jQAssistant

Treat your code as a Graph

HeisenBug, St.Petersburg, 2019

Oliver B. Fischer

Who am I? Кто я? Wer bin ich?

- Sometimes a developer
- Sometimes a architect
- Sometimes someone who writes Wiki pages
- Community conference organizer
- JUG lead in Berlin
- Co-Founder of the Hackergarten in Berlin
- Core developer of jQAssistant

sweblogtweets @ o.b.fischer@swe-blog.net

What is Architecture

SEI @ CMI: John Carter

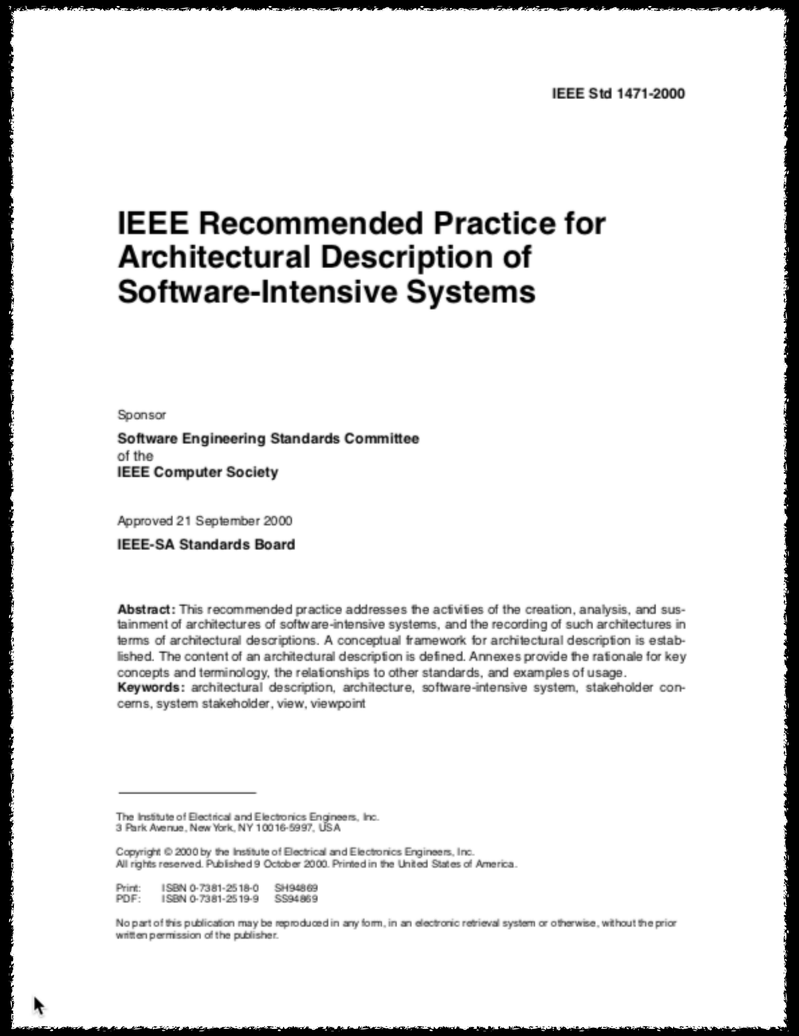
**...from the plethora of definitions the vary concept
of Software Architecture is fuzzy to the point of utter uselessness...**

A lot of folks mean:

**That peculiar consistency and homogenity of Design that would give
arise to the ease of use and reuse, which is wished for by the
developer who calls himself „The Software Architect“.**

...[which] implodes immediatly on the appointment of a second Software Architect...

What is Architecture



IEEE Std 1471-2000

IEEE Recommended Practice for Architectural Description of Software-Intensive Systems

Sponsor

Software Engineering Standards Committee
of the
IEEE Computer Society

Approved 21 September 2000

IEEE-SA Standards Board

Abstract: This recommended practice addresses the activities of the creation, analysis, and sustainment of architectures of software-intensive systems, and the recording of such architectures in terms of architectural descriptions. A conceptual framework for architectural description is established. The content of an architectural description is defined. Annexes provide the rationale for key concepts and terminology, the relationships to other standards, and examples of usage.
Keywords: architectural description, architecture, software-intensive system, stakeholder concerns, system stakeholder, view, viewpoint

The Institute of Electrical and Electronics Engineers, Inc.
3 Park Avenue, New York, NY 10016-5997, USA

Print: ISBN 0-7381-2518-0 SH94869
PDF: ISBN 0-7381-2519-9 SS94869

The fundamental organisation of a system embodied in its components, their relationships to each other, and the environment, and the principles guiding its design and evolution.

IEEE 1471-2000

What is Architecture

Software Architecture arises as soon as two or more people agree on a certain aspect of a software system or the interaction of various aspect.

Different Levels of Architecture

Software Architect

Business Architect

Confluence Architect

Data Architect

Domain Architect

Chief Data Architect

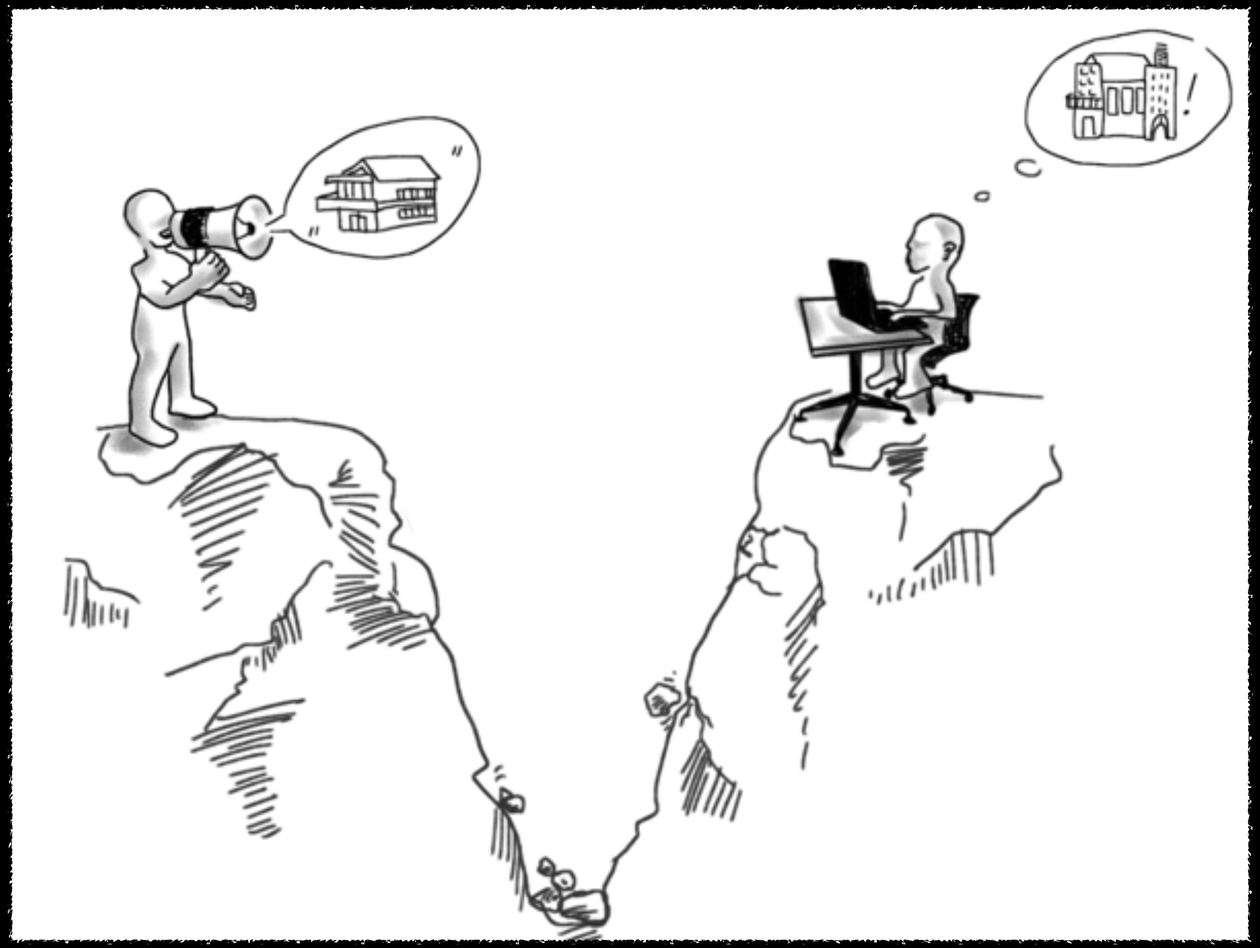
Enterprise Domain Architect

Security Architect

You!

sweblogtweets @ o.b.fischer@swe-blog.net

Mind the Gap!



The aim of software architect is not only to explain how a system should be build, but also why it must be build in this way.

sweblogtweets @ o.b.fischer@swe-blog.net

Different Levels of Architecture

sweblogtweets @ o.b.fischer@swe-blog.net

Focus on Software Architecture

sweblogtweets

@ o.b.fischer@swe-blog.net

Rules are everywhere

Designing Software is creating rules how a system should be build, how it should work and how it should evol.

Software Architecture is whole set of rules which must be applied to a system.

Architectural Rules

```
me@mac:jqa/jqa-uber-parent> ls -lh
total 224
-rw-r--r--  1 fischoli  164175240    34K 14 Mai  2016 LICENSE
-rw-r--r--  1 fischoli  164175240   574B  4 Mai  2018 jqa-uber-parent.iml
-rw-r--r--  1 fischoli  164175240    25K 27 Apr 23:17 pom.xml
-rw-r--r--  1 fischoli  164175240   223B 12 Sep  2016 readme.adoc
drwxr-xr-x  3 fischoli  164175240    96B  5 Mai 13:03 target
```

Rules can be simple

Each Git repository must contain a `readme.adoc` with a short project description.

Architectural Rules

```
@ConfigurationProperties("tomcat")
@Validated
public class TomcatProperties {
}
```

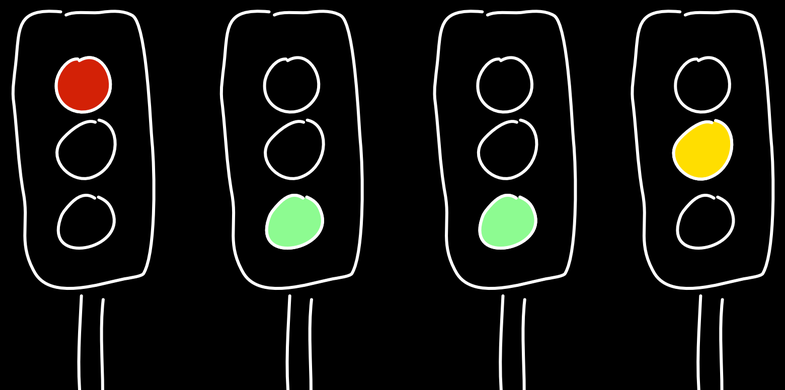
A Rule can be complex

All configuration classes in a Spring project must be the xyz.configuration package, must be validated, their name must end with Properties and start with the prefix of the properties to bind to the concrete instance of the class.

