Analysis of software systems using jQAssistant and Neo4j

E-POST Tech Talk &
JUG Berlin-Brandenburg

Dirk Mahler

AGENDA

- About Structures, Rules And Code
- Software As A Graph
- Demo#1
- jQAssistant
- Demo#2

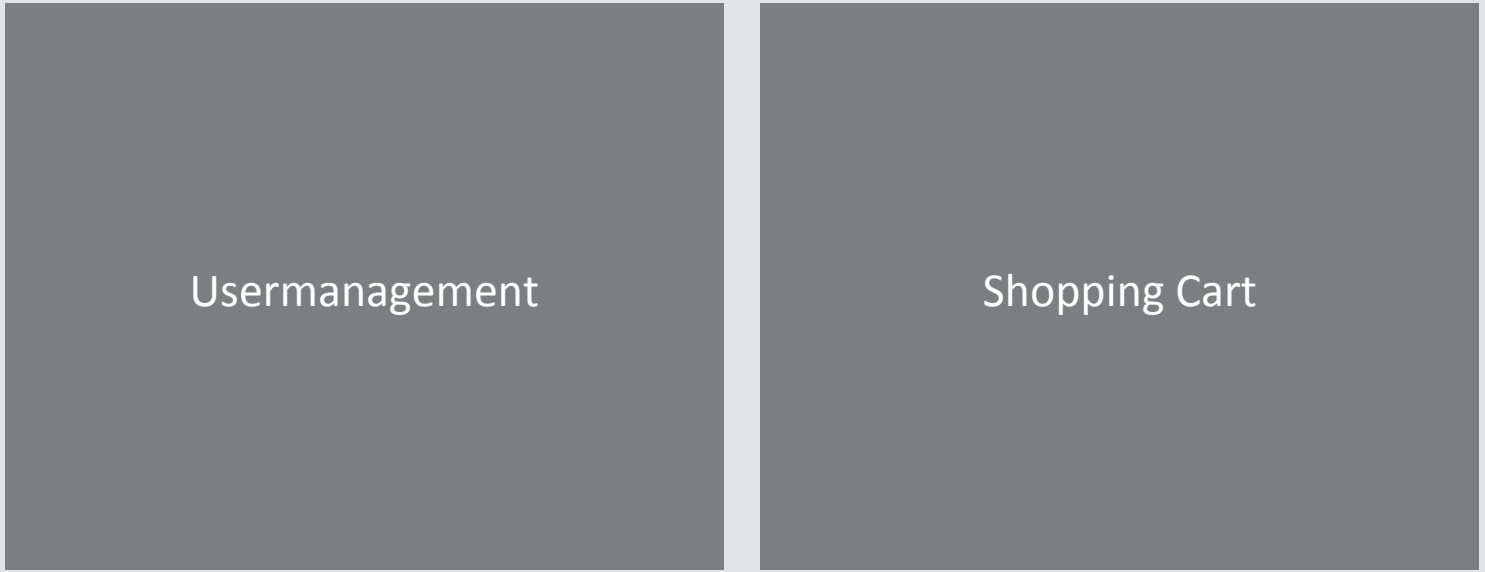
Analysis of software systems using jQAssistant and Neo4j

