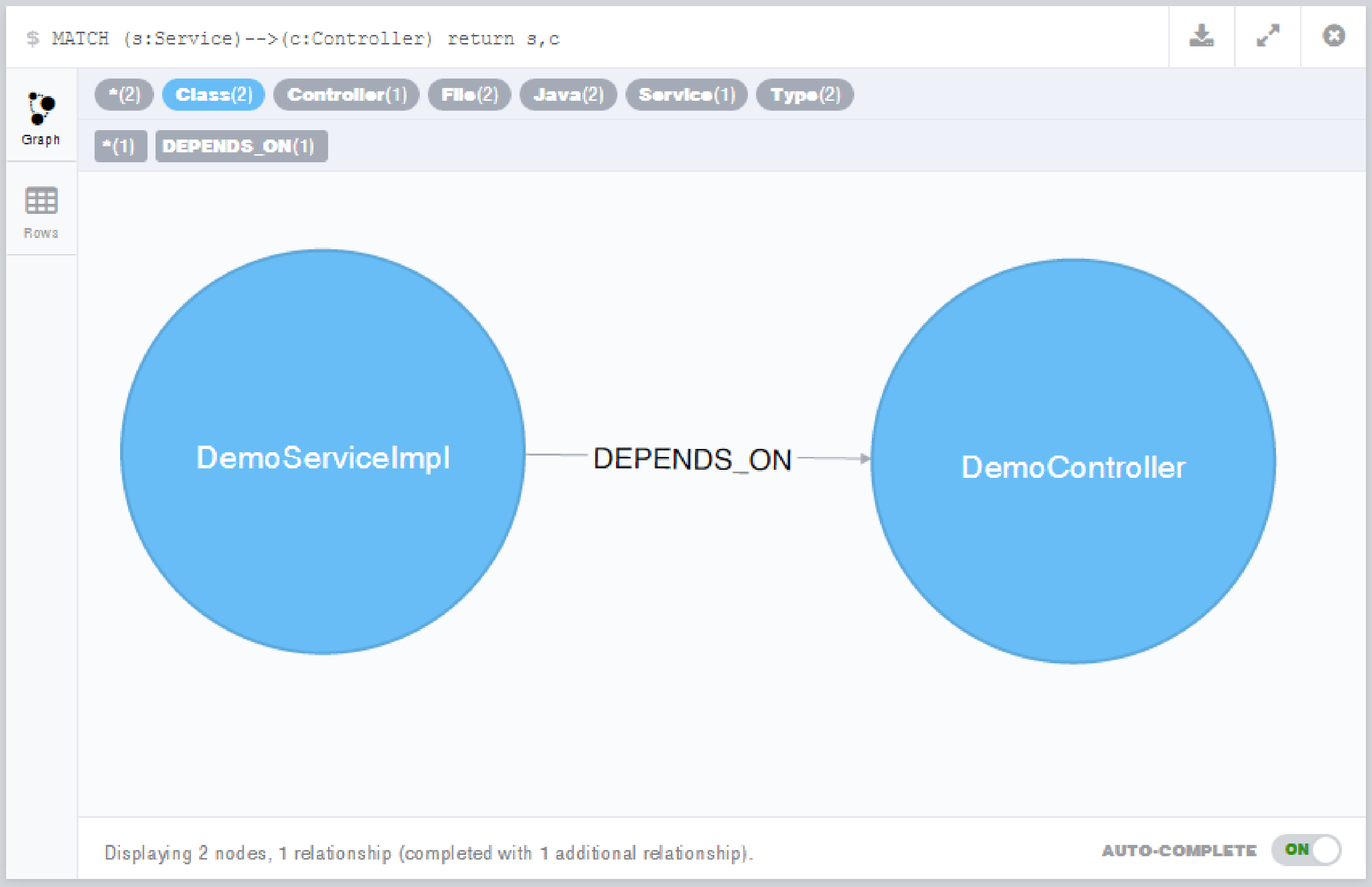

DemoServiceImpl
DEPENDS_ON
DemoController



```
(Service) - [DEPENDS_ON] -> (Controller)
```

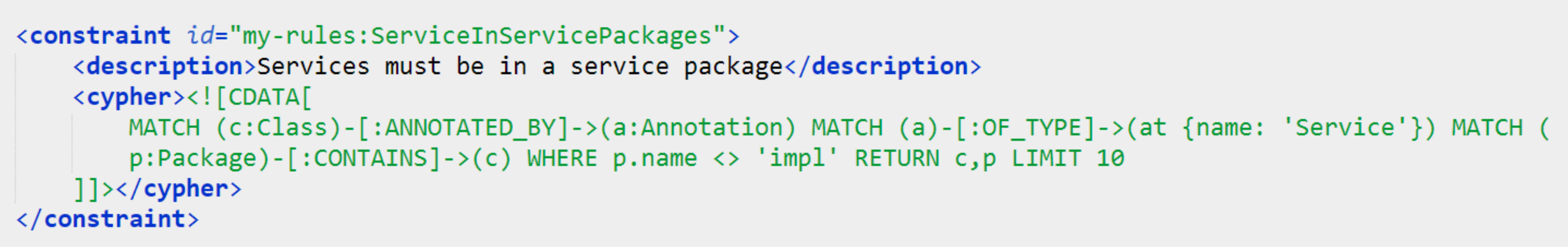


```
MATCH (s:Service)-[:DEPENDS_ON]->(c:Controller) return s,c
```

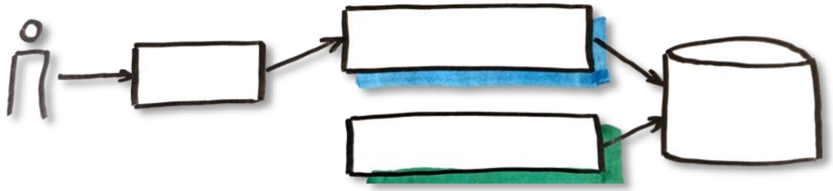


```
$ MATCH (s:Service)-->(c:Controller) return s,c
```

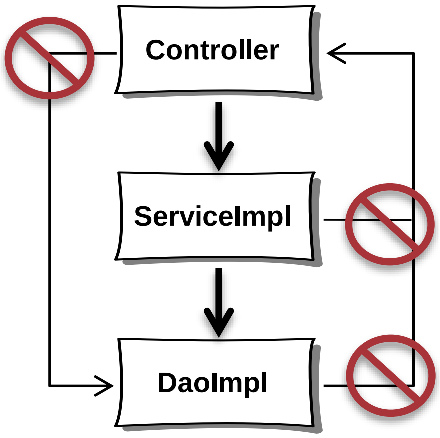
Displaying 2 nodes, 1 relationship (completed with 1 additional relationship).



```
<constraint id="my-rules:ServiceInServicePackages">
    <description>Services must be in a service package</description>
    <cypher><![CDATA[
        MATCH (c:Class)-[:ANNOTATED_BY]->(a:Annotation) MATCH (a)-[:OF_TYPE]->(at {name: 'Service'}) MATCH (
        p:Package)-[:CONTAINS]->(c) WHERE p.name <> 'impl' RETURN c,p LIMIT 10
    ]]></cypher>
</constraint>
```



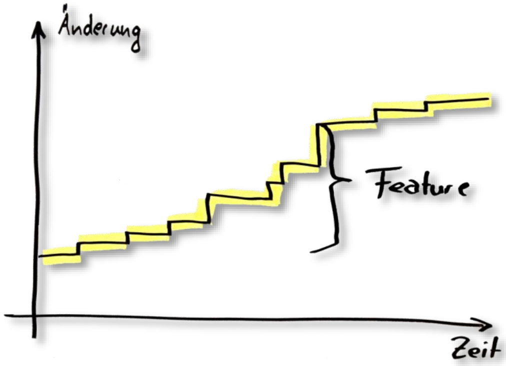
Wollen Sie das alles?

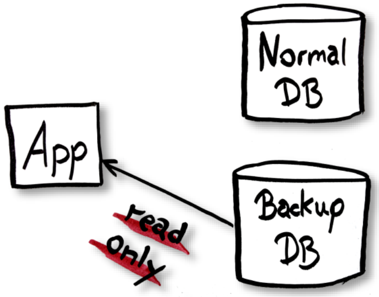


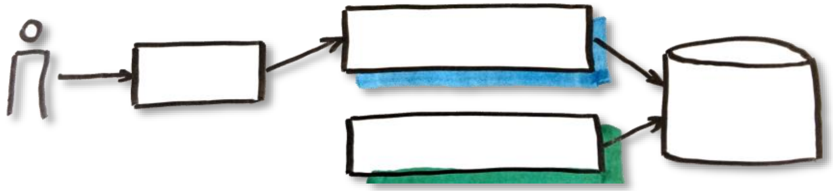
Controller

ServiceImpl

DaoImpl

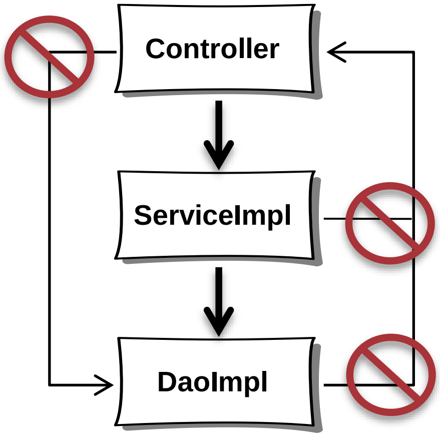


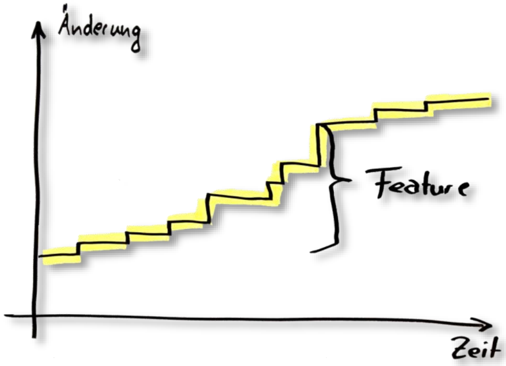


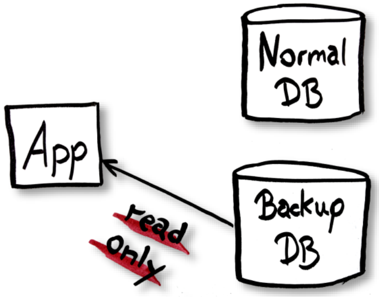


Wollen Sie das alles?