Architectural Rules

A Rule can be fuzzy

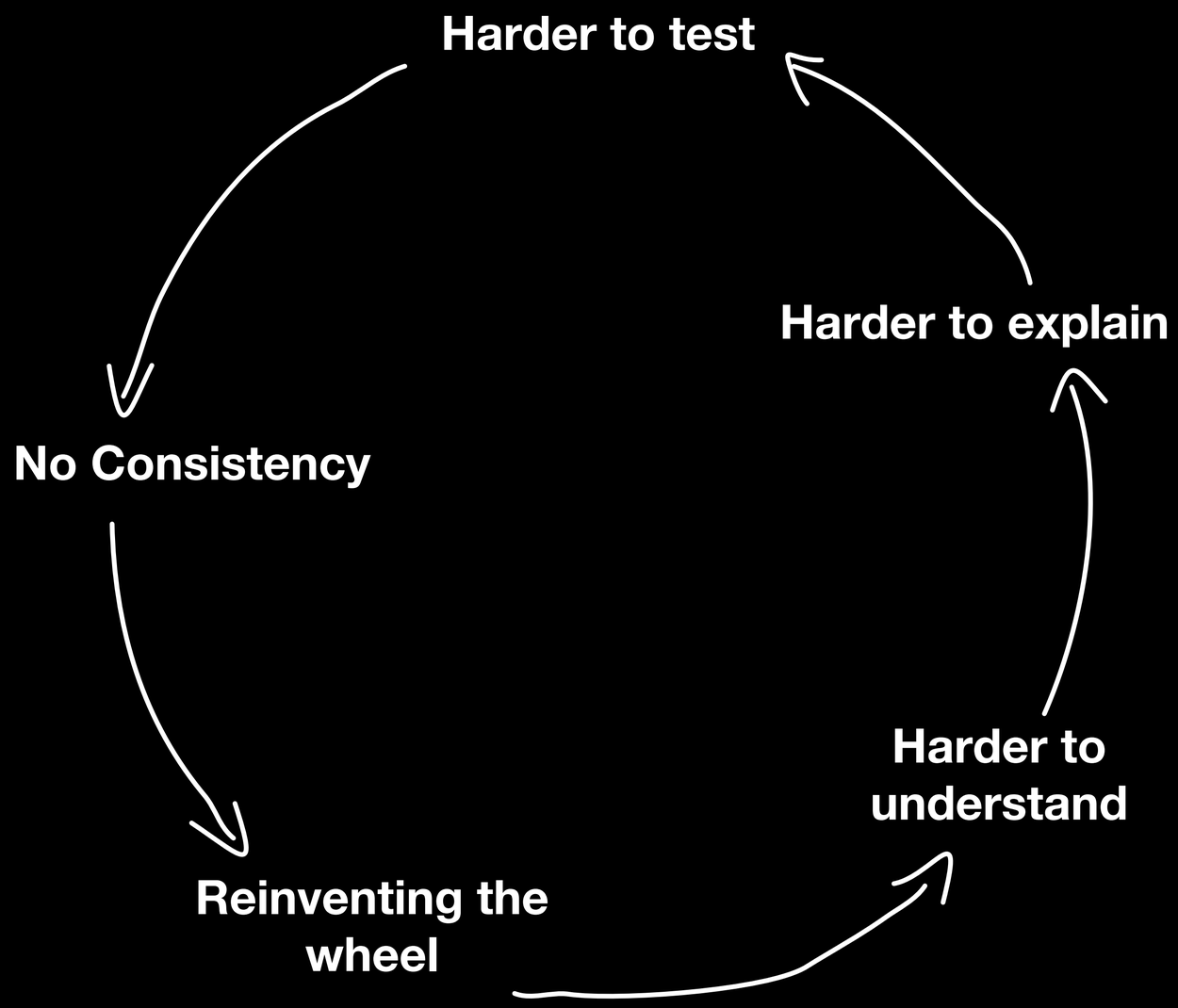
Each System must monitor the availability of its downstream systems. In the case of the unavailability of a downstream system, the system must avoid the loss of data.



sweblogtweets @ o.b.fischer@swe-blog.net


Tap here

Circle of Death



How to prove our Architecture?

w(° o °)w

I Don't Always Test My Code

But When I Do, It's In Production

How to prove our Architecture?

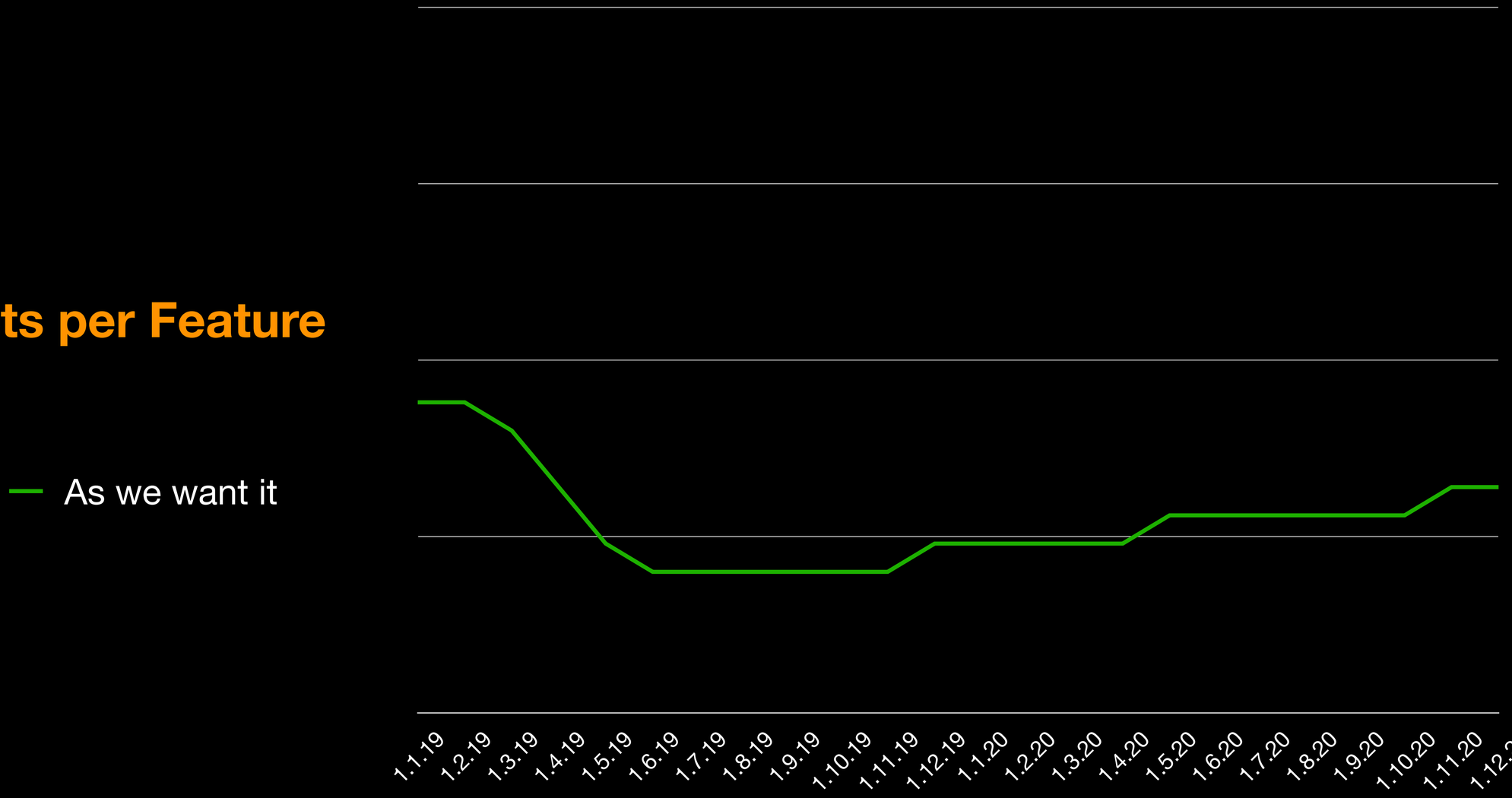
How to prove our Architecture?