About Structures, Rules And Code

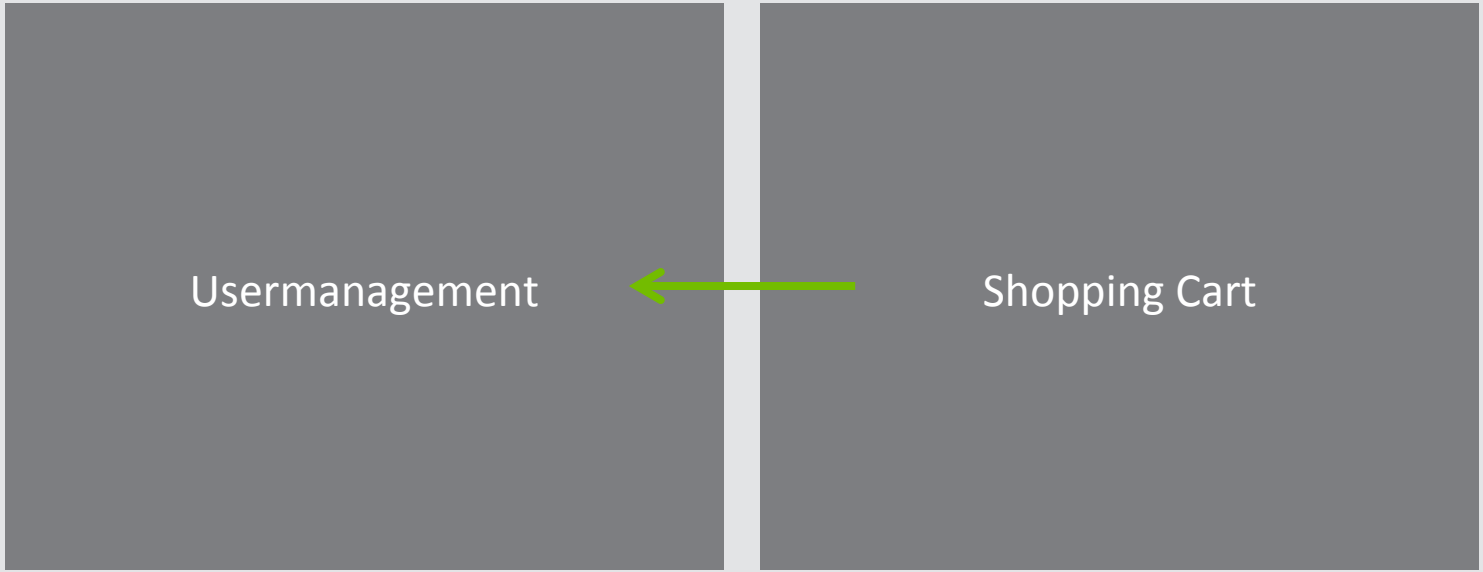
- Sketch of an architecture

- Sketch of an architecture

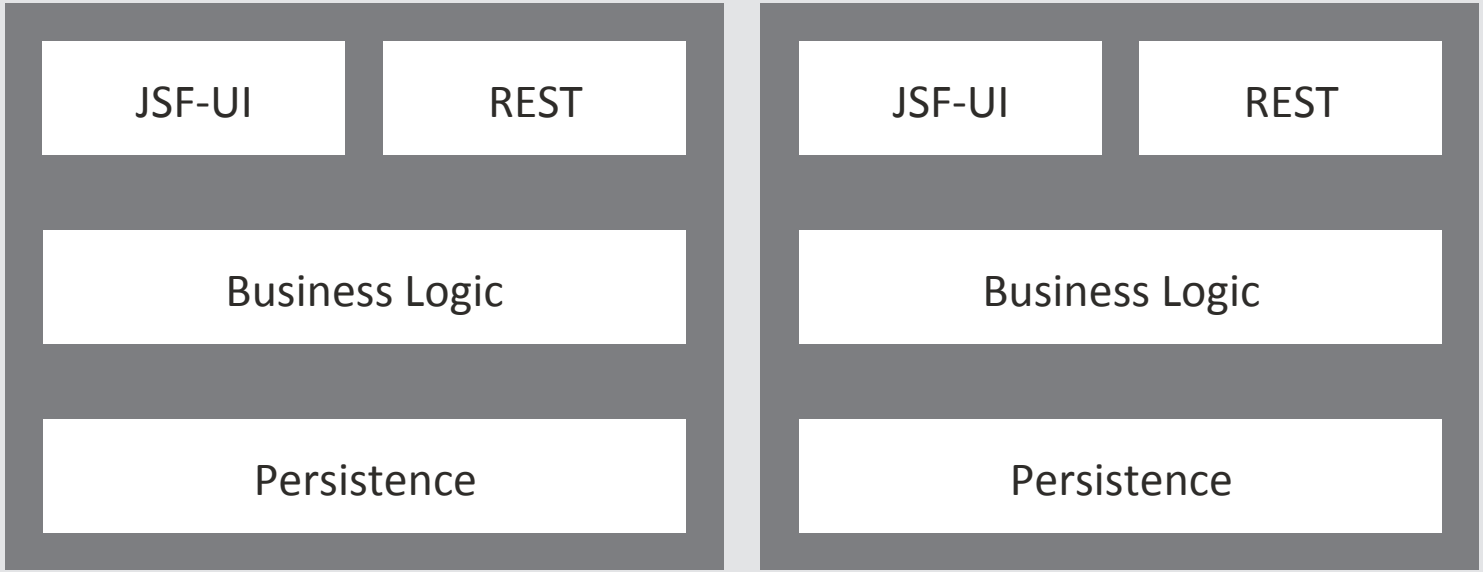
- Sketch of an architecture
  - Business modules