Will Ihr Kunde das???





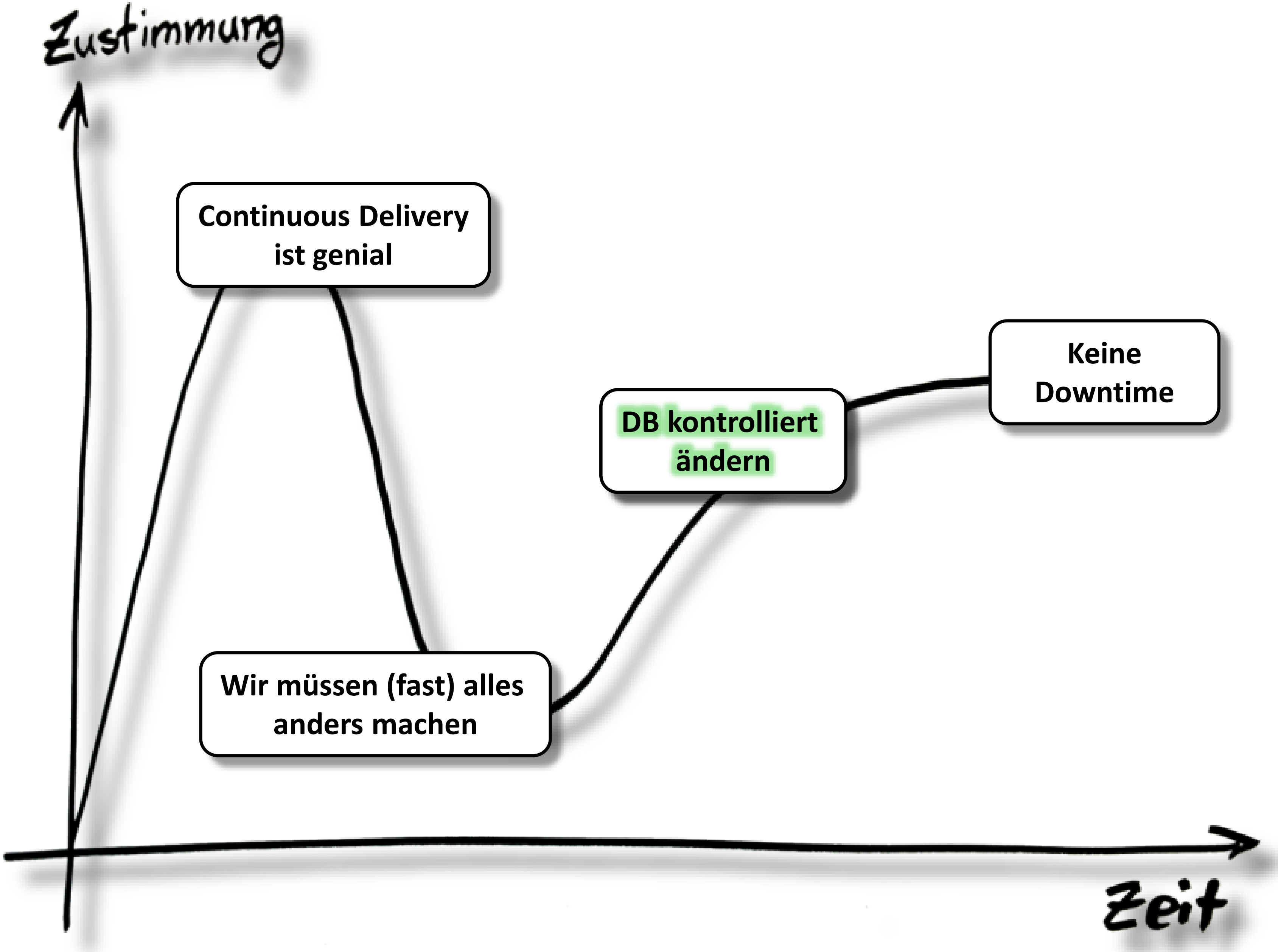


Dann benötigen wir

kontrollierte Änderungen

der Datenbank

ohne Downtime



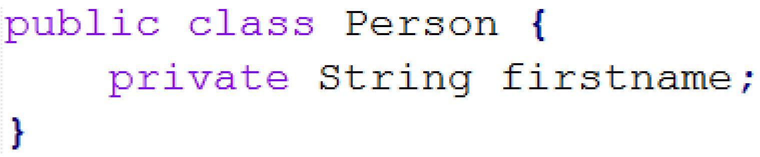
Continuous Delivery ist genial

Wir müssen (fast) alles anders machen

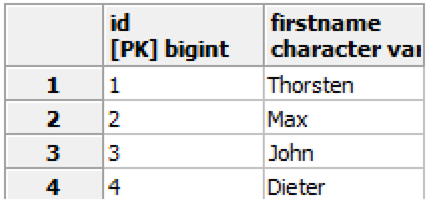
DB kontrolliert ändern

Keine Downtime

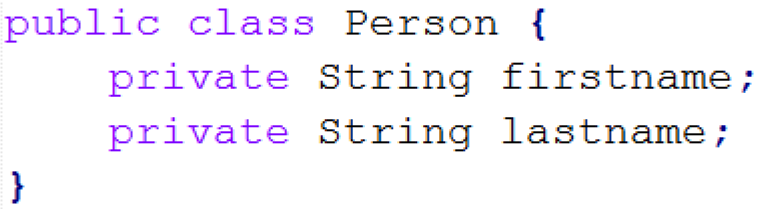
Code und Datenbank müssen synchron bleiben



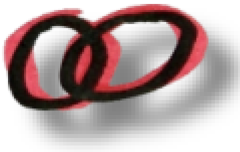
```
public class Person {
    private String firstname;
}
```

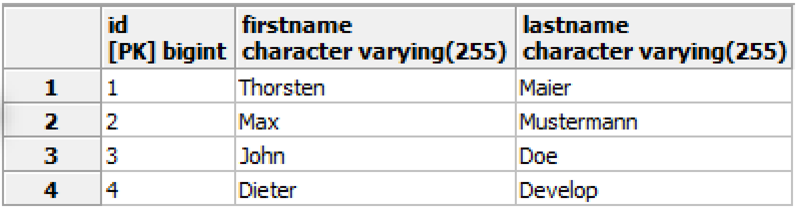


	id [PK] bigint	firstname character var
1	1	Thorsten
2	2	Max
3	3	John
4	4	Dieter



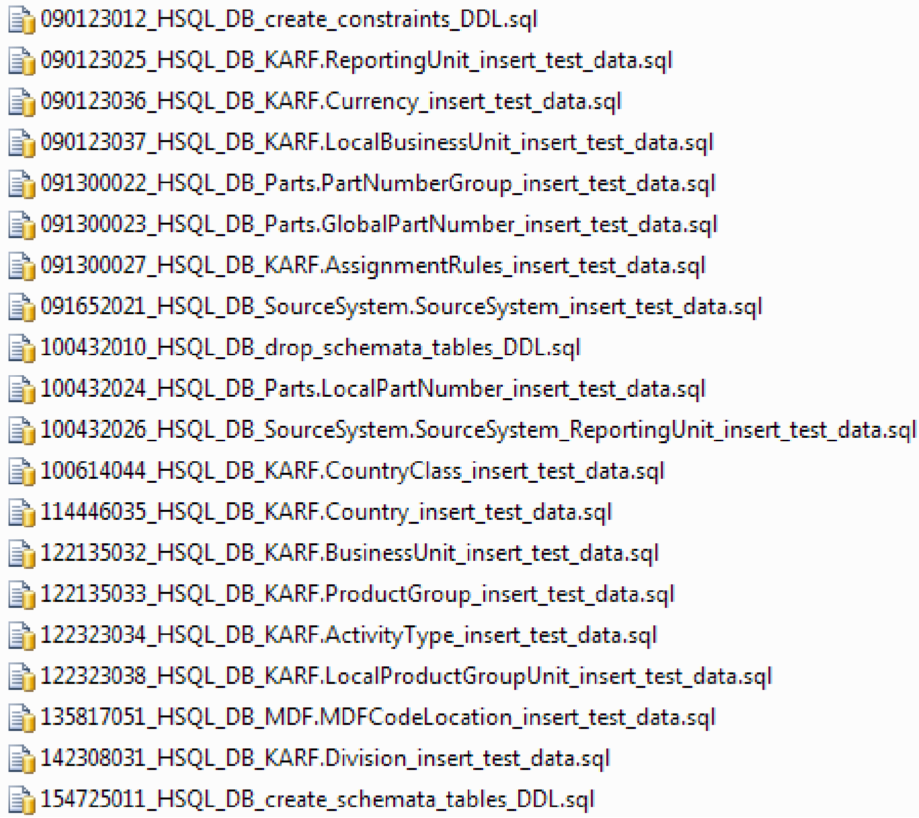
```
public class Person {
    private String firstname;
    private String lastname;
}
```





	id [PK] bigint	firstname character varying(255)	lastname character varying(255)
1	1	Thorsten	Maier
2	2	Max	Mustermann
3	3	John	Doe
4	4	Dieter	Develop