sweblogtweets @ o.b.fischer@swe-blog.net

Benefit of a well-structured System

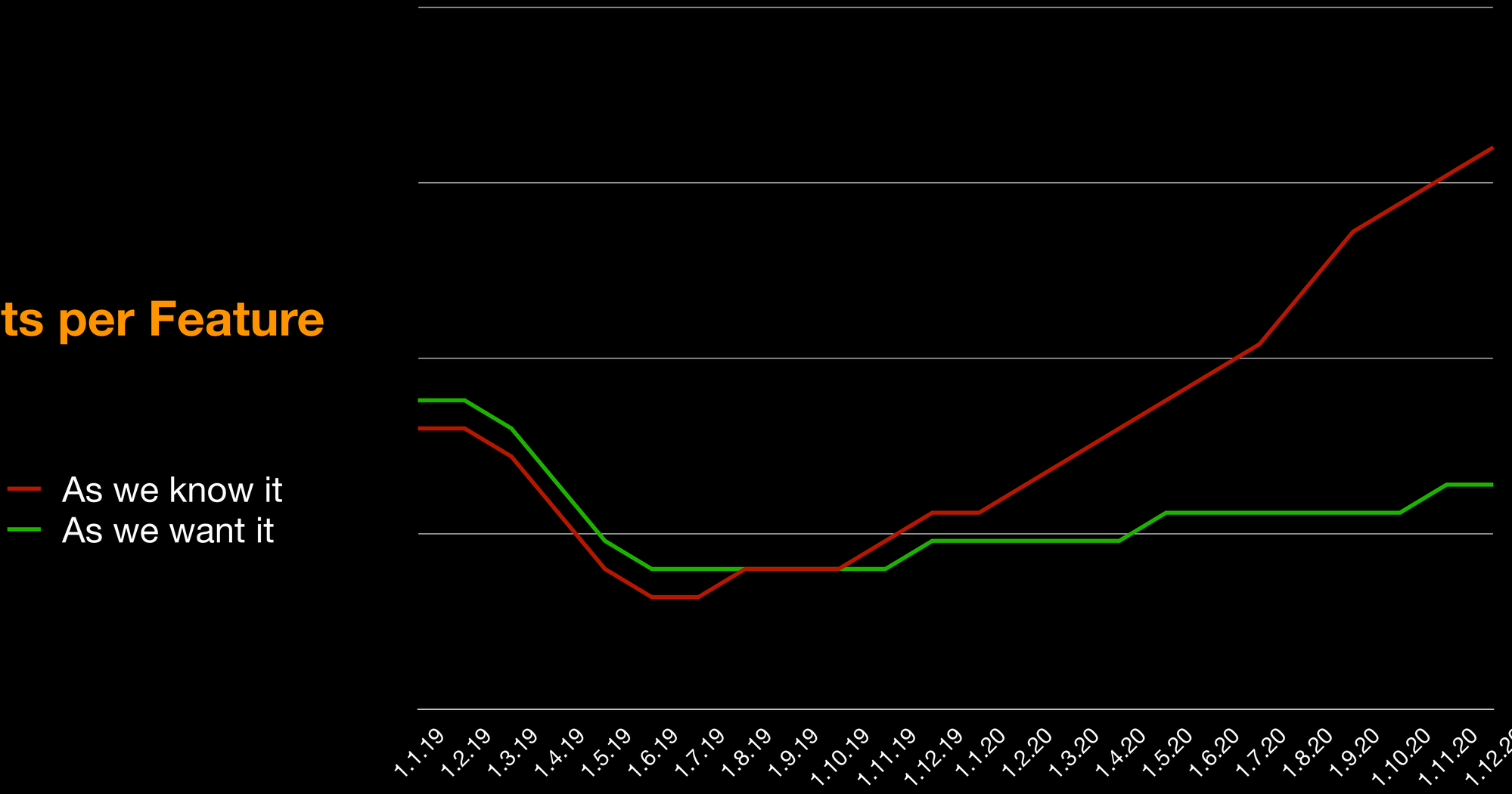
Costs per Feature



sweblogtweets @ o.b.fischer@swe-blog.net

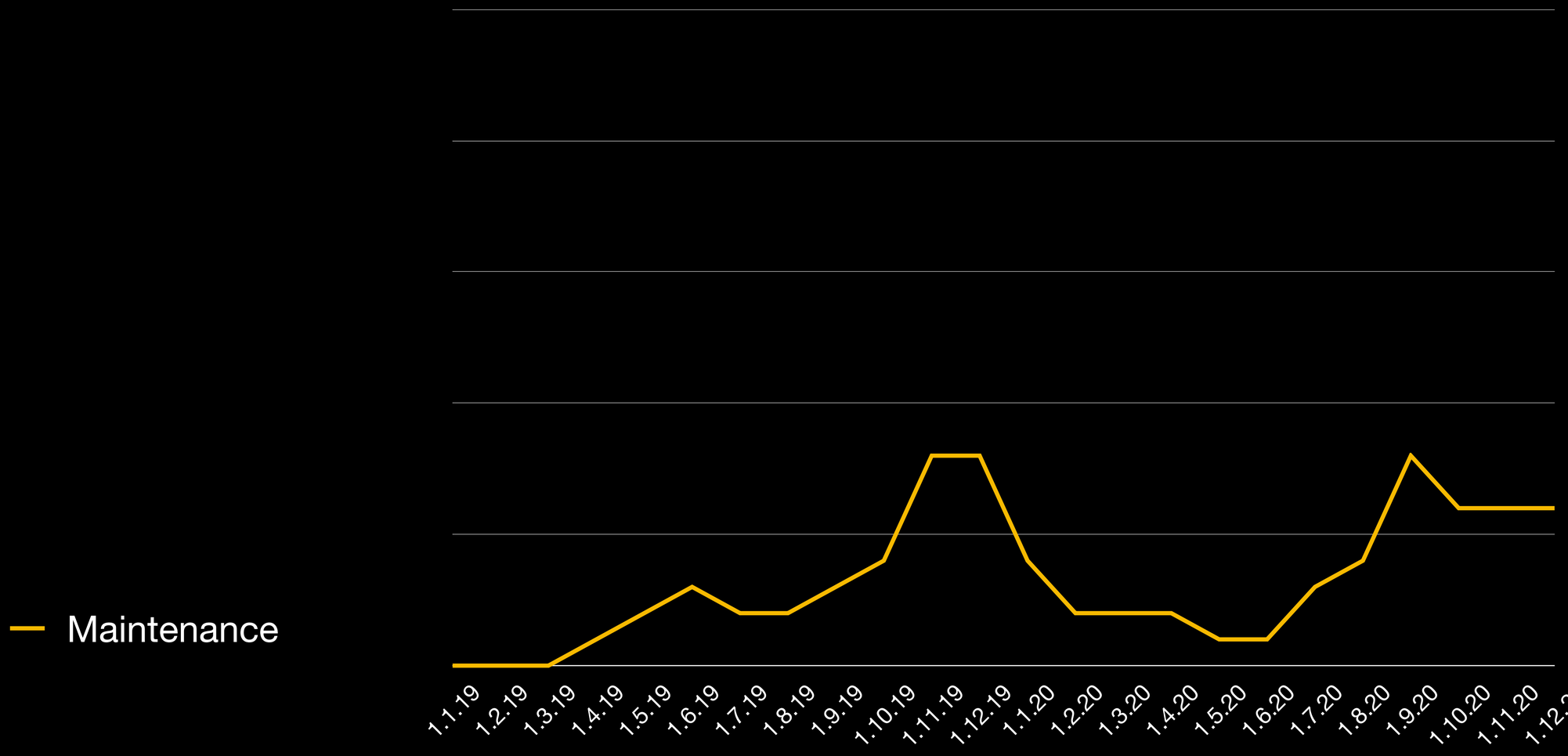
Benefit of a well-structured System

Costs per Feature



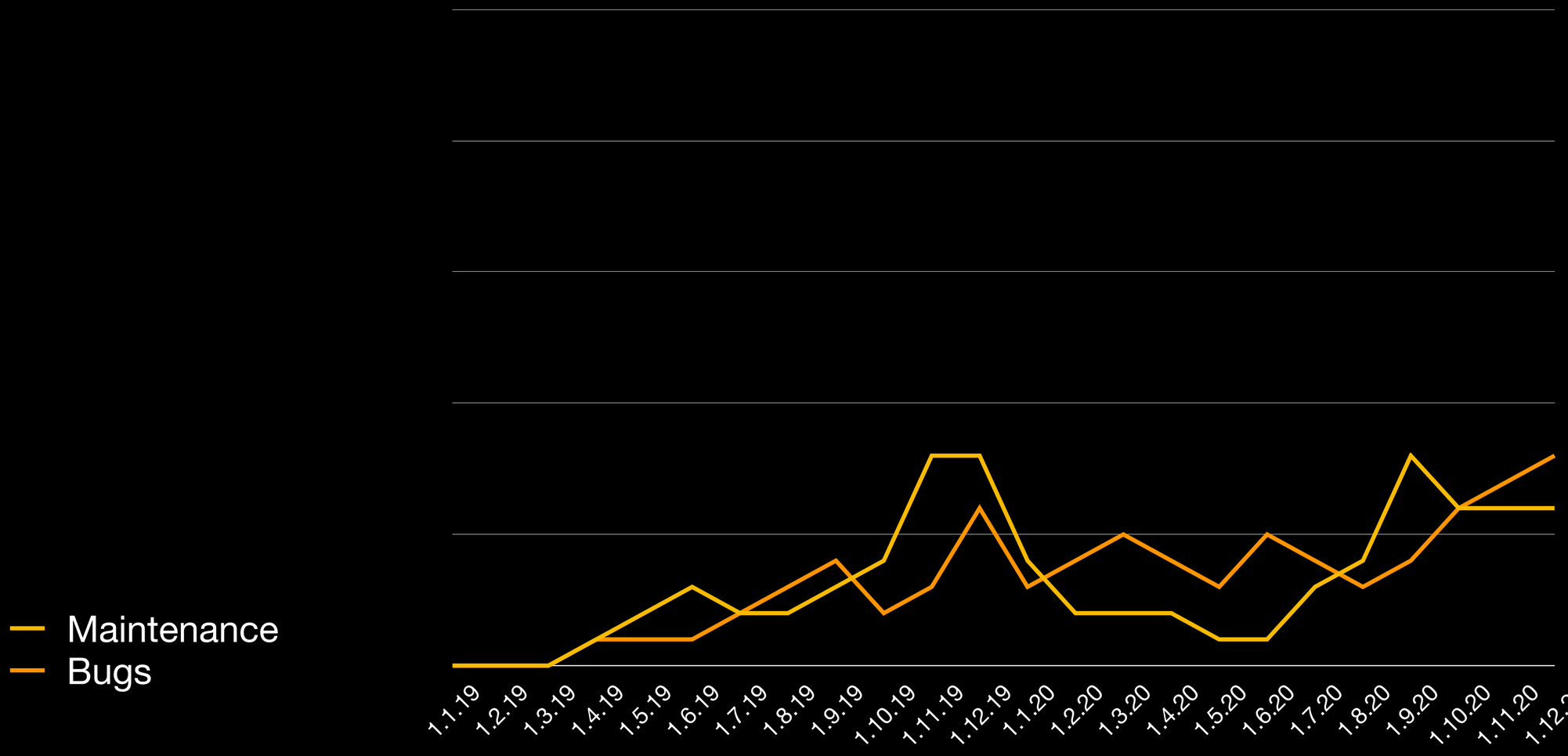
sweblogtweets @ o.b.fischer@swe-blog.net

Benefit of a well-structured System



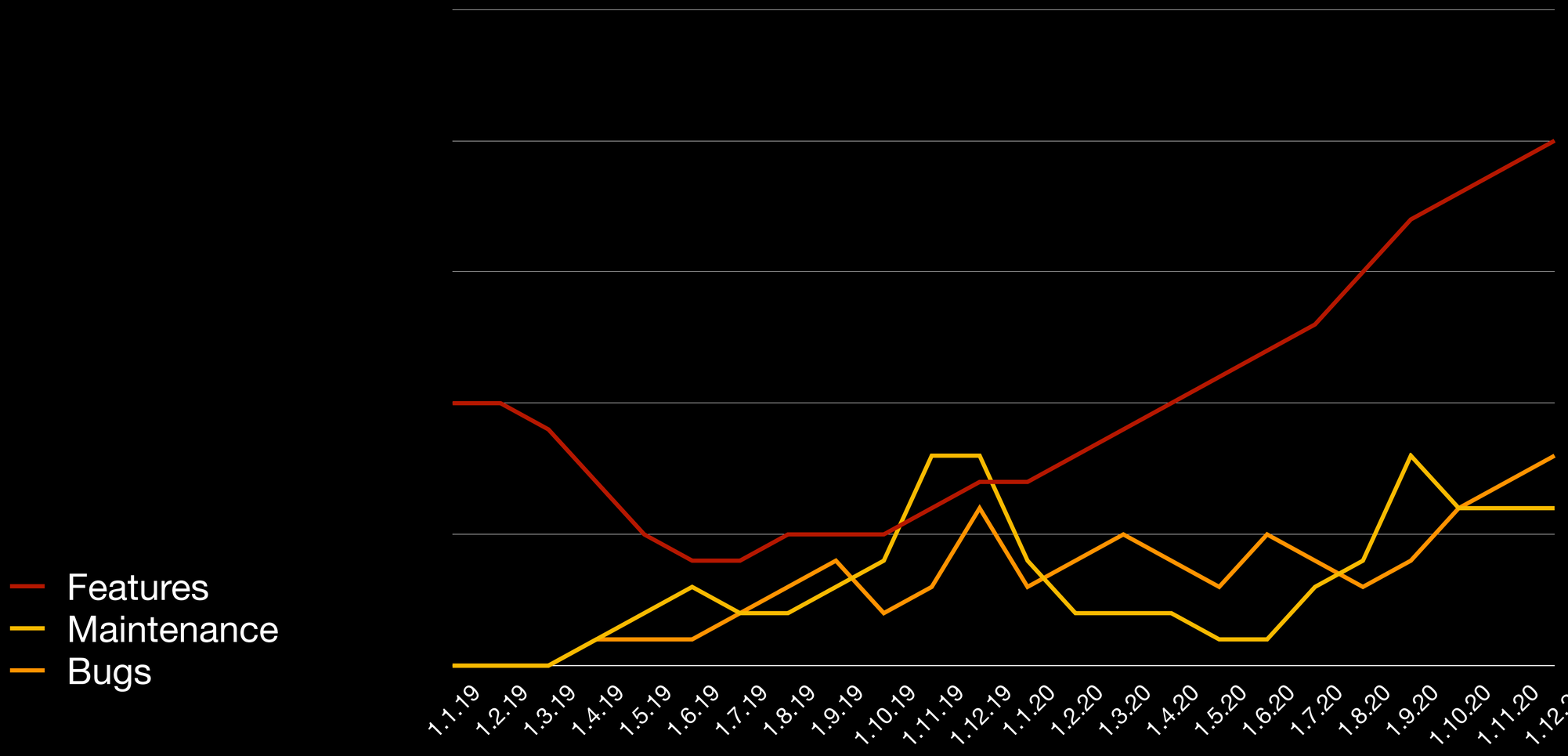
sweblogtweets @ o.b.fischer@swe-blog.net

Benefit of a well-structured System



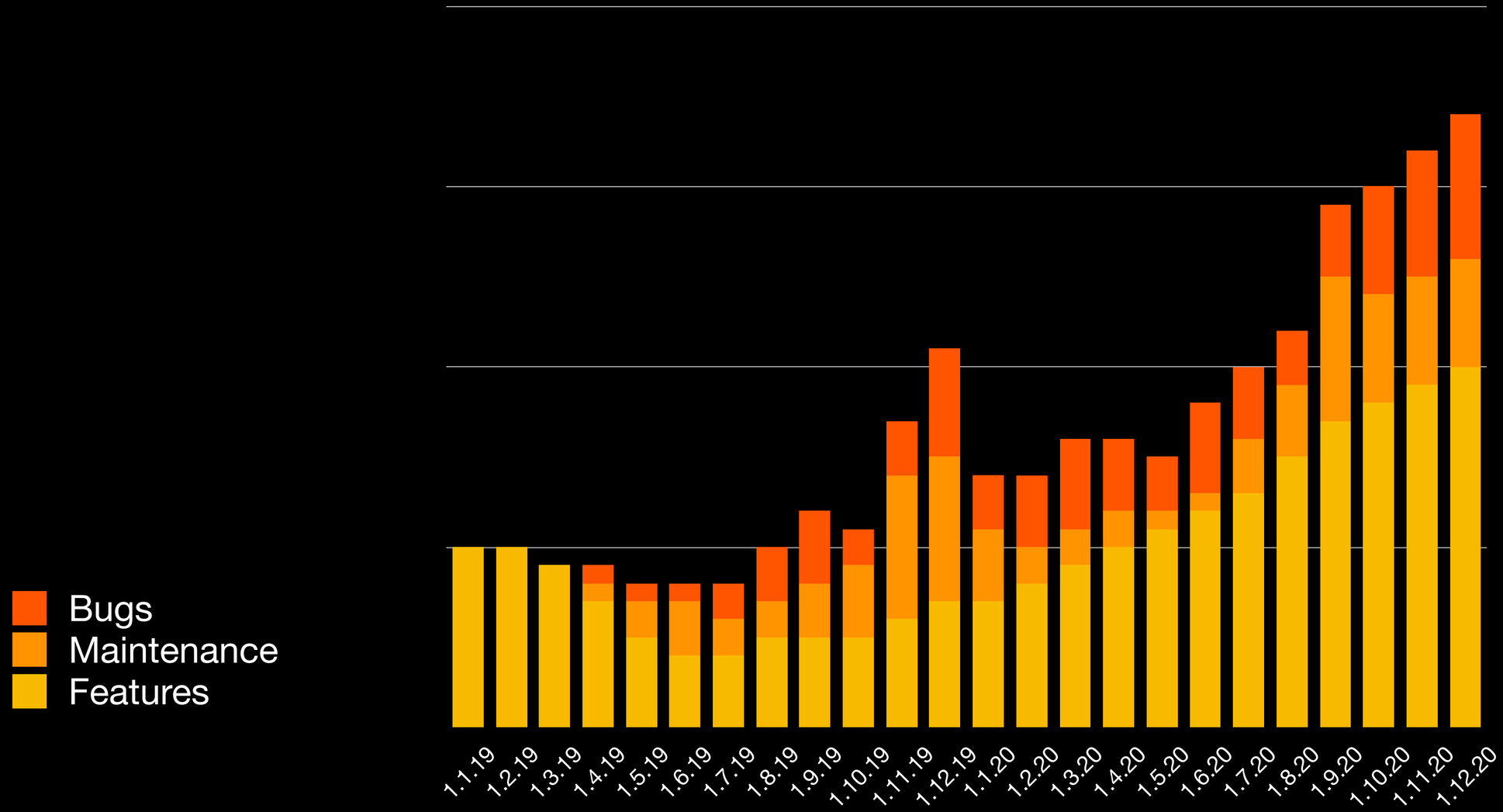
sweblogtweets @ o.b.fischer@swe-blog.net