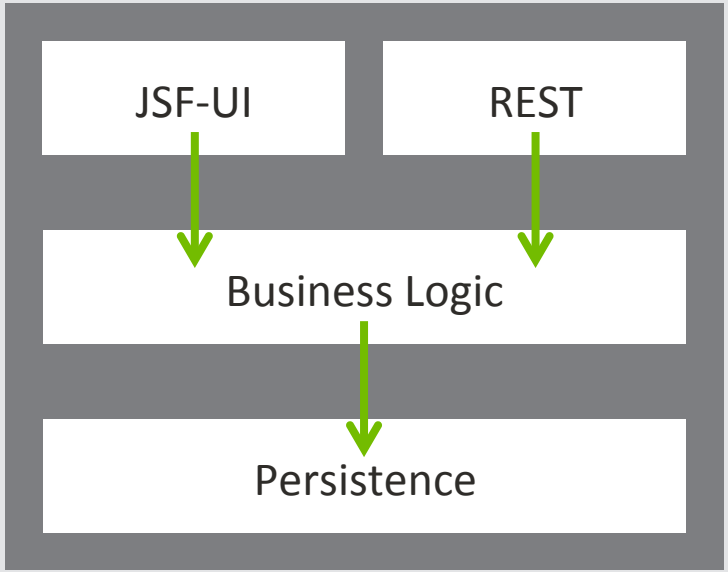
- Sketch of an architecture
  - Allowed dependencies between business modules