Kennen Sie das?



```
090123012_HSQL_DB_create_constraints_DDL.sql
090123025_HSQL_DB_KARF.ReportingUnit_insert_test_data.sql
090123036_HSQL_DB_KARF.Currency_insert_test_data.sql
090123037_HSQL_DB_KARF.LocalBusinessUnit_insert_test_data.sql
091300022_HSQL_DB_Parts.PartNumberGroup_insert_test_data.sql
091300023_HSQL_DB_Parts.GlobalPartNumber_insert_test_data.sql
091300027_HSQL_DB_KARF.AssignmentRules_insert_test_data.sql
091652021_HSQL_DB_SourceSystem.SourceSystem_insert_test_data.sql
100432010_HSQL_DB_drop_schemata_tables_DDL.sql
100432024_HSQL_DB_Parts.LocalPartNumber_insert_test_data.sql
100432026_HSQL_DB_SourceSystem.SourceSystem_ReportingUnit_insert_test_data.sql
100614044_HSQL_DB_KARF.CountryClass_insert_test_data.sql
114446035_HSQL_DB_KARF.Country_insert_test_data.sql
122135032_HSQL_DB_KARF.BusinessUnit_insert_test_data.sql
122135033_HSQL_DB_KARF.ProductGroup_insert_test_data.sql
122323034_HSQL_DB_KARF.ActivityType_insert_test_data.sql
122323038_HSQL_DB_KARF.LocalProductGroupUnit_insert_test_data.sql
135817051_HSQL_DB_MDF.MDFCodeLocation_insert_test_data.sql
142308031_HSQL_DB_KARF.Division_insert_test_data.sql
154725011_HSQL_DB_create_schemata_tables_DDL.sql
```

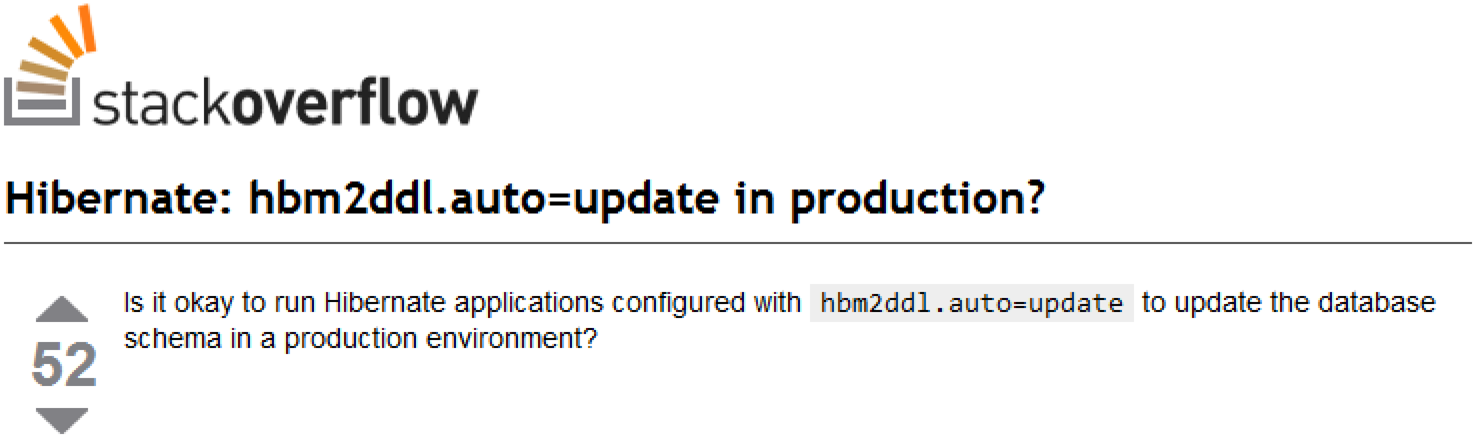
„Vor dem Update von **Version 1.0.2** auf **1.1**
musst du erst noch **die neuen SQL-Skripte** einspielen."

2 Lösungsmöglichkeiten

1. hbm2ddl.auto=update

2. Manuelle SQL-Skripte

1. hbm2ddl.auto=update?



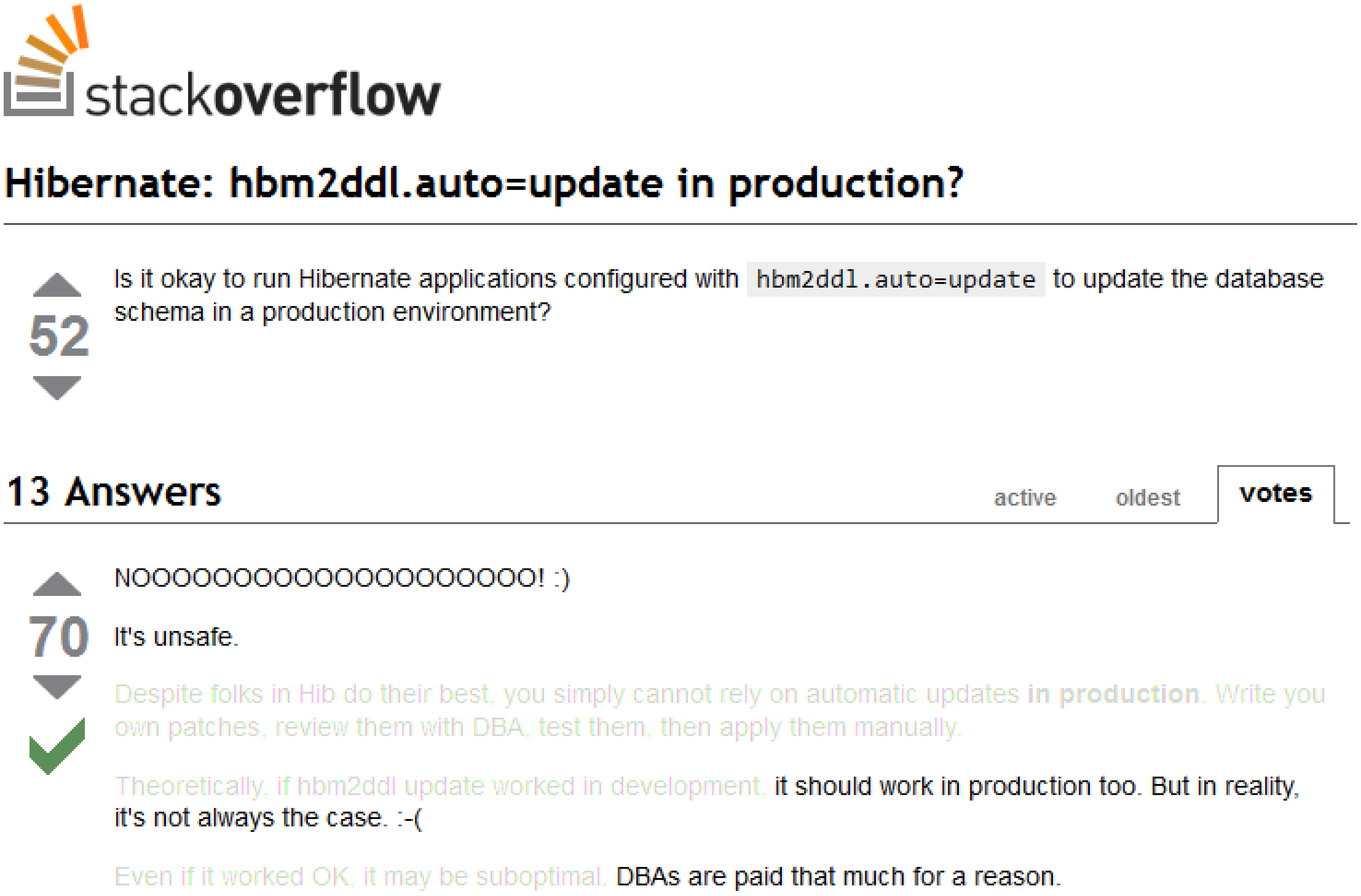
stackoverflow

Hibernate: hbm2ddl.auto=update in production?

52

Is it okay to run Hibernate applications configured with `hbm2ddl.auto=update` to update the database schema in a production environment?

1. hbm2ddl.auto=update?



stackoverflow

Hibernate: hbm2ddl.auto=update in production?

52

Is it okay to run Hibernate applications configured with `hbm2ddl.auto=update` to update the database schema in a production environment?

13 Answers

active oldest **votes**

70 ✓

NOOOOOOOOOOOOOOOOOOOOOO! :)

It's unsafe.

Despite folks in Hib do their best, you simply cannot rely on automatic updates **in production**. Write you own patches, review them with DBA, test them, then apply them manually.

Theoretically, if hbm2ddl update worked in development, it should work in production too. But in reality, it's not always the case. :-(

Even if it worked OK, it may be suboptimal. DBAs are paid that much for a reason.

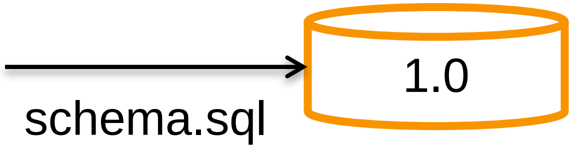
http://stackoverflow.com/questions/221379/hibernate-hbm2ddl-auto-update-in-production