Benefit of a well-structured System



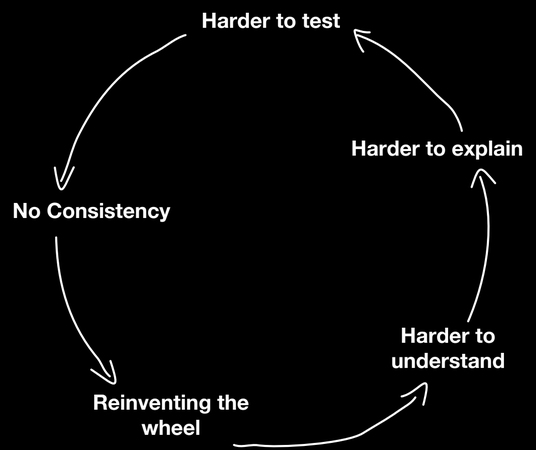
sweblogtweets @ o.b.fischer@swe-blog.net

Cost Distribution over Time



sweblogtweets @ o.b.fischer@swe-blog.net

The Advantage of High Quality

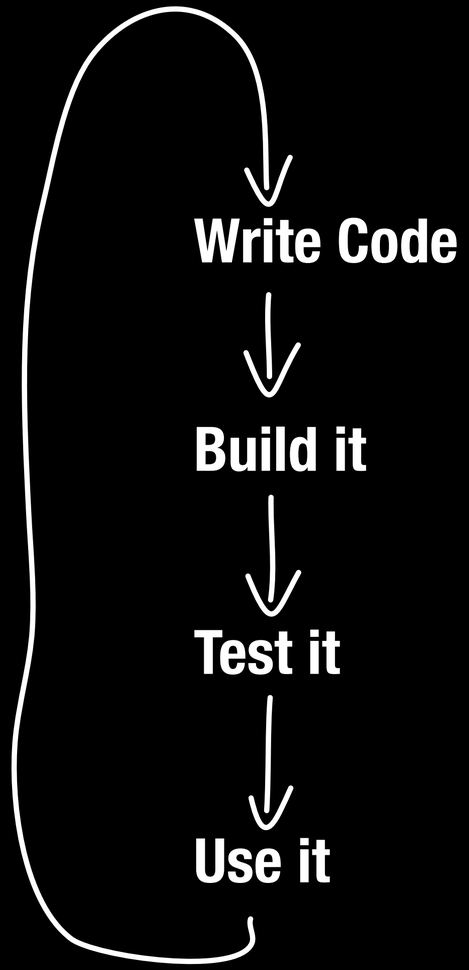


1. Easy to explain
2. Easy to understand
3. Easy to test
4. Easy to extend
5. High consistency

sweblogtweets @ o.b.fischer@swe-blog.net

How do we it now?

Extending Spotbugs




```java
@Override
public void sawOpcode(int seen) {
    if (seen != Const.GETSTATIC) {
        return;
    }
    if (getClassConstantOperand().equals("java/lang/System")
            && getNameConstantOperand().equals("out")) {

        // report bug when System.out is used in code
        BugInstance bug = new BugInstance(this, "MY_BUG", NORMAL_PRIORITY)
                .addClassAndMethod(this)
                .addSourceLine(this, getPC());
        bugReporter.reportBug(bug);
    }
}
```

The right tool is needed

sweblogtweets

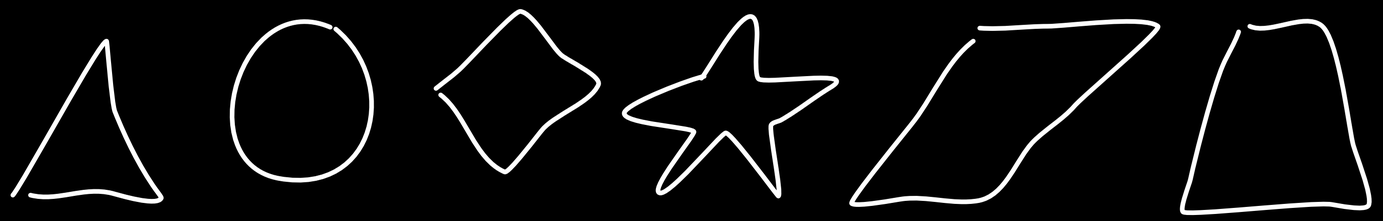
@ o.b.fischer@swe-blog.net

The right Tool should be…

1. Easy to extend
2. Able to combine information from different sources
3. Easy to express custom rules and constraints

A Uniform Data Model

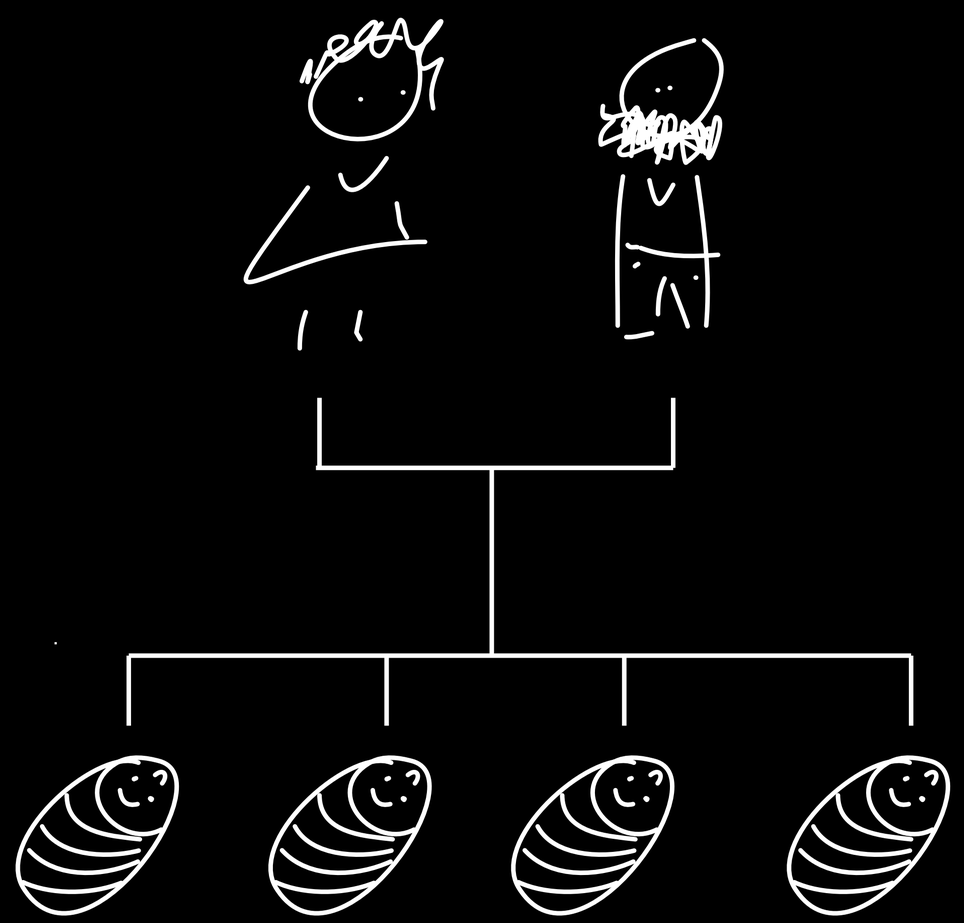
If we could find a representation for each software artifact and connect all of these models with each other, it would be possible to search for structures, patterns, and to gather metrics with tool.