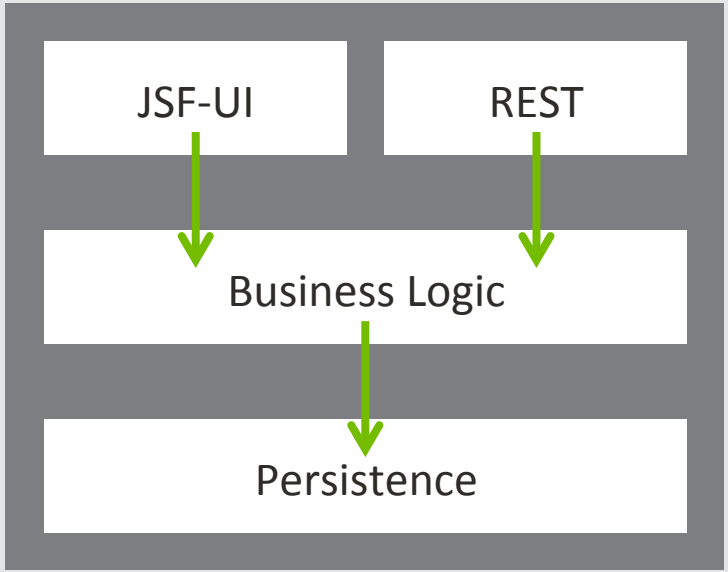


- Sketch of an architecture
  - Technical layering

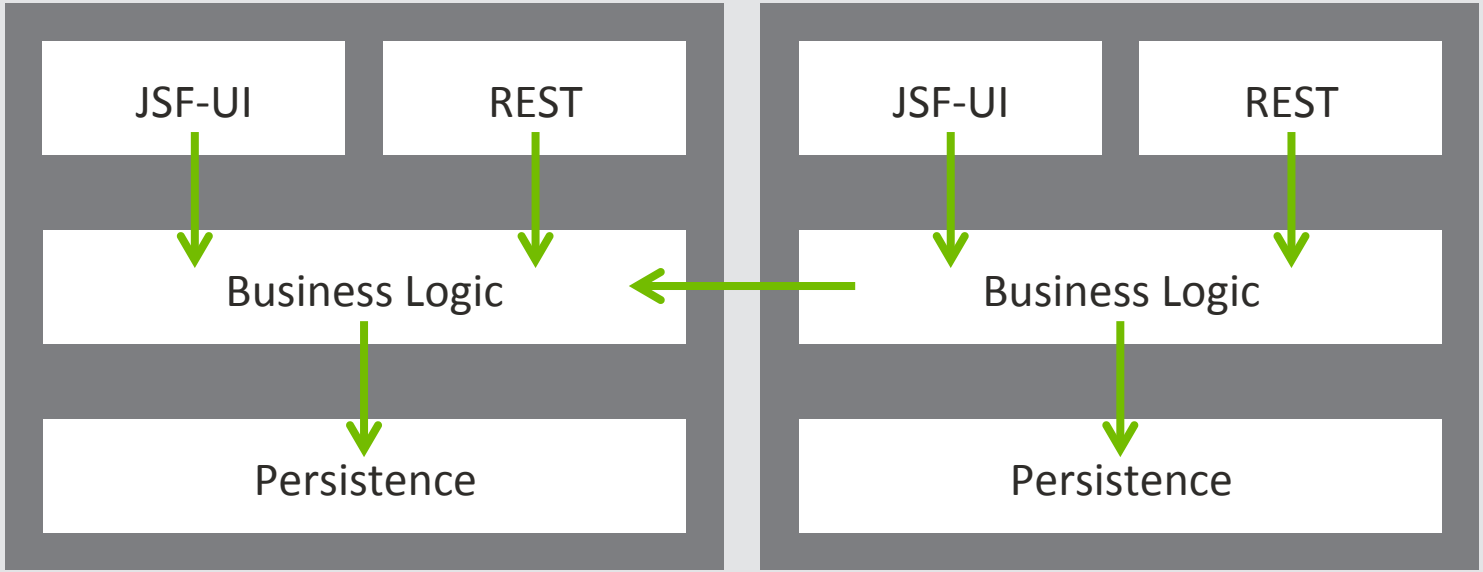


- Sketch of an architecture
  - Allowed dependencies between technical layers