2 Lösungsmöglichkeiten

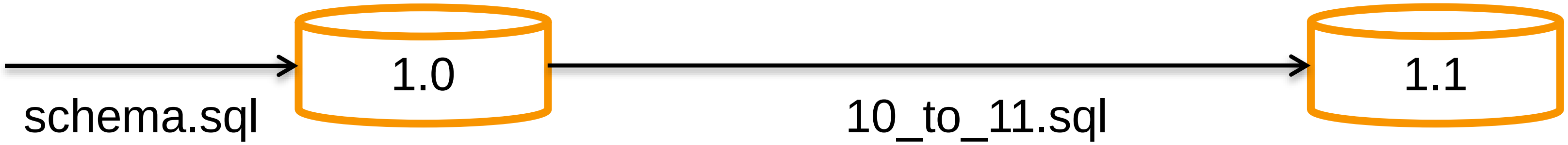
1. ~~hbm2ddl.auto=update~~

2. Manuelle SQL-Skripte

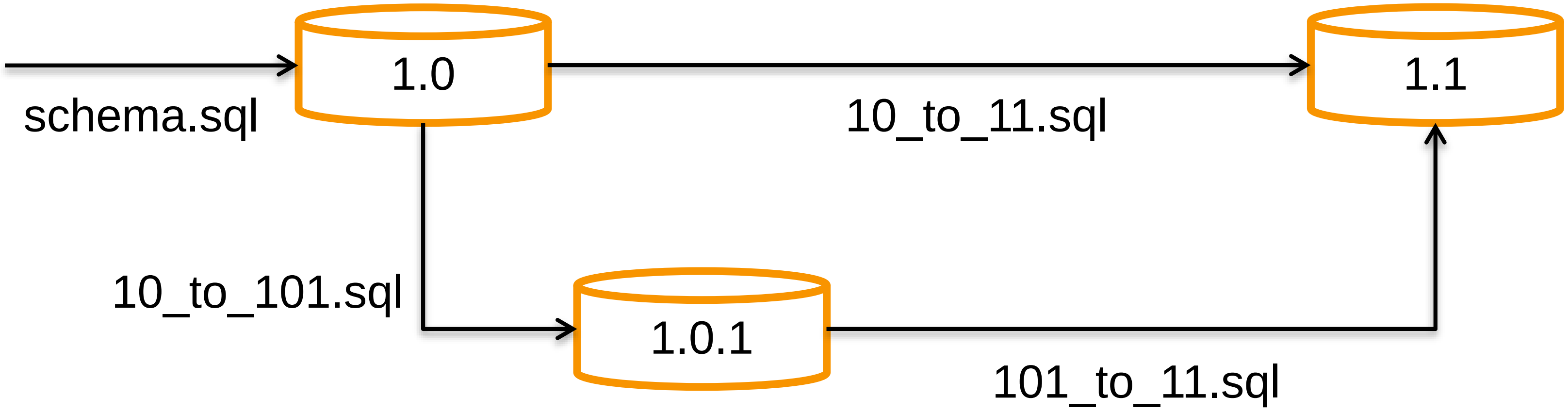
2. Manuelle SQL-Skripte



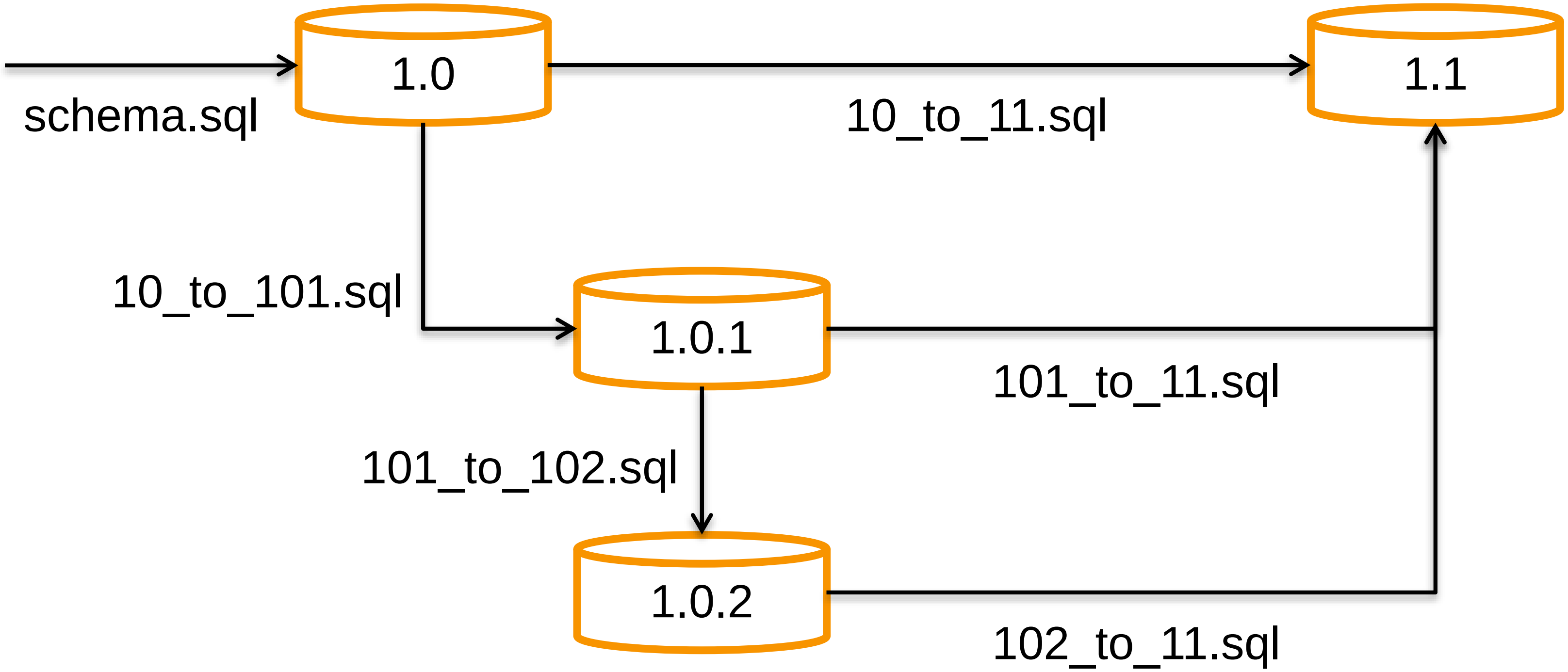
2. Manuelle SQL-Skripte



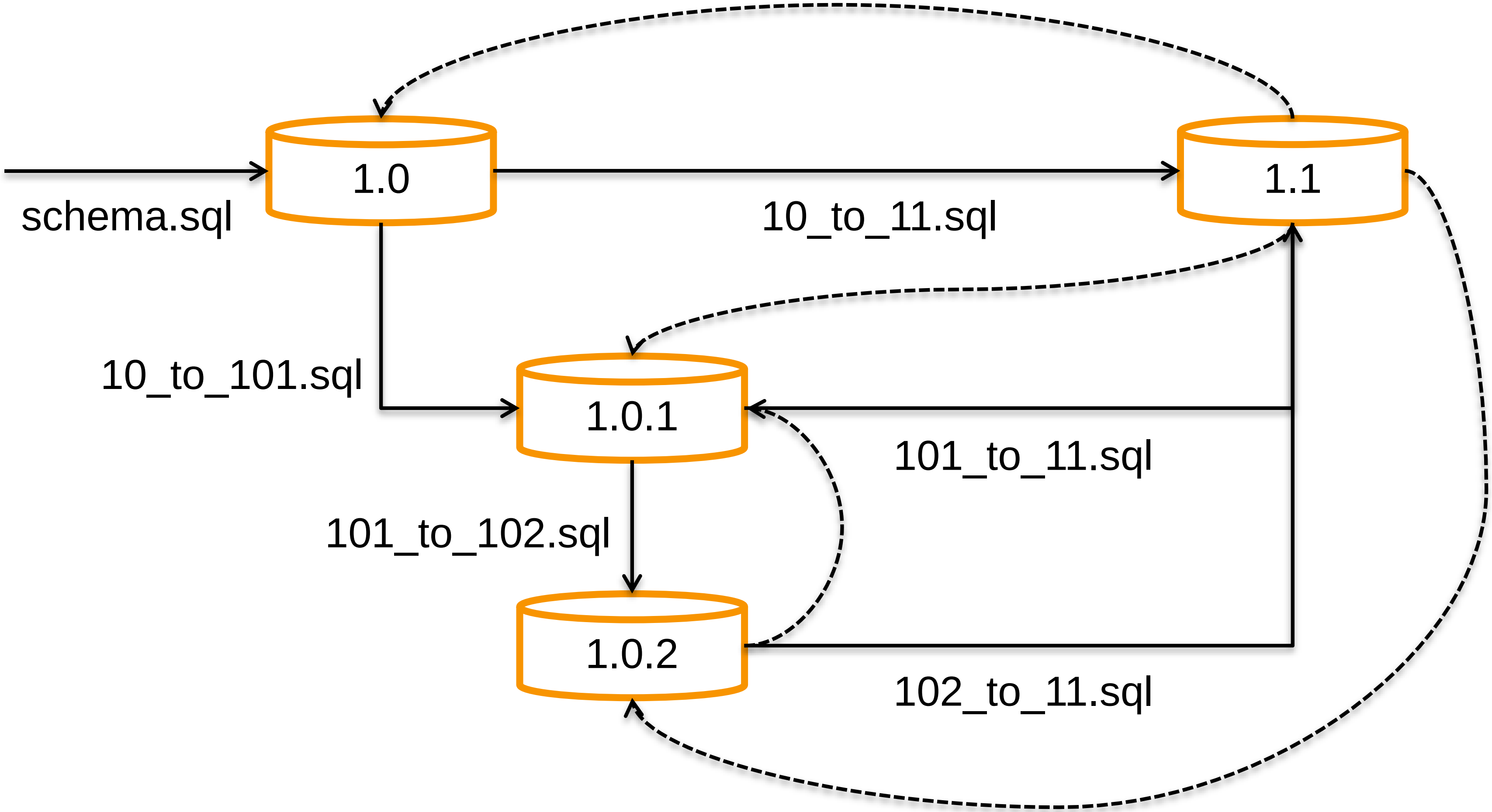
2. Manuelle SQL-Skripte



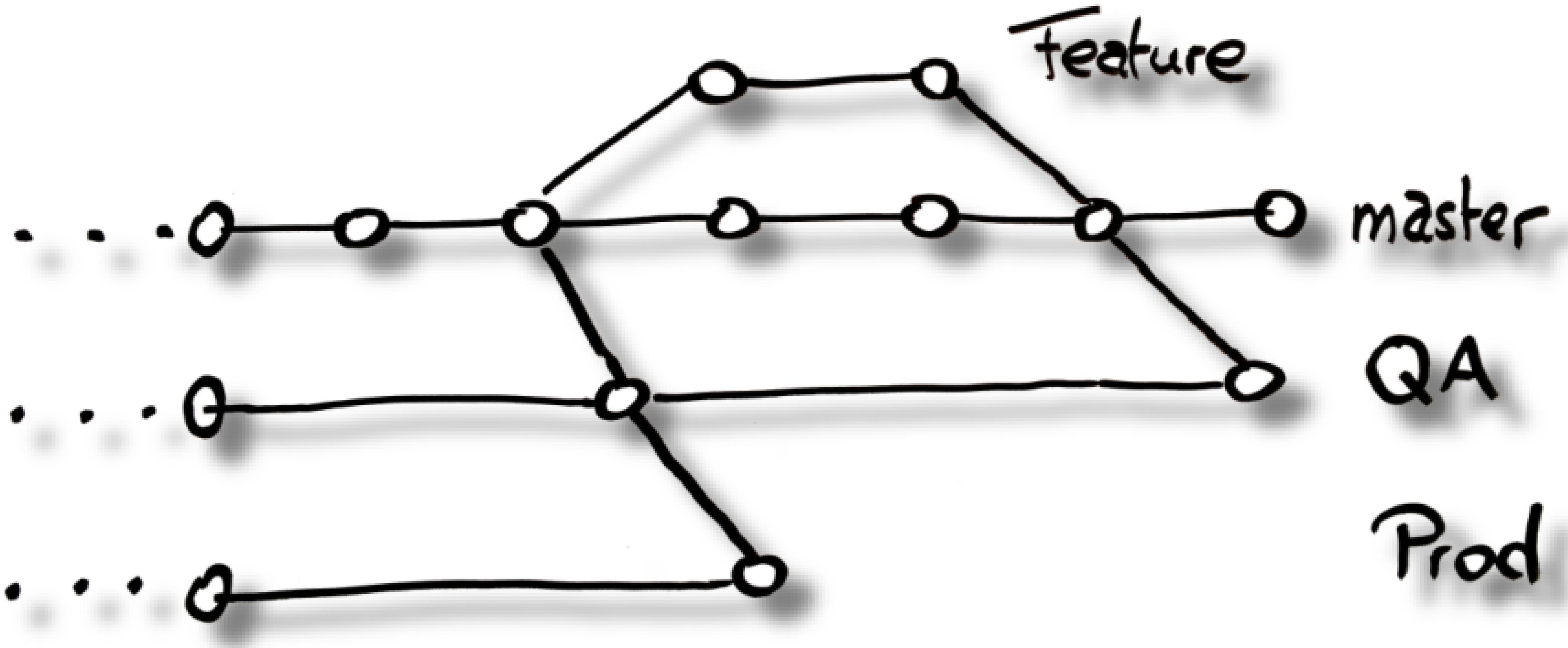
2. Manuelle SQL-Skripte



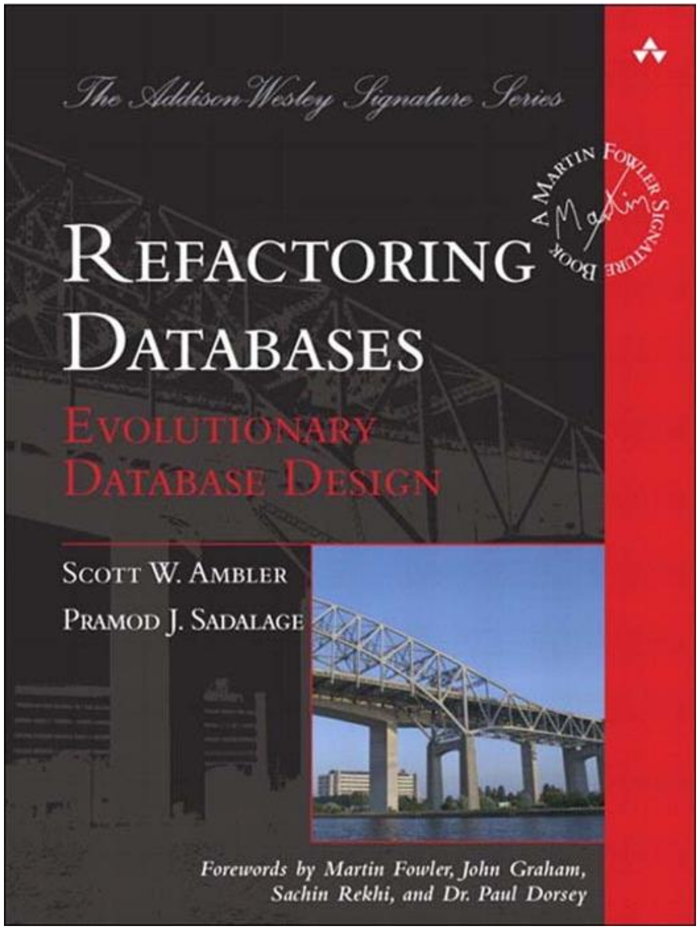