Everything can be represented by a graph

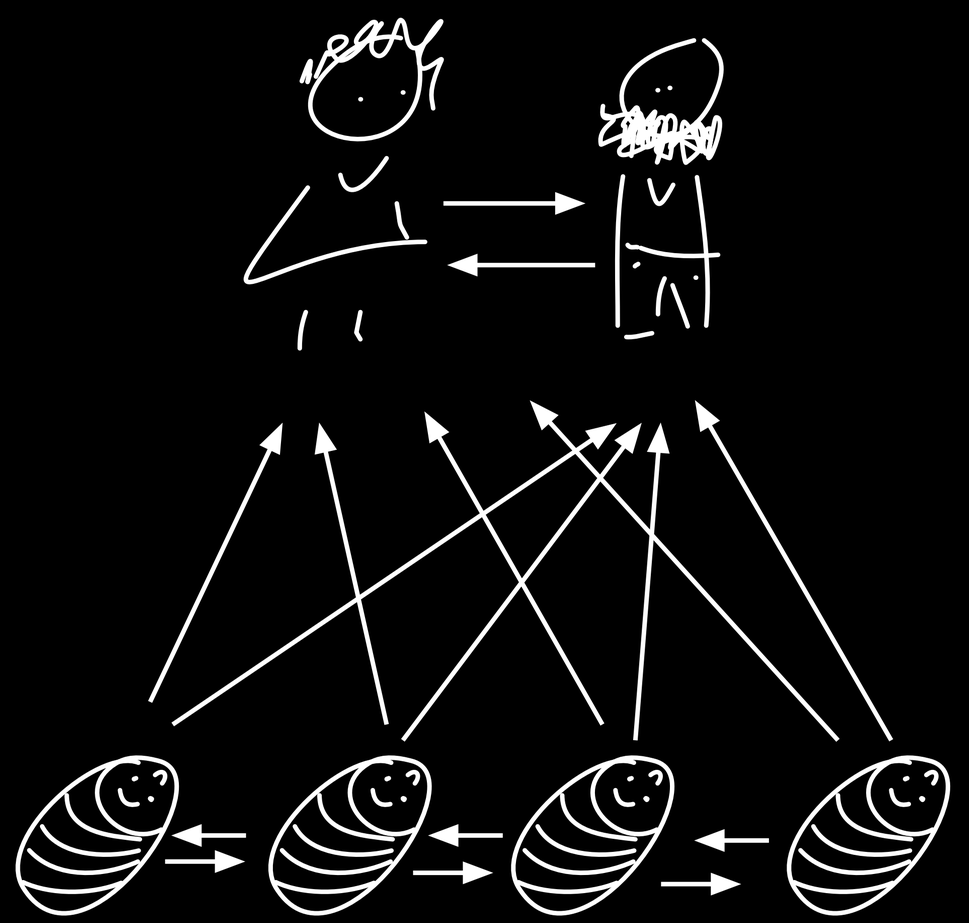
Graph Representations

Our personal relationship within our family can be represented as a graph



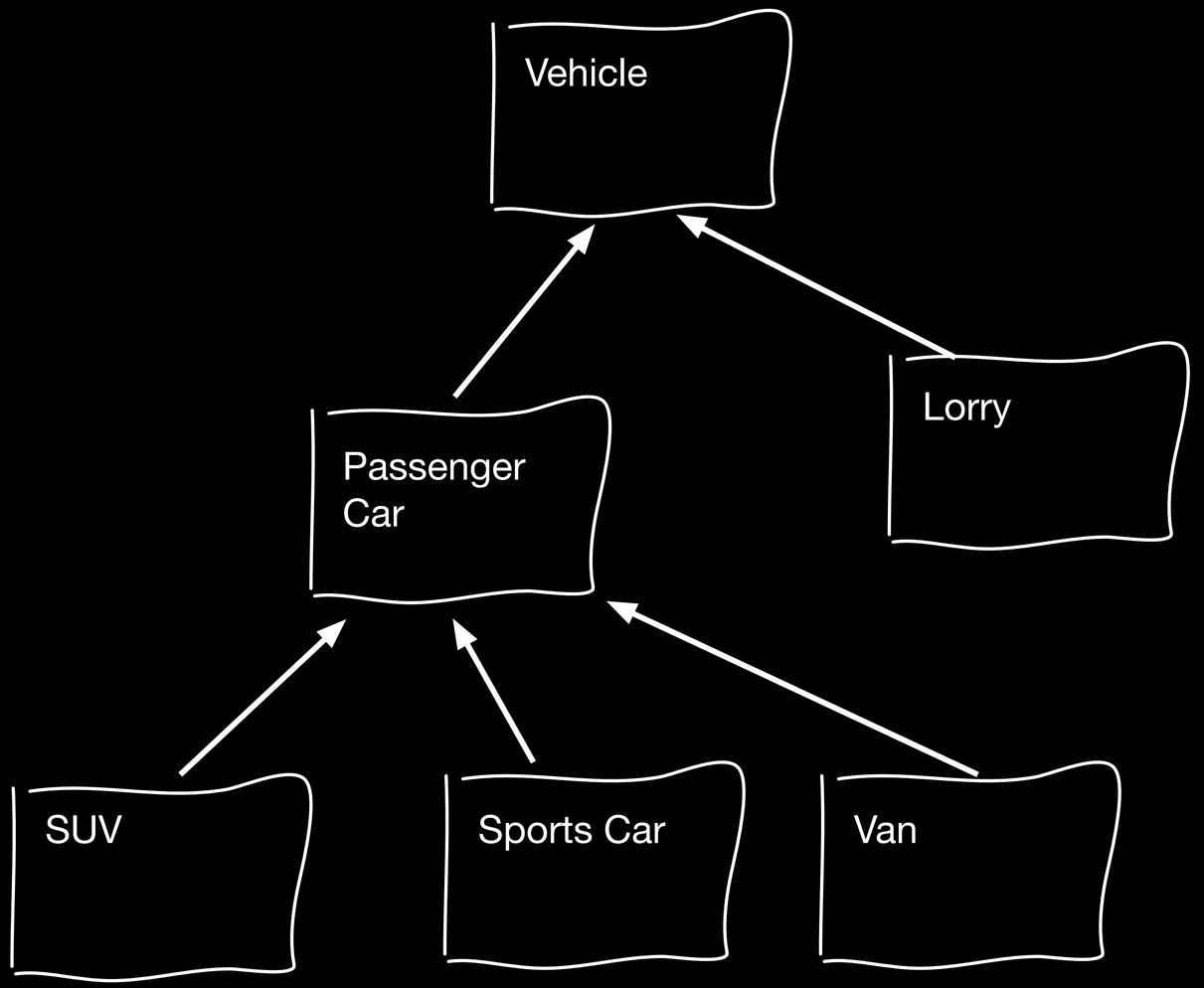
Graph Representations

Our personal relationship within our family can be represented as a graph



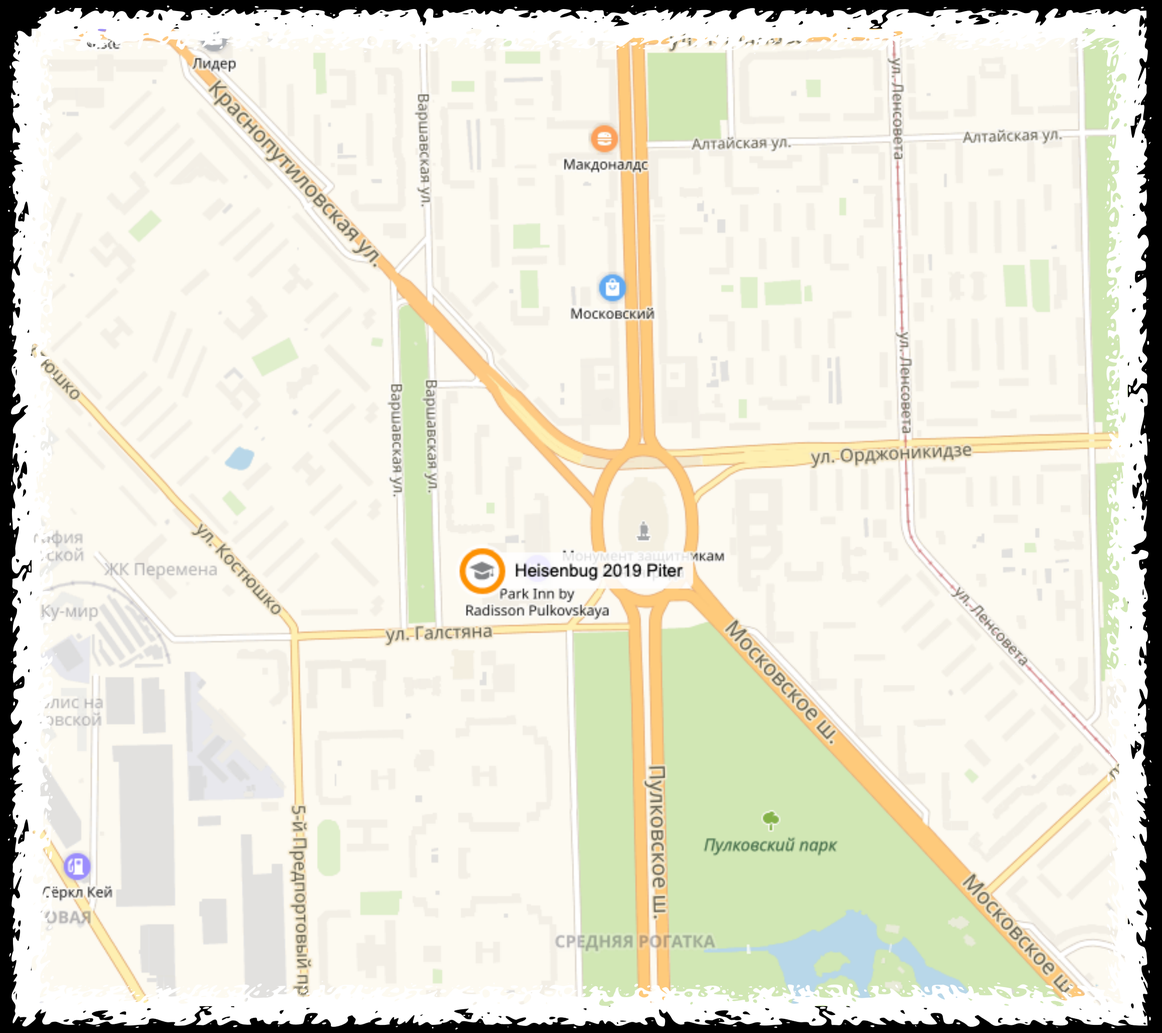
Graph Representations

A UML class diagram is also a specialized graph



Graph Representations

A map is also a graph.



- Initiated in March 2013
- First stable release in April 2015
- Current version is 1.6.0, released in December 2015
- Neo4j 3 used as graph database
- License is GPLv3

The Idea

1. Be able to scan every possible software artifact
2. Provide the possibility to analyze the result of the scan
3. Write Reports
4. Enable the integration in a CD/CI pipeline

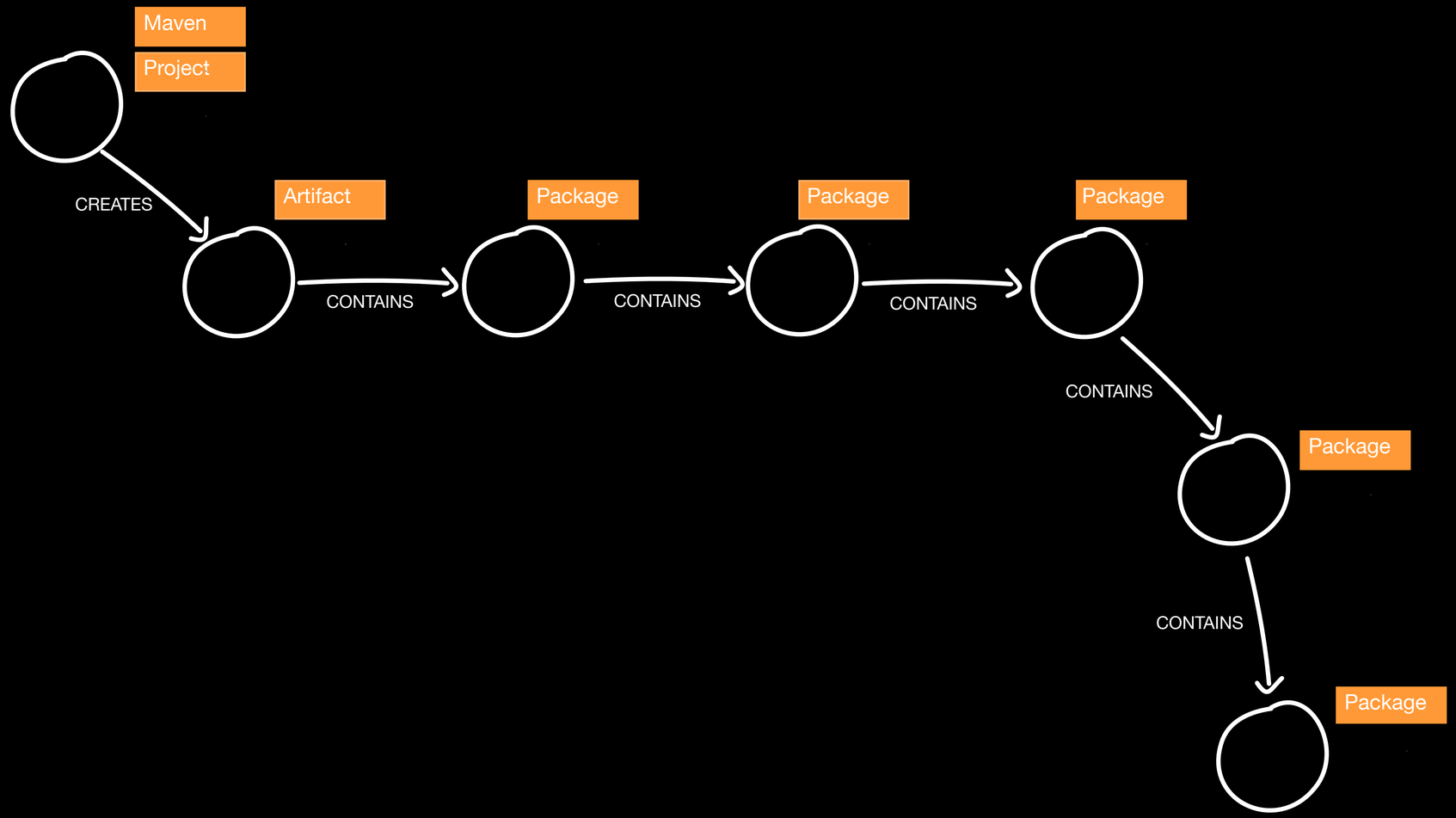
Under the hood

1. Graph Database
2. Public available since 2010
3. Open Source with dual license
4. Open Source Edition licensed under GPLv3
5. Commercial license available