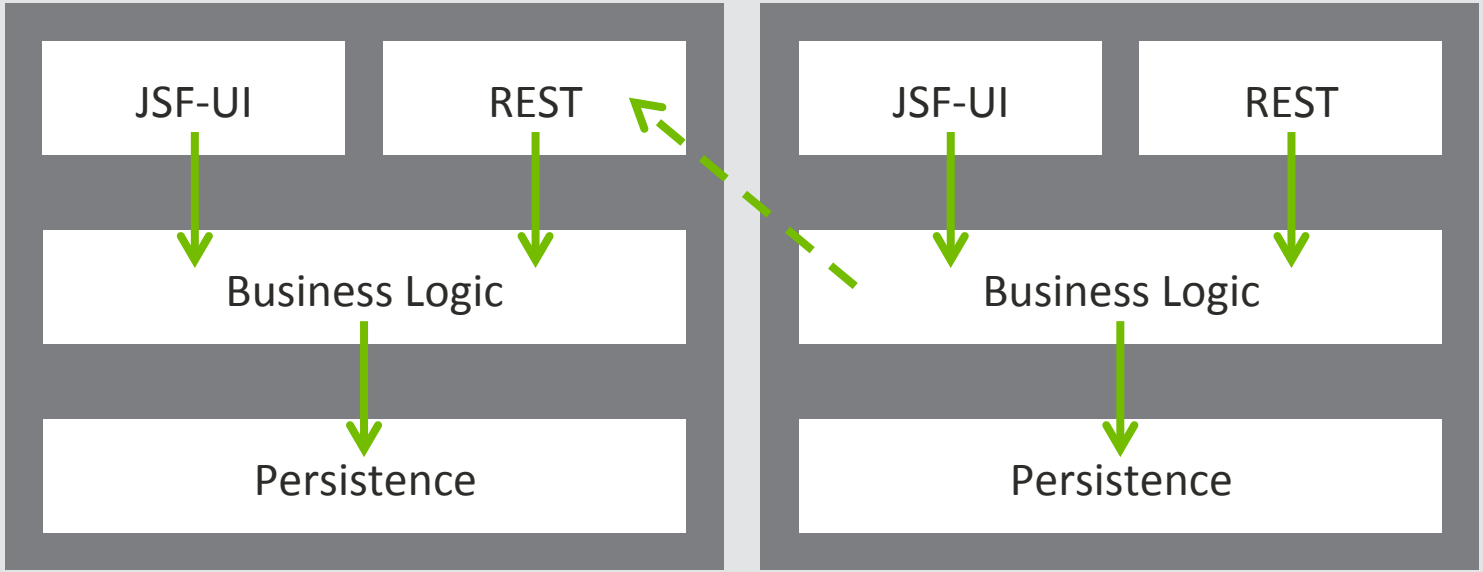




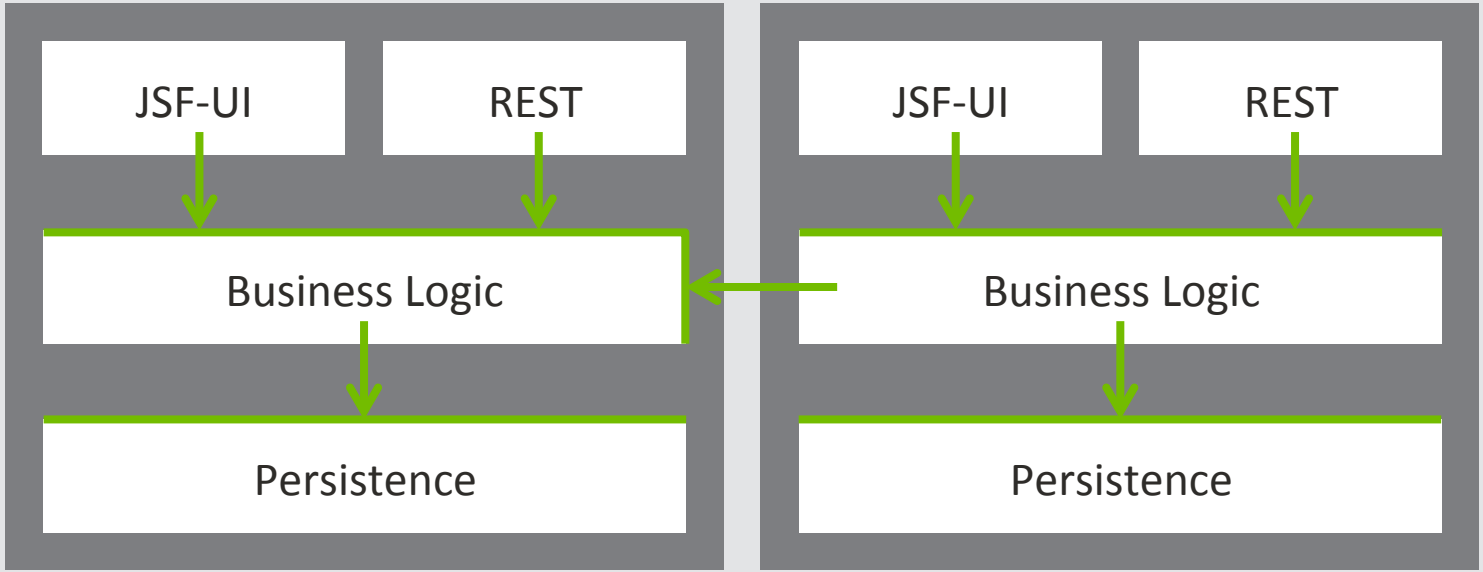
- Sketch of an architecture
  - Allowed dependencies between business modules and technical layers