2. Manuelle SQL-Skripte



Eigentlich noch komplizierter



Die Lösung



The Addison-Wesley Signature Series

A Martin Fowler Signature Book

REFACTORING DATABASES

EVOLUTIONARY DATABASE DESIGN

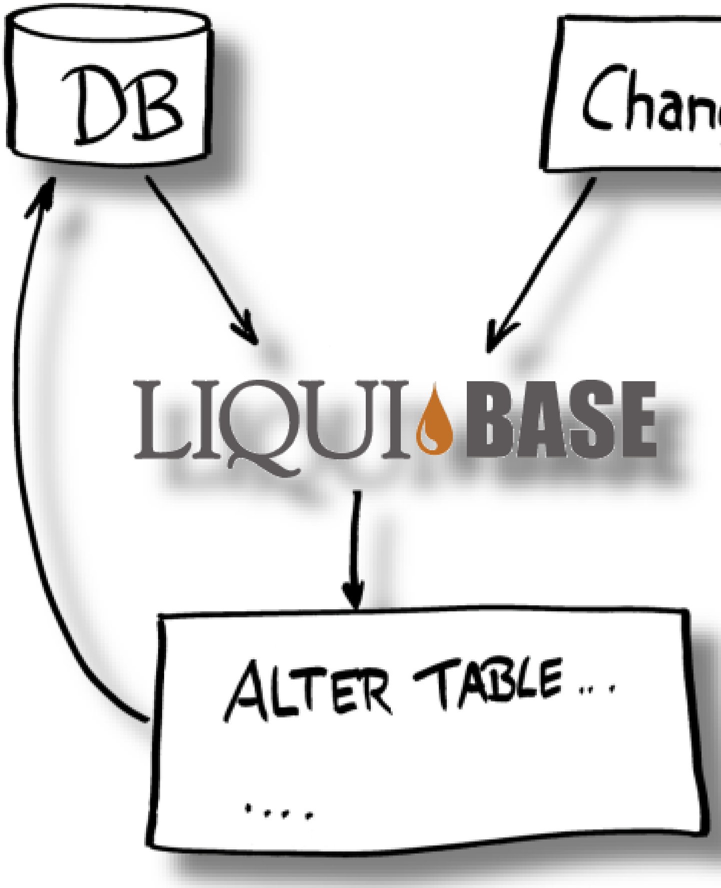
SCOTT W. AMBLER

PRAMOD J. SADALAGE

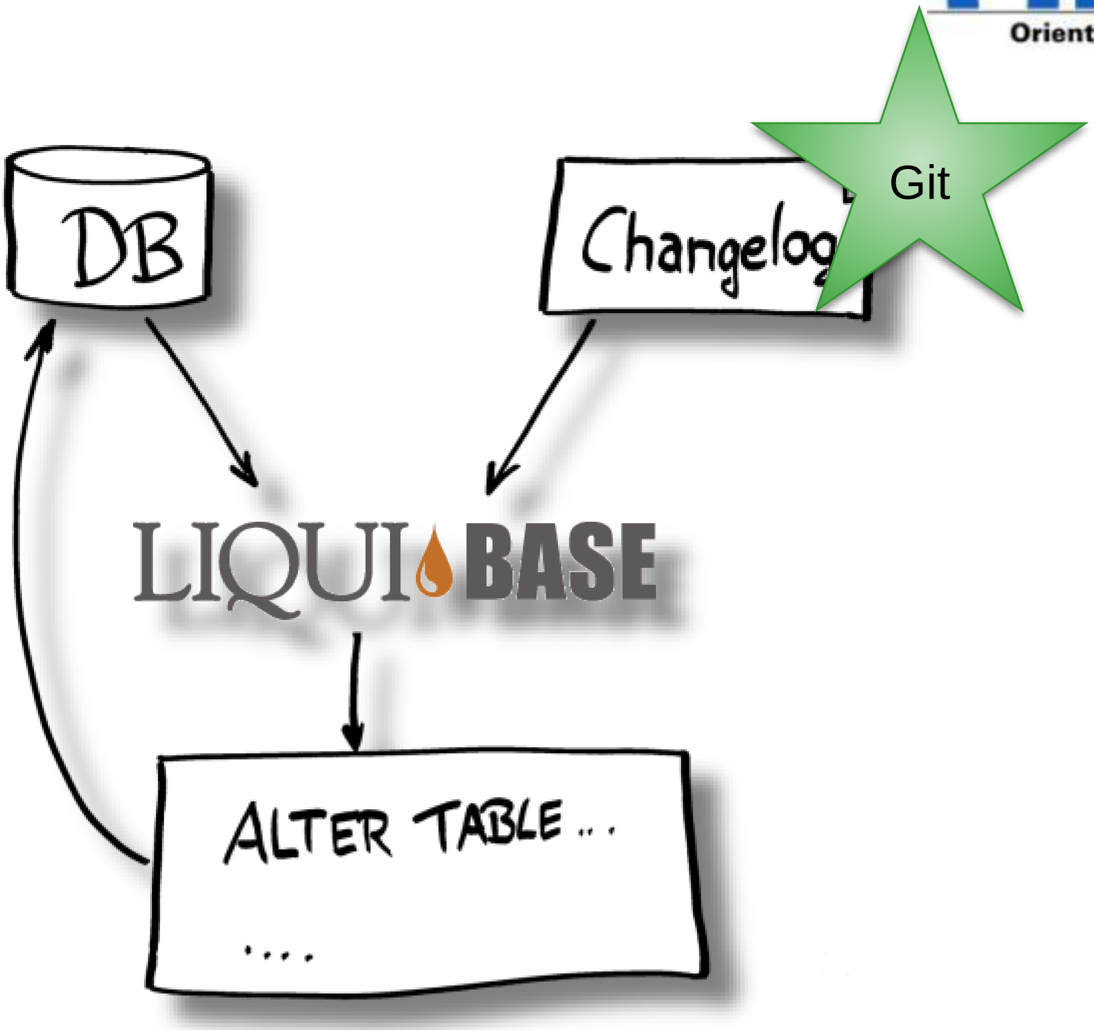
Forewords by Martin Fowler, John Graham, Sachin Rekhi, and Dr. Paul Dorsey