A Maven Project

sweblogtweets @ o.b.fischer@swe-blog.net

A Maven Project



sweblogtweets @ o.b.fischer@swe-blog.net

An JPA Entity

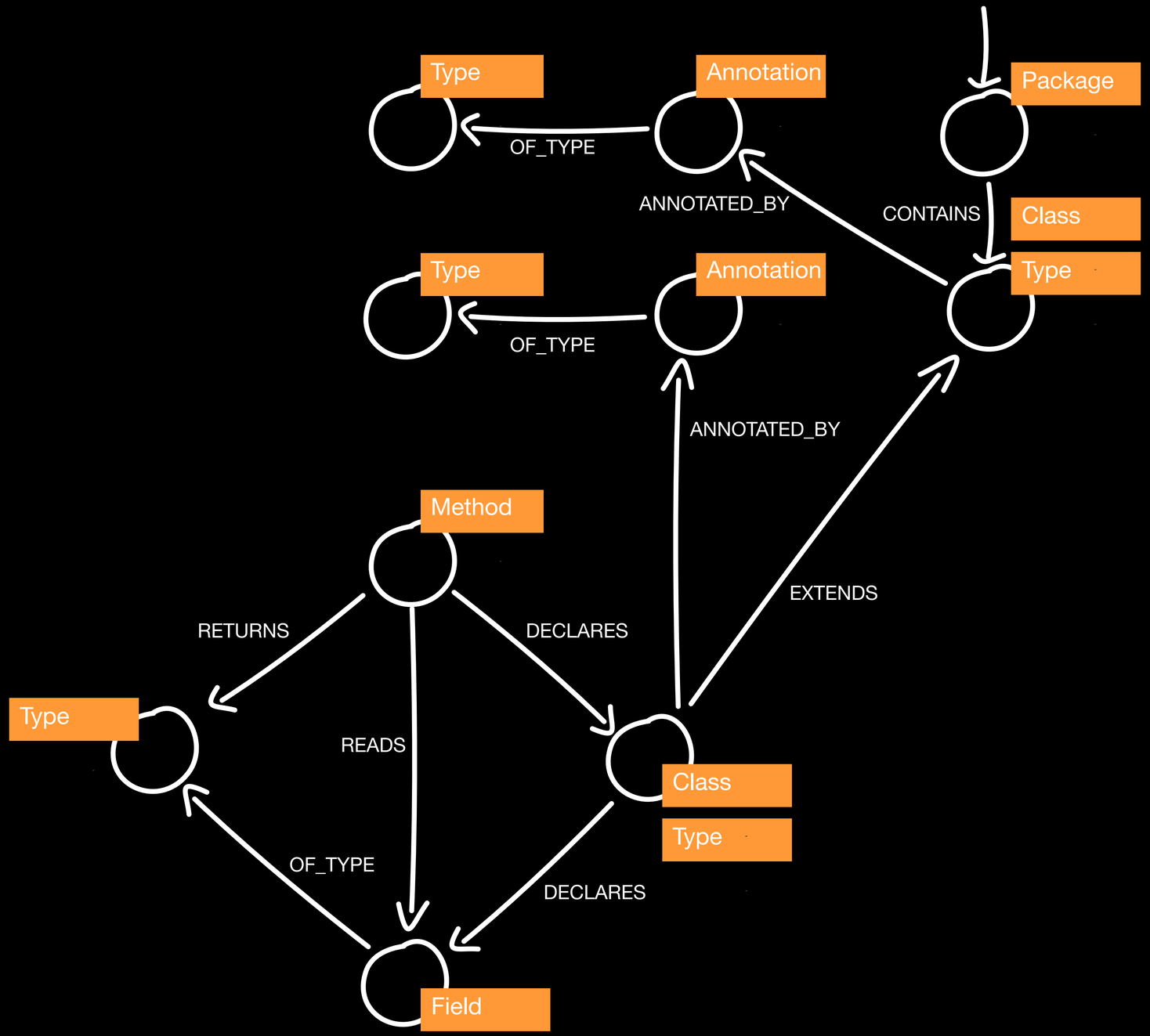
```java
@MappedSuperclass
public class Person extends BaseEntity {
}
```

```java
@Entity
@Table(name = "owners")
public class Owner extends Person {
    @Column(name = "address")
    @NotEmpty
    private String address;
}
```

An JPA Entity

```java
@MappedSuperclass
public class Person extends BaseEntity {
}
```

```java
@Entity
@Table(name = "owners")
public class Owner extends Persons {
    @Column(name = "address")
    @NotEmpty
    private String address;
}
```

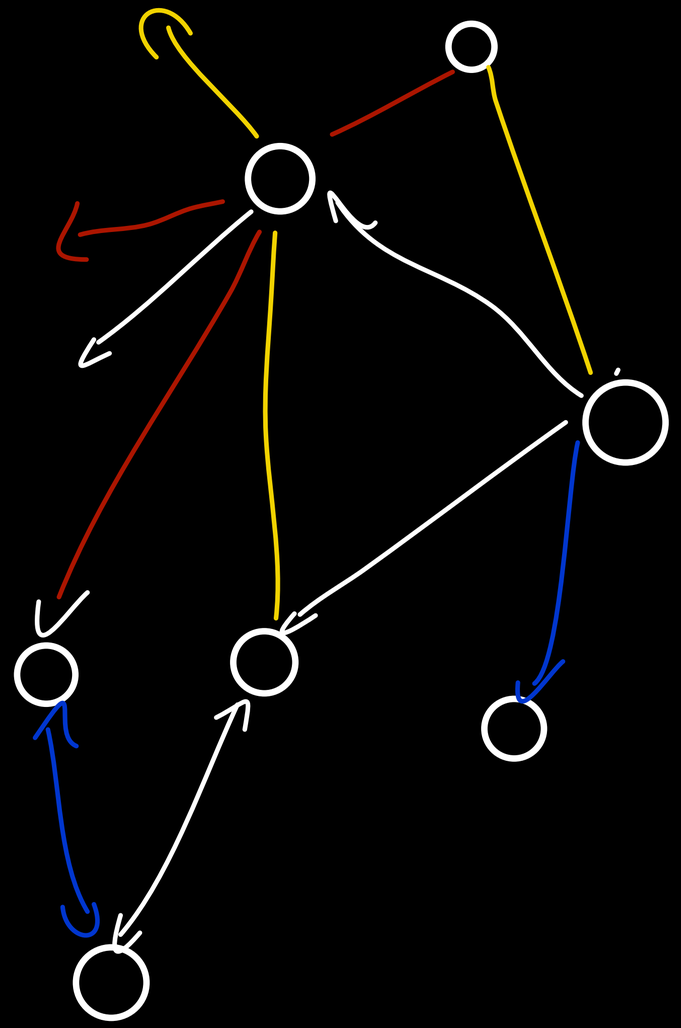


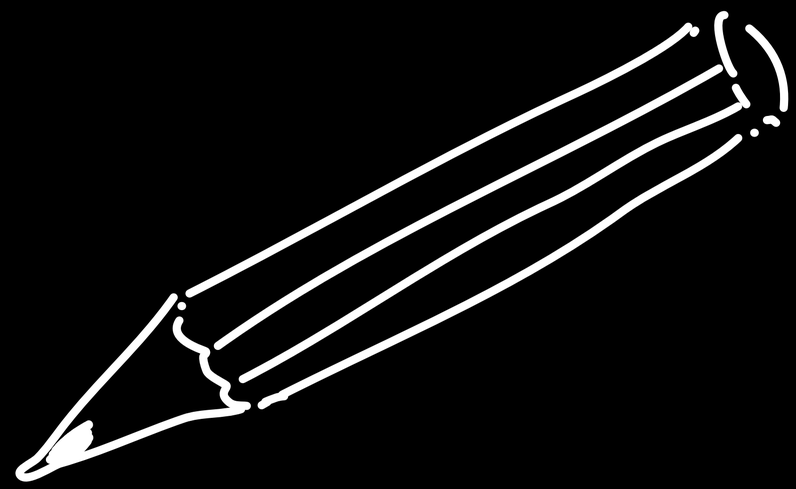
sweblogtweets @ o.b.fischer@swe-blog.net

All we need is..