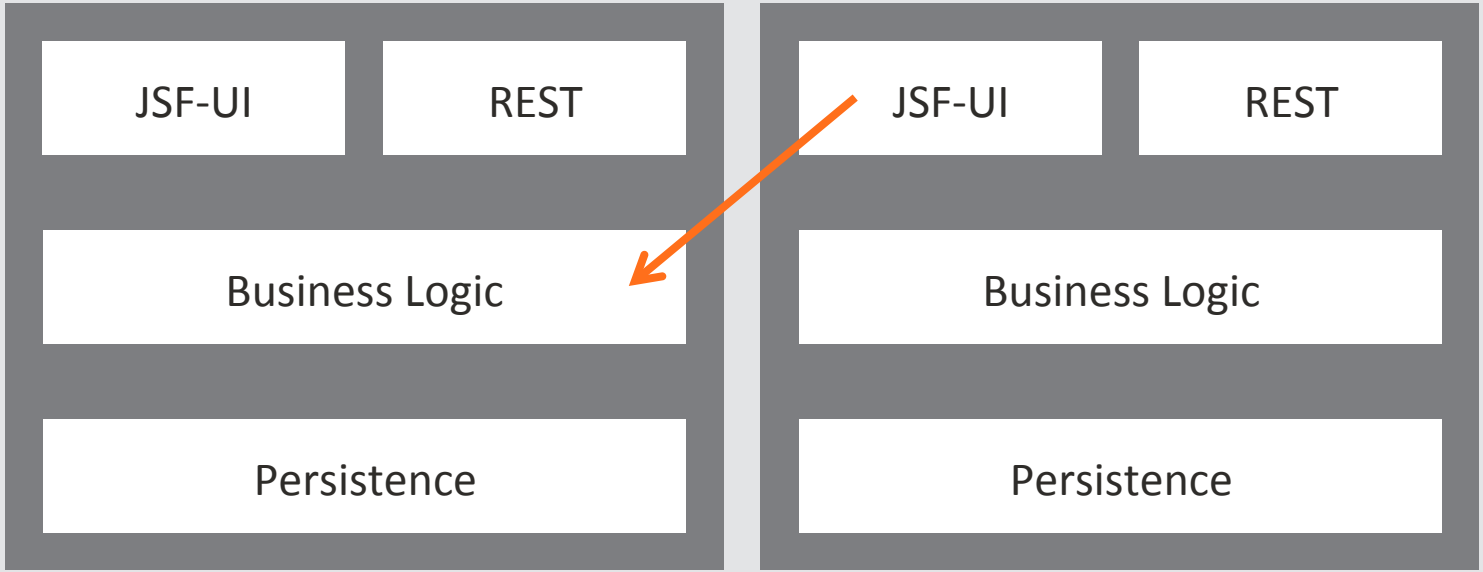
- Sketch of an architecture
  - Allowed dependencies between business modules and technical layers