LIQUIBASE

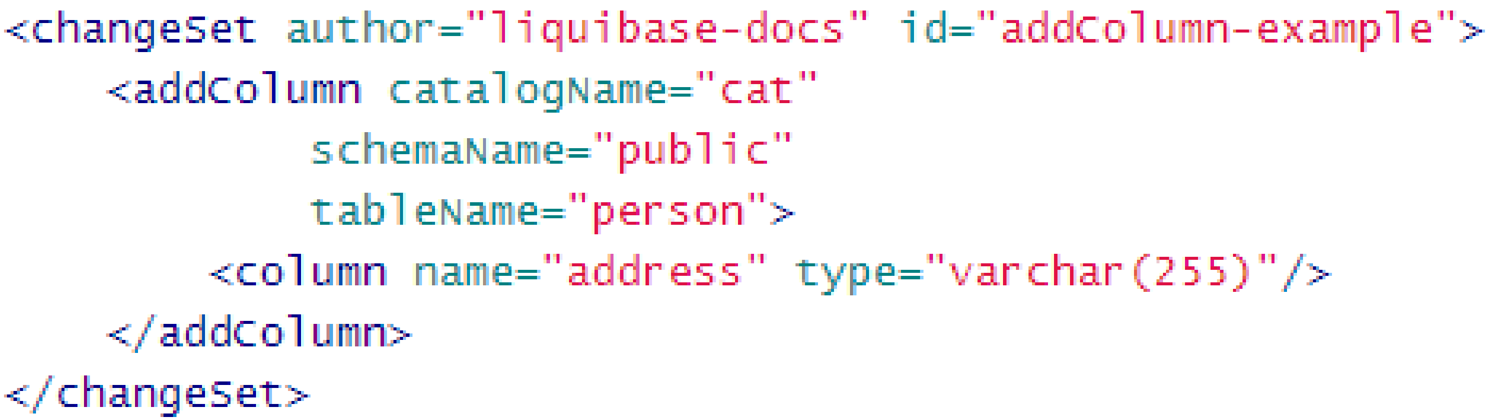
Liquibase



Liquibase



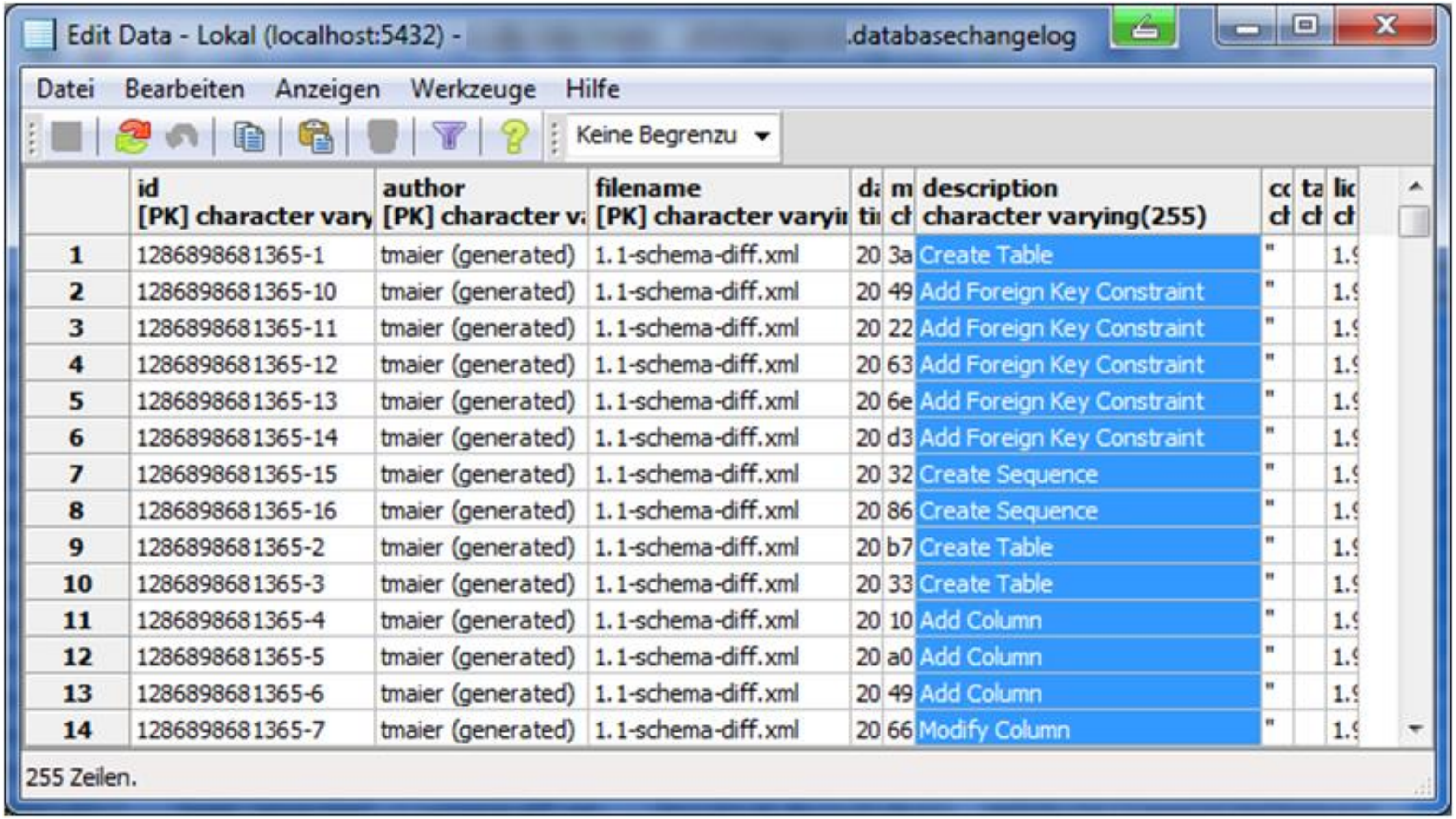
Changelog.xml



```
<changeSet author="liquibase-docs" id="addColumn-example">
    <addColumn catalogName="cat"
            schemaName="public"
            tableName="person">
        <column name="address" type="varchar(255)"/>
    </addColumn>
</changeSet>
```

Git

Metadaten in der Datenbank

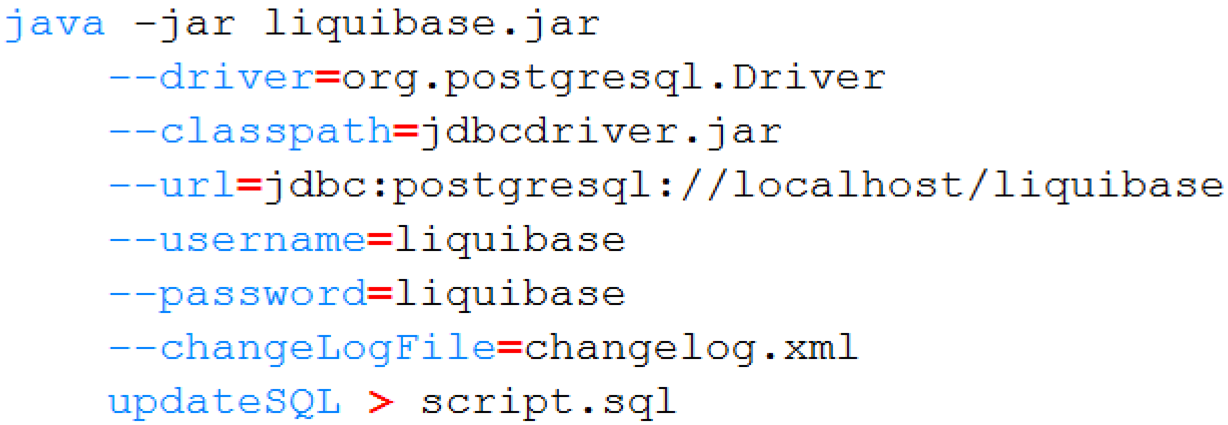

Edit Data - Lokal (localhost:5432) - [redacted] .databasechangelog
Datei Bearbeiten Anzeigen Werkzeuge Hilfe
Keine Begrenzu

	id [PK] character vary	author [PK] character v:	filename [PK] character varyi	d: ti	m cł	description character varying(255)	cc cł	ta cł	lic cł
1	1286898681365-1	tmaier (generated)	1.1-schema-diff.xml	20	3a	Create Table	"		1.9
2	1286898681365-10	tmaier (generated)	1.1-schema-diff.xml	20	49	Add Foreign Key Constraint	"		1.9
3	1286898681365-11	tmaier (generated)	1.1-schema-diff.xml	20	22	Add Foreign Key Constraint	"		1.9
4	1286898681365-12	tmaier (generated)	1.1-schema-diff.xml	20	63	Add Foreign Key Constraint	"		1.9
5	1286898681365-13	tmaier (generated)	1.1-schema-diff.xml	20	6e	Add Foreign Key Constraint	"		1.9
6	1286898681365-14	tmaier (generated)	1.1-schema-diff.xml	20	d3	Add Foreign Key Constraint	"		1.9
7	1286898681365-15	tmaier (generated)	1.1-schema-diff.xml	20	32	Create Sequence	"		1.9
8	1286898681365-16	tmaier (generated)	1.1-schema-diff.xml	20	86	Create Sequence	"		1.9
9	1286898681365-2	tmaier (generated)	1.1-schema-diff.xml	20	b7	Create Table	"		1.9
10	1286898681365-3	tmaier (generated)	1.1-schema-diff.xml	20	33	Create Table	"		1.9
11	1286898681365-4	tmaier (generated)	1.1-schema-diff.xml	20	10	Add Column	"		1.9
12	1286898681365-5	tmaier (generated)	1.1-schema-diff.xml	20	a0	Add Column	"		1.9
13	1286898681365-6	tmaier (generated)	1.1-schema-diff.xml	20	49	Add Column	"		1.9
14	1286898681365-7	tmaier (generated)	1.1-schema-diff.xml	20	66	Modify Column	"		1.9