1. Nodes
2. Labels
3. Properties
4. Relationships

Modelling is just

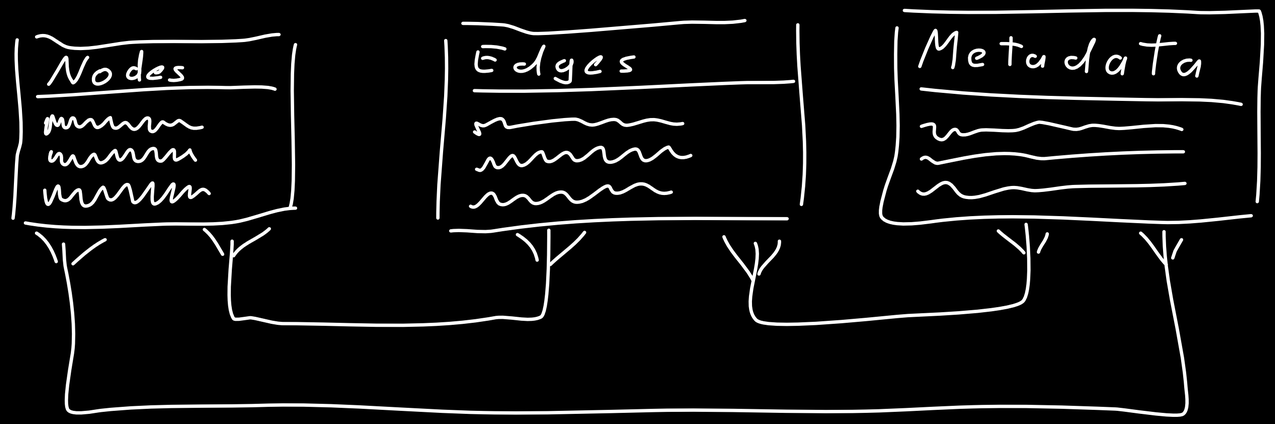




1. Reading the specification
2. Taking a pen
3. Drawing the structures on a whiteboard

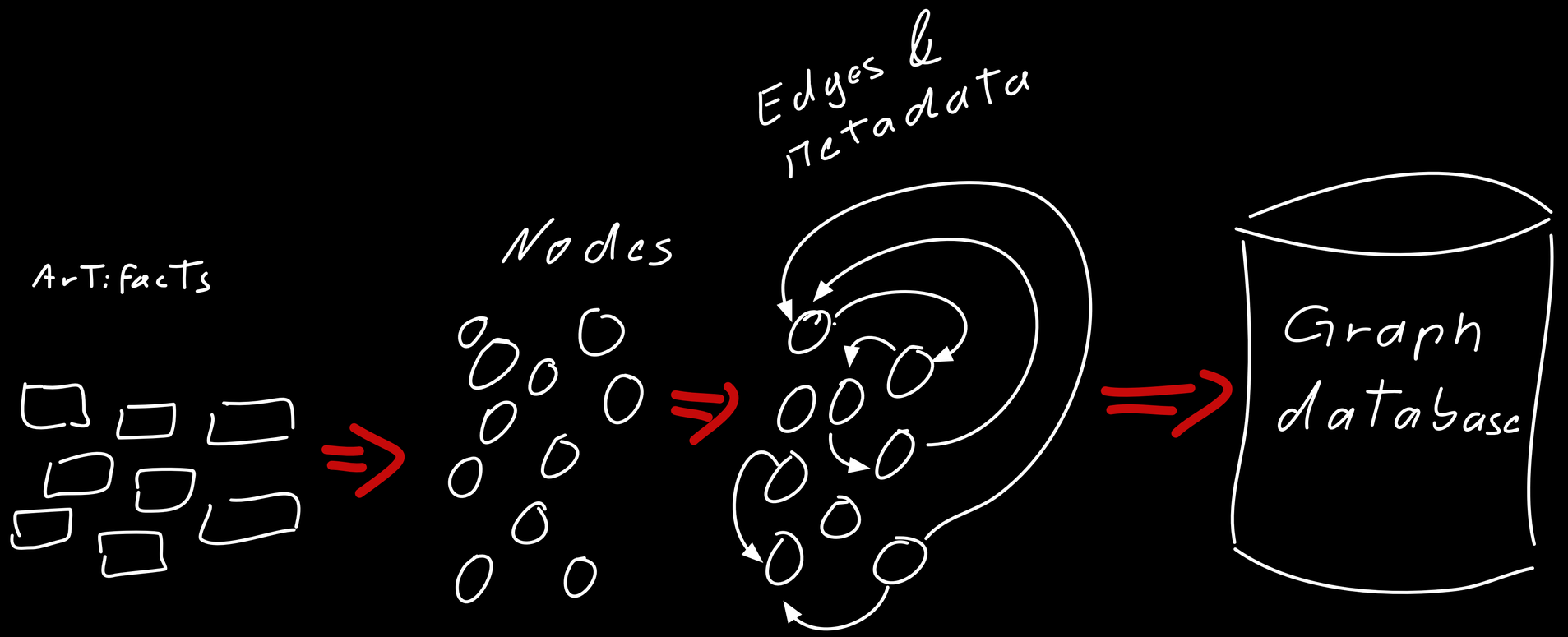
What we don’t need is…

1. Foreign keys
2. Predefined tables and schemas
3. Deep knowledge in graph theory

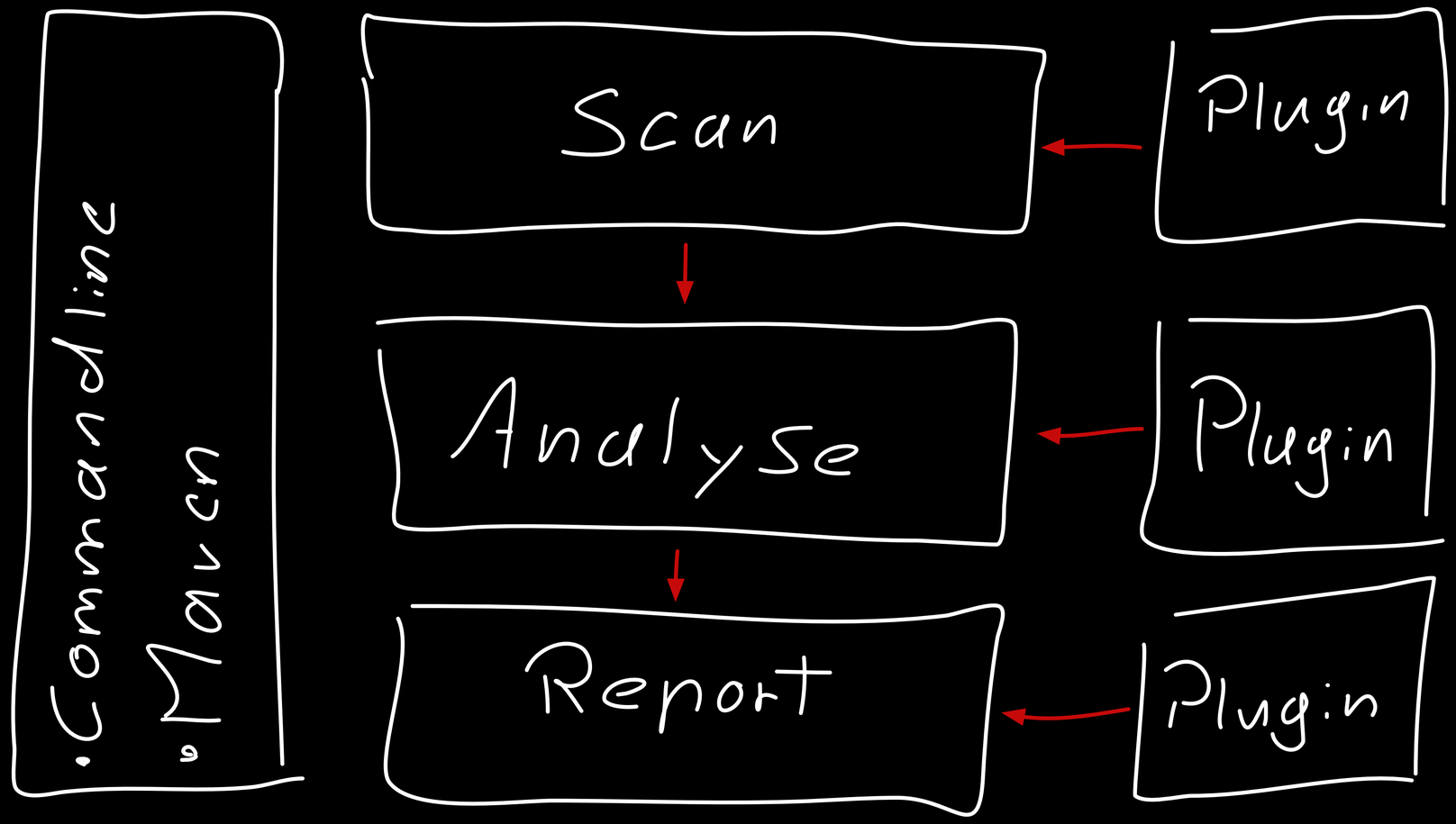


sweblogtweets @ o.b.fischer@swe-blog.net

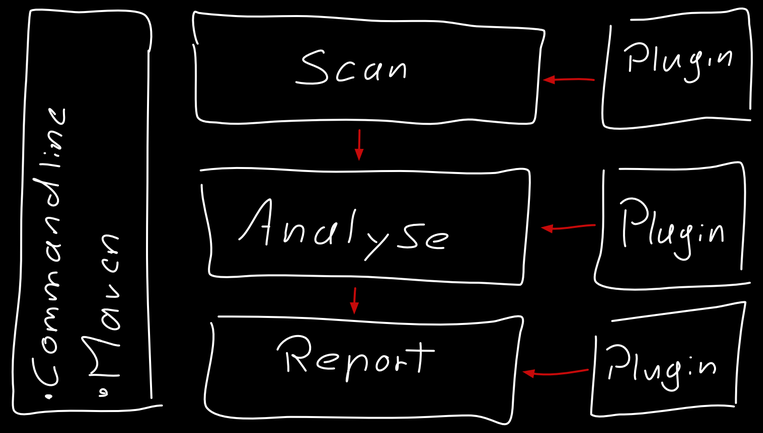
The Idea implemented



The Architecture



Available Scanner Plugins



.class
.json
.xml
.xml
beans.xml
web.xml
application.xml
pom.xml
.[jar|ear|war]
/META-INF/services/*
RDBMS Schema
M2 Repository
…

Public API available

.git
SpotBugs
JaCoCo
…

tl;dr - Cypher

Cypher is SQL for graphs

- Query language for Graph databases
- Developed by Neo4j for their graph database
- Now specified by the openCypher project
- Support for all CRUD operations

Used by:

- Neo4J
- SAP HANA Graph
- Upcoming Apache Spark version 3.0

tl;dr - Cypher

`MATCH`	Description of the pattern to match
`WHERE`	Optional clause for additional constraints on the pattern to match
`SET`	Changing properties, Labels and
`RETURN`	Selection of the information to return

tl;dr - Cypher

`()`	A node in a graph
`--`	An edge of a graph
`<--`	An directed edge of a graph
`(a)`	A variable for a node
`(:LABEL)`	A node with a label
`-[:RELATIONSHIPTYPE]->`	A directed edge with a label

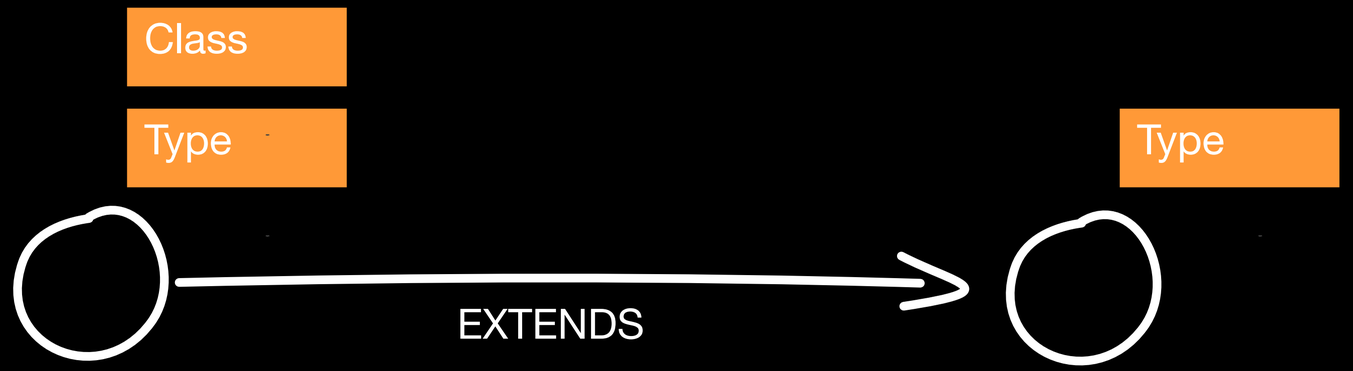
Cypher Example

```
public class Clazz extends SuperClazz {
}
```

Cypher Example


```
() -- ()
() --> ()
(c1) --> (c2)
(c1) -[:EXTENDS:]-> (c2)
(c1:Class) -[:EXTENDS]-> (c2:Type)
```

Cypher Example




```
MATCH   (c1:Class)-[:EXTENDS]->(c2:Type)
RETURN  c1.fqn, c2.fqn
```

Browser Settings

User Interface

Theme

- (•) Normal
- ( ) Outline
- ( ) Dark

- [x] Enhanced query editor
- [ ] Enable multi statement query editor

Preferences

- [ ] Show sample scripts

Initial command to execute

:play start

Result Frames

Maximum number of result frames

30

Max History

30

- [x] Scroll To Top

Graph Visualization

Initial Node Display

300

Max Neighbours

100

Max Rows

1000

- [x] Connect result nodes

$

To enjoy the full Neo4j Browser experience, we advise you to use Neo4j Browser Sync

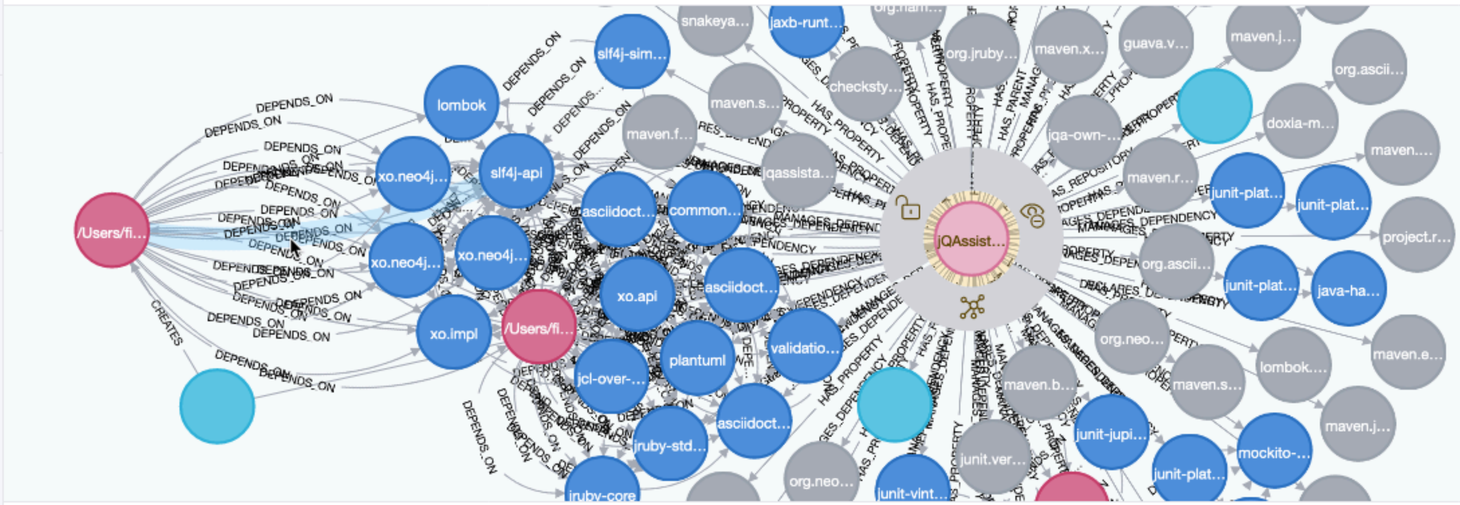
X

$ match (pom:Pom) where pom.name = 'jQAssistant Core Store' return pom

Graph | Table | Text | Code

*(241) Effective(1) Maven(45) Pom(1) Artifact(41) File(42) Main(3) jQAssistant(4) Directory(3) Project(1) Repository(2) Java(2) Property(48) Value(48)