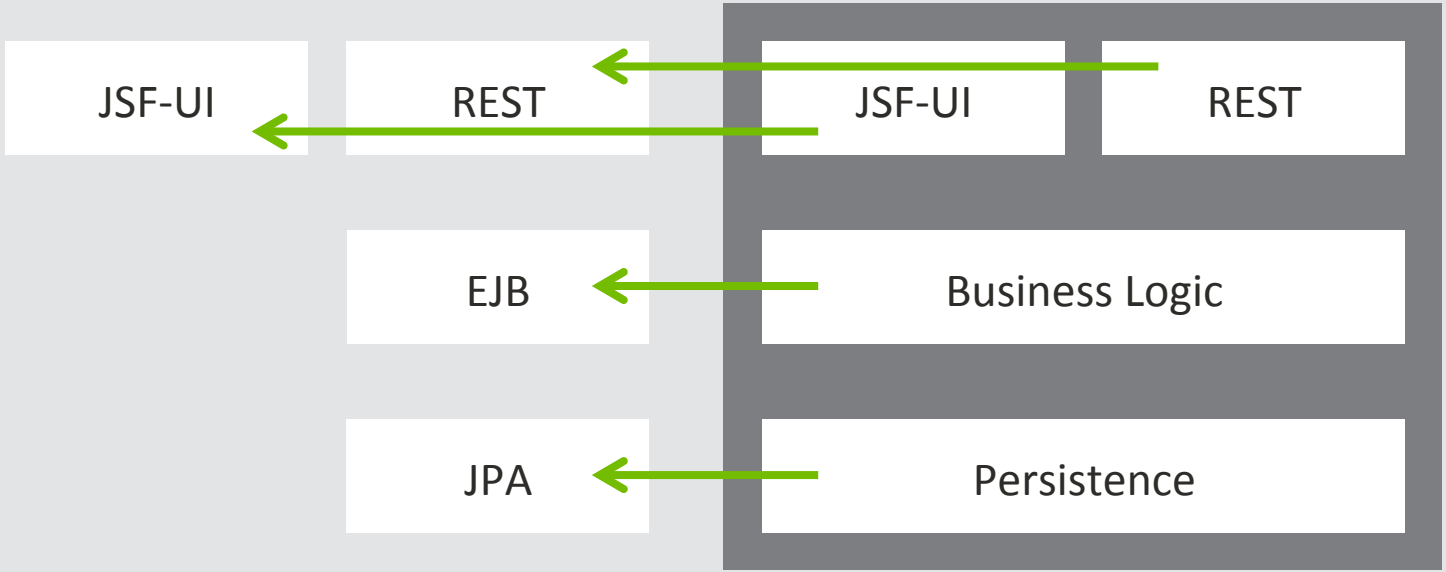
- Sketch of an architecture
  - Abstraction between layers (API vs. Implementation)



- Sketch of an architecture
  - Forbidden dependency



- Sketch of an architecture
  - Allowed external dependencies per layer

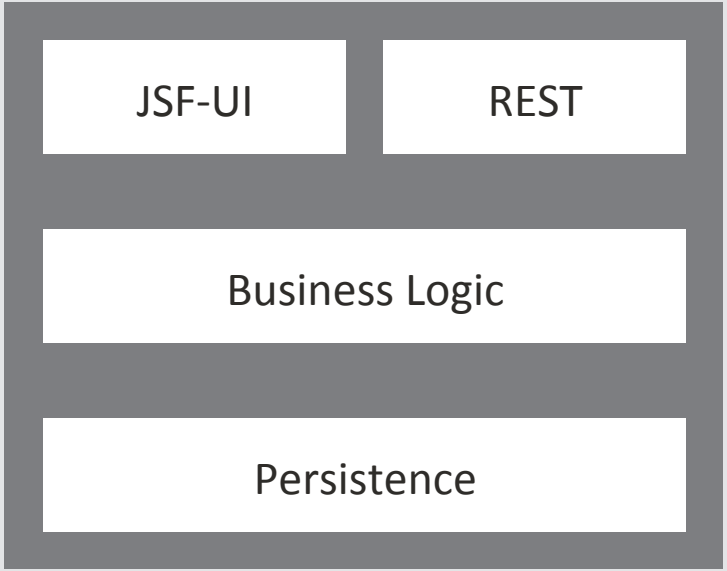


- Translation of architecture rules into the project structure
  - Java language element: Package