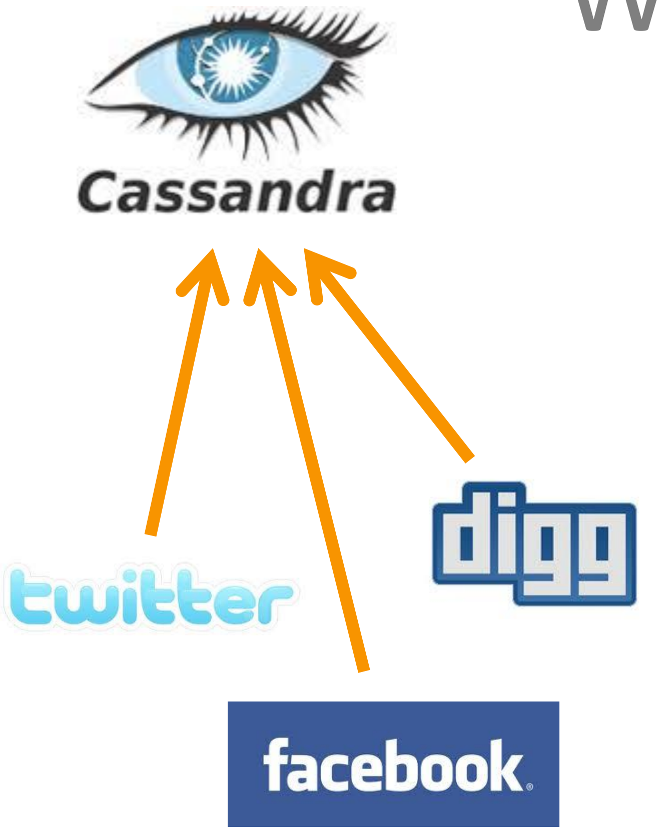
255 Zeilen.

Liquibase beim Starten der Anwendung automatisch ausführen!

```
<bean id="liquibase" class="liquibase.integration.spring.SpringLiquibase">
    <property name="dataSource" ref="dataSource" />
    <property name="changeLog" value="classpath:changelog.xml" />
    <property name="contexts" value="test, production" />
</bean>
```



```
java -jar liquibase.jar
     --driver=org.postgresql.Driver
     --classpath=jdbcdriver.jar
     --url=jdbc:postgresql://localhost/liquibase
     --username=liquibase
     --password=liquibase
     --changeLogFile=changelog.xml
     updateSQL > script.sql
```



Warum nehmen wir eigentlich keine **schemalose NoSQL DB?**

Relationale Datenbanken
skalieren nicht

Warum?

Normalisierung => Joins

Forderung nach Konsistenz => Transaktionen

NoSQL

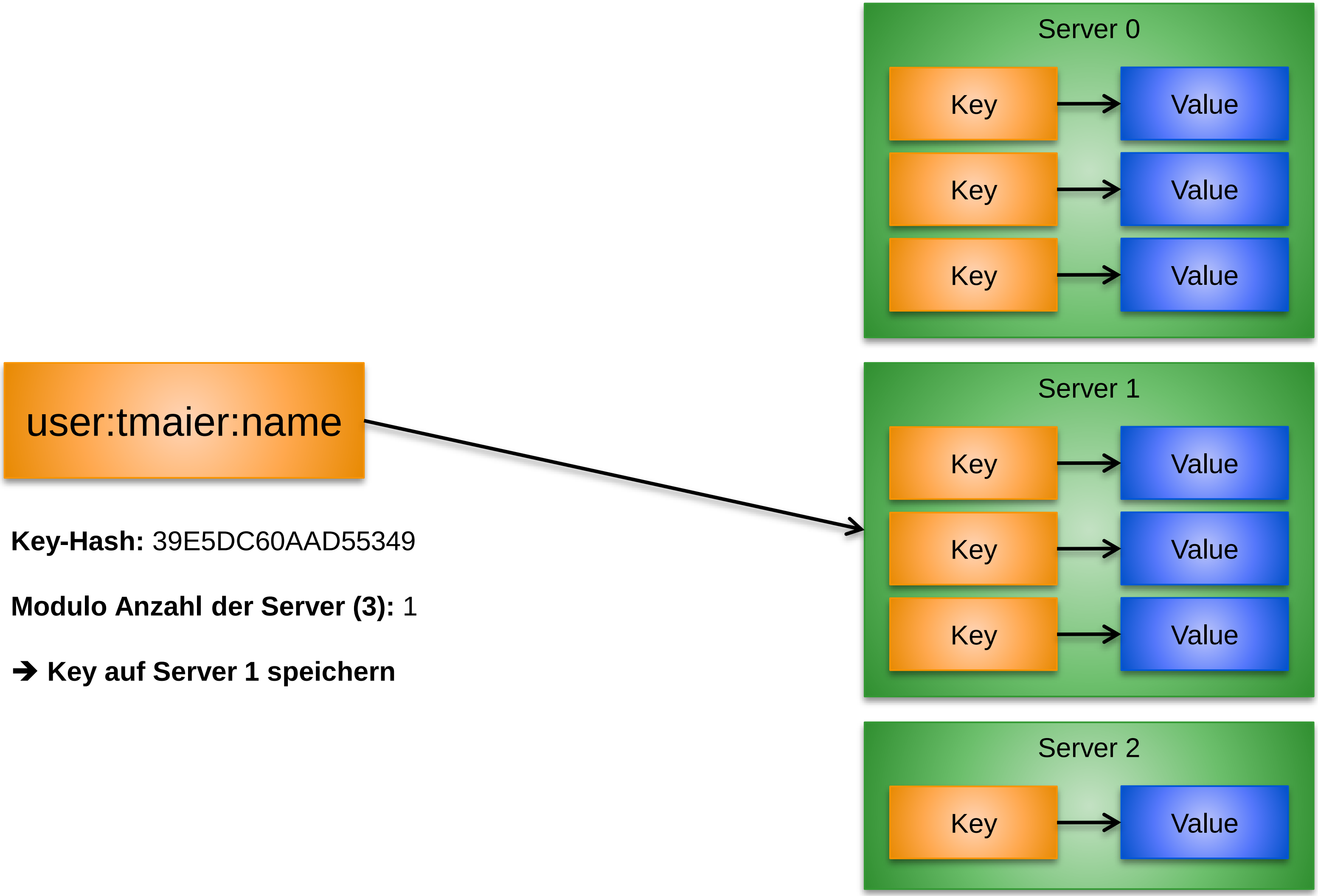
=

Relationale Datenbank
- Transaktionen - Normalisierung - Joins - Konsistenz - hartes Schema