*(287) HAS_PARENT(1) HAS_EFFECTIVE_MODEL(1) MANAGES_DEPENDENCY(38) HAS_REPOSITORY(2) DECLARES_DEPENDENCY(9) DESCRIBES(1) HAS_PROPERTY(48) CREATES(1) DEPENDS_ON(186)



DEPENDS_ON <id>: 132778 optional: false scope: compile

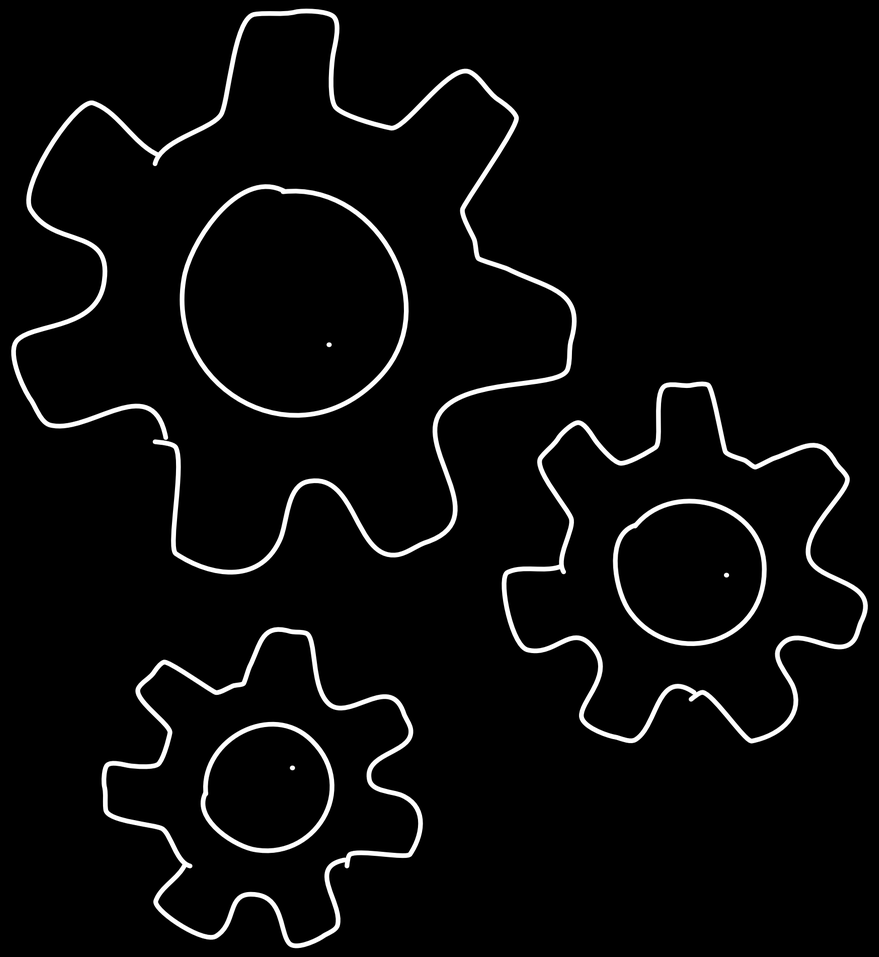
$ match (x:Xml) return count(x) as nodes

Table | Text | Code

nodes
11699

Started streaming 1 records after 1 ms and completed after 1 ms.

Automation with jQAssistant



- Rules can be read from the local filesystem
- Rules can be shared via a plugin mechanism
- Rules can have a description
- Rules can have a severity

Rules

Rule

Rule is superordinate concept

Constraint

Detects a violation (missing or illegal pattern in the graph model)

A package must not have more then 20 public interfaces.

Concept

Enrichs the graph data model with own information (meta data, relations, labels, …)

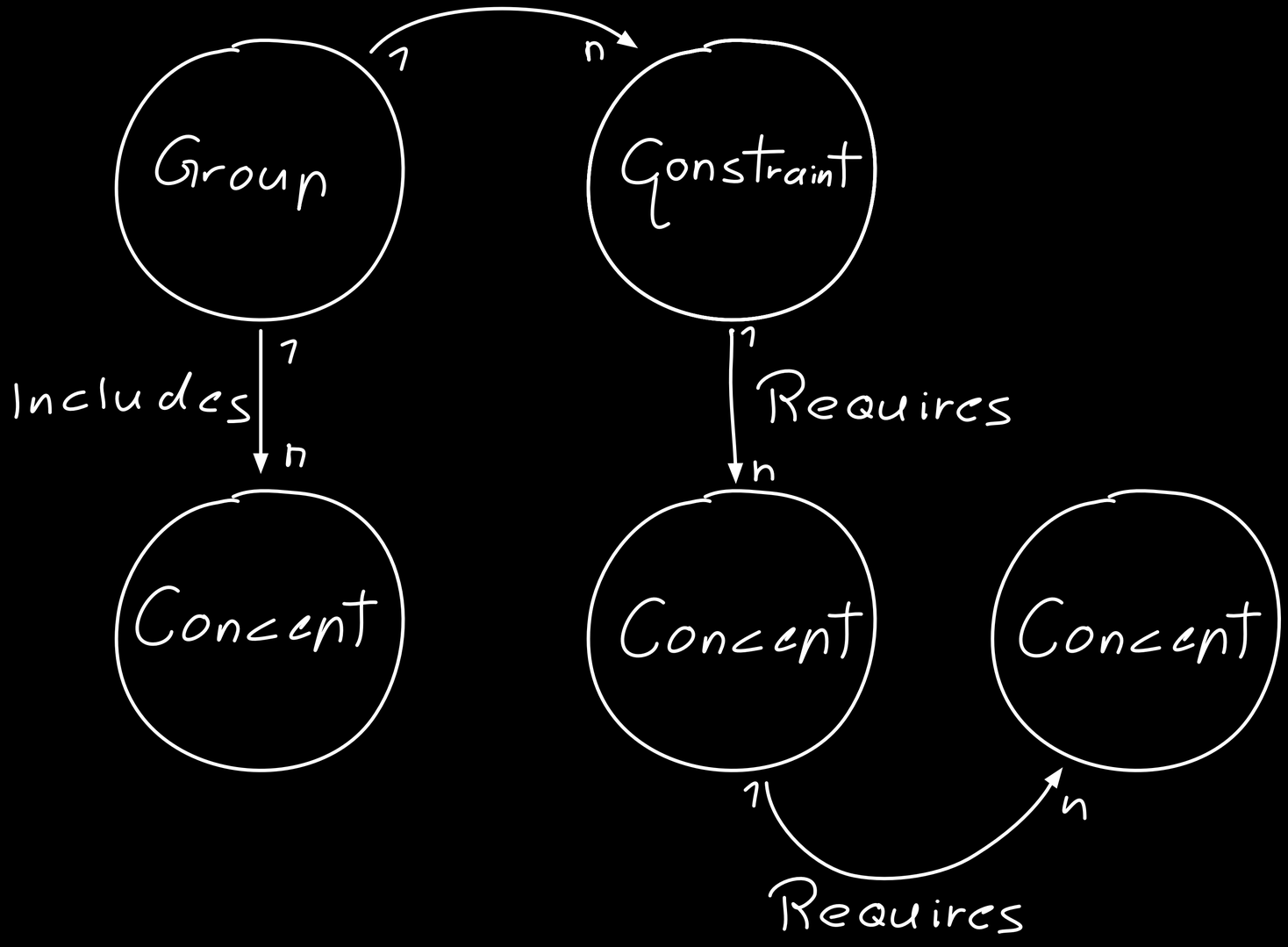
A class in the package model and annotated with @Entity is an JPA Entity

Group

Allows grouping of constraint as different execution profiles

All constraints related to software metrics.

Relationships between Rules



Writing Rules

Upcoming version
1.7.0

Yet Another Markup Language

eXtensible Markup Language

Asciidoctor

sweblogtweets @ o.b.fischer@swe-blog.net

Example № 1

Every Jar must contain a file called License

```
MATCH    (license:File)
WHERE    license.fileName =~ '/LICENSE'
RETURN   *
```

Example № 1

Every Jar must contain a file called License

```
- id: jqa-legal:license:exists
  severity: blocker
  description: |
    A file called ,LICENSE' must exist in the
    root directory of the Jar.
  cypher: |-
    MATCH      (license:File)
    WHERE      license.fileName =~ '/LICENSE'
    RETURN      *
```

Example № 2

The Maven version property used for com:buschmais.jqassistant.plugin:common must be called plugin-common.version

```
MATCH  (project:Maven:Project)-[:HAS_MODEL]->
       (pom:Pom)-[:DECLARES_DEPENDENCY]->
       (dependency {
           group: 'com.buschmais.jqassistant.plugin',
           name:  'common'
        })

WHERE  dependency.version <> '${plugin-common.version}'

RETURN dependency
```

Example № 2

The Maven version property used for com:buschmais.jqassistant.plugin:common must be called plugin-common.version

```
<constraint id="jqa-dependency-constraints:common-declaration"
            severity="blocker">
    <description><![CDATA[
A good description of the constraint to fulfill
]]></description>

    <cypher><![CDATA[
MATCH   (project:Maven:Project)-[:HAS_MODEL]->
        (pom:Pom)-[:DECLARES_DEPENDENCY|:MANAGES_DEPENDENCY]
        ->(dependency { group: 'com.buschmais.jqassistant.plugin',
                        name:  ,common' })
WHERE    dependency.version <> '${plugin-common.version}'
RETURN   dependency
]]></cypher>
</constraint>
```

Example № 3

All JPA entity classes must be located in the package model

A Concept to describe a JPA entity

```
MATCH    (t:Type)-[:ANNOTATED_BY]->()
          -[:OF_TYPE]->(a:Type)
WHERE    a.fqn="javax.persistence.Entity"
SET      t:Jpa:Entity
RETURN   t AS JpaEntity
```

A Constraint for the actual Rule

```
MATCH    (pkg:Package)-[:CONTAINS]->(entity:Jpa:Entity)
WHERE    pkg.name <> "model"
RETURN   entity AS EntityInWrongPackage
```

Example № 3

All JPA entity classes must be located in the package model

A Concept to describe a JPA entity

```
<concept id="jpa2:Entity">
    <description>Labels all types annotated with
@javax.persistence.Entity with "Jpa" and "Entity".</
description>
    <cypher><![CDATA[
        MATCH
          (t:Type)-[:ANNOTATED_BY]->()-[:OF_TYPE]-
>(a:Type)
        WHERE
          a.fqn="javax.persistence.Entity"
        SET
          t:Jpa:Entity
        RETURN
          t AS JpaEntity
    ]]></cypher>
</concept>
```

Example № 3

All JPA entity classes must be located in the package model

A Constraint for the actual Rule

```
[[model:JpaEntityInModelPackage]]
.All JPA entities must be located in packages named
"model".
[source,cypher,role=constraint,
 requiresConcepts="jpa2:Entity"]
----
MATCH
  (package:Package)-[:CONTAINS]->(entity:Jpa:Entity)
WHERE
  package.name <> "model"
RETURN
  entity AS EntityInWrongPackage
----
```

Example № 4

Compute the efferent coupling of a package

```
MATCH            (p:Package)
OPTIONAL MATCH   (p)-[:CONTAINS]->(it:Java:Type)
                 -[:DEPENDS_ON]->(et:Java:Type)
                 <-[:CONTAINS]-(ep:Package)
WHERE            p <> ep
WITH             p, COUNT(et) AS efferentCouplings
SET              p.ce = COALESCE(efferentCouplings, 0)
RETURN           p
```

Efferent Coupling

The number of data types a class knows. Usefull to rate the instability of a component.

sweblogtweets @ o.b.fischer@swe-blog.net