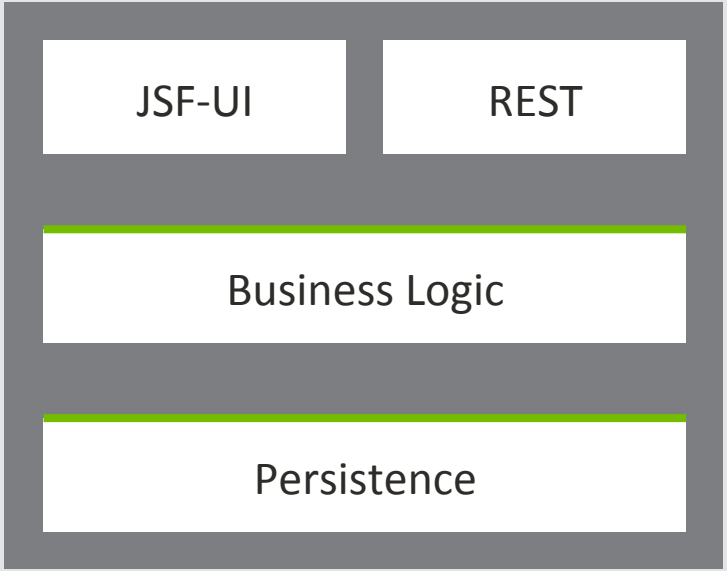
- Translation of architecture rules into the project structure
  - Java language element: Package

  - Definition of business modules on „top level“

- Translation of architecture rules into the project structure
  - Java language element: Package
  - Technical layers
    - …demo.cart.ui
    - …demo.cart.rest
    - …demo.cart.logic
    - …demo.cart.persistence



- Translation of architecture rules into the project structure
  - Java language element: Package
  - Technical layers
    - …demo.cart.ui
    - …demo.cart.rest
    - …demo.cart.logic.api
    - …demo.cart.logic.impl
    - …demo.cart.persistence.api
    - …demo.cart.persistence.impl



- Package Names
  - All packages in a Maven module must be prefixed with ${groupId}.${artifactId}
  - Example:
    groupId=org.jqassistant
    artifactId=demo

    => org.jqassistant.demo

- Class Names
  - Message Driven Beans must have a suffix „MDB“.

- Class location
  - JPA entities must be located in „api.model“ packages.

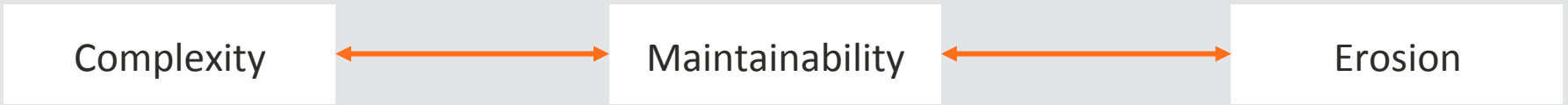
- Test design
  - Every test method must contain at least one assertion.
  - Each assertion must provide a human readable message.

- Abstraction
  - Remote APIs must be interfaces declaring only primitives or immutables as parameter or return types.
  - OSGi-Bundles must only export dedicated API packages.

Source Code

Java, Properties,
XML, YML

„Code Quality“



Source Code

Java, Properties,
XML, YML

People

Architect, Developer,
Test

Tooling

Compiler, Static Code
Analysis, CI

Documentation

UML, Wiki,
readme.txt

Source Code – Abstraction levels

Architecture	Module, Layer, Dependency
Design	Abstraction, Immutable, Factory
Java	Package, Class, Field, Method, Annotation
File System	Folder, File

Source Code – Automated validation

Architecture	Module, Layer, Dependency
Design	Abstraction, Immutable, Factory
Java	Package, Class, Field, Method, Annotation
File System	Folder, File

- Java elements represent higher level concepts
  - Package ⇔ Module
  - Annotated Class ⇔ Entity
- Constraints apply to these concepts
  - JPA entities must be located in „api.model“ packages