(+) Skalierung

(+) Performance

Skalierung von Key-Value-Stores



user:tmaier:name

Server 0

Key → Value

Key → Value

Key → Value

Server 1

Key → Value

Key → Value

Key → Value

Server 2

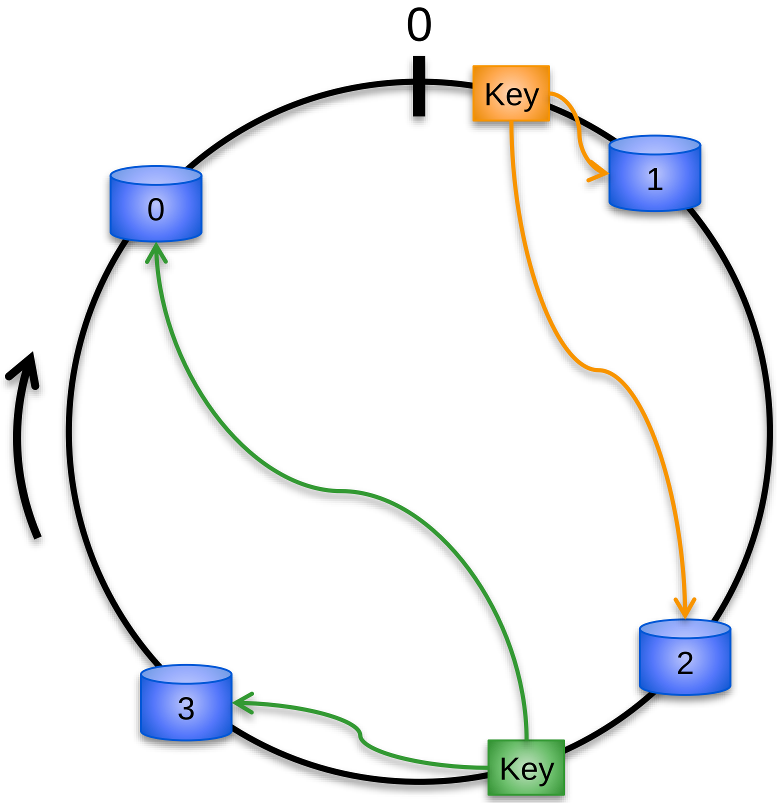
Key → Value

Key-Hash: 39E5DC60AAD55349

Modulo Anzahl der Server (3): 1

➔ Key auf Server 1 speichern

Konsistente Hash-Funktion



Noch mehr schöne neue Welt

Twitter

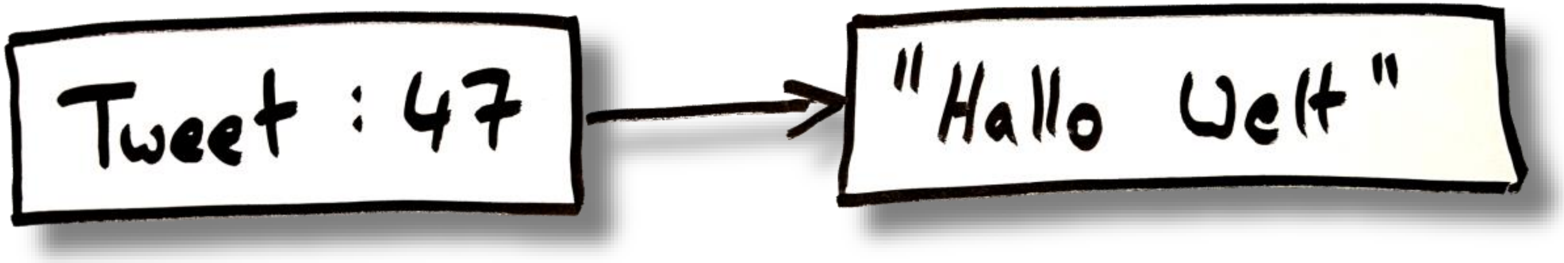
300.000.000 Benutzer

300.000 reads/sec

6000 writes/sec

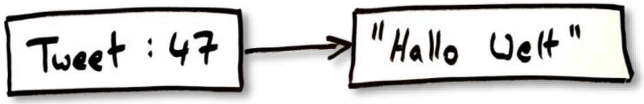
400.000.000 Tweets/day

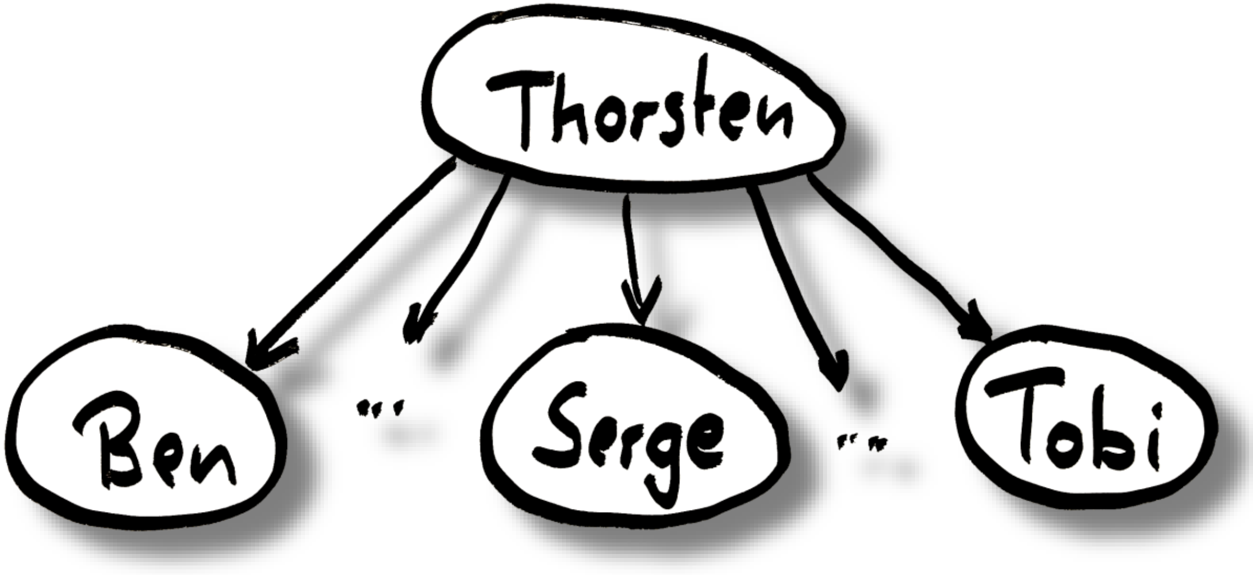
Lady Gaga hat **49.000.000** Follower



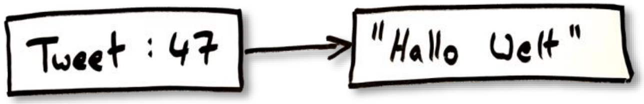
Tweet speichern in ***Key-Value-Store***

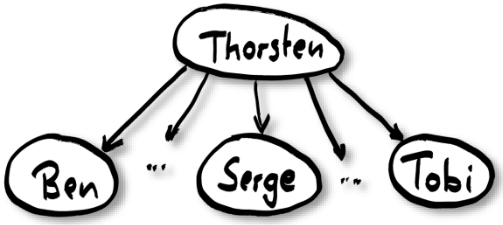
@ThorstenMaier

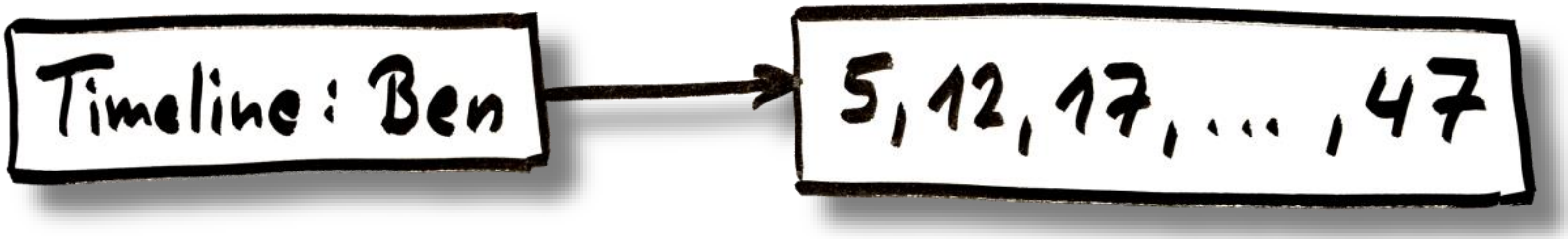




Follower aus **Graph-DB** laden

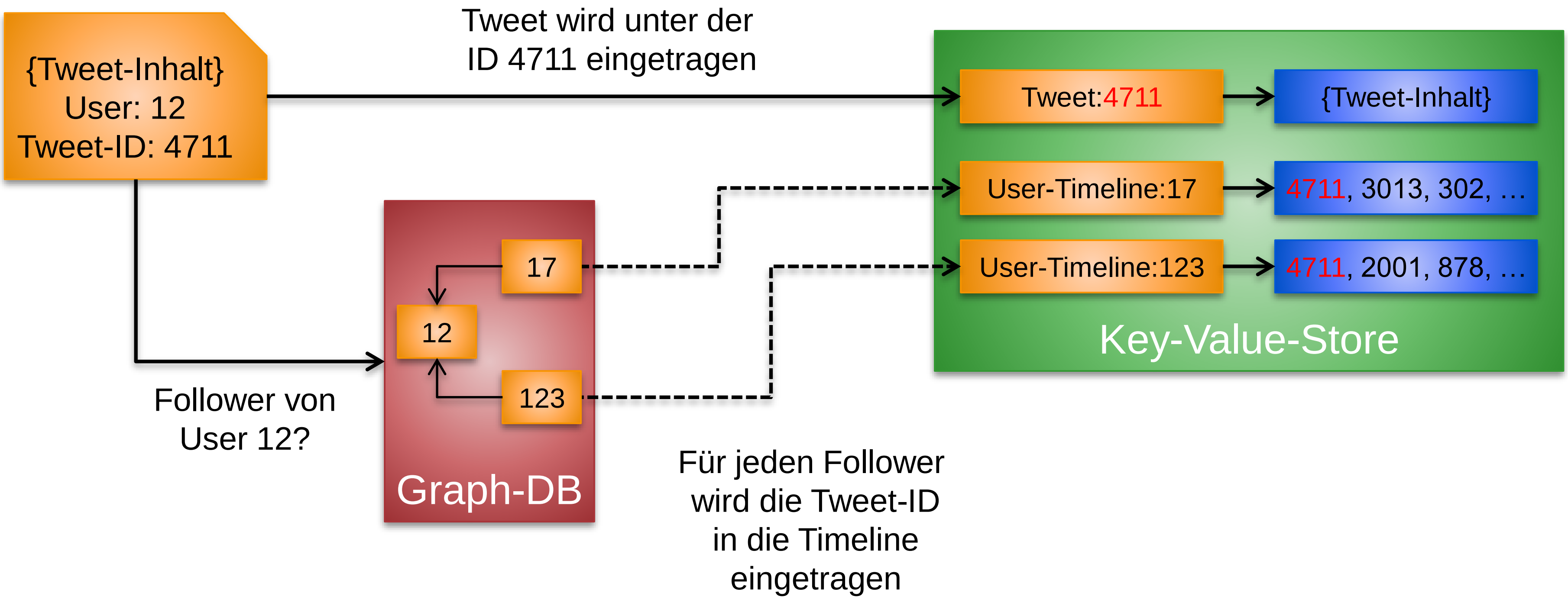






Tweet-ID in ***Key-Value-Store*** speichern

The Architecture Twitter



{Tweet-Inhalt}
User: 12
Tweet-ID: 4711

Tweet wird unter der
ID 4711 eingetragen

Tweet:4711 → {Tweet-Inhalt}

User-Timeline:17 → 4711, 3013, 302, …

User-Timeline:123 → 4711, 2001, 878, …

Key-Value-Store

Follower von
User 12?

17

12

123

Graph-DB

Für jeden Follower
wird die Tweet-ID
in die Timeline
eingetragen

Fragen ?

Orientation in Objects GmbH

Weinheimer Str. 68
68309 Mannheim

www.oio.de
info@oio.de

Vielen Dank für ihre Aufmerksamkeit !

Orientation in Objects GmbH

Weinheimer Str. 68
68309 Mannheim

www.oio.de
info@oio.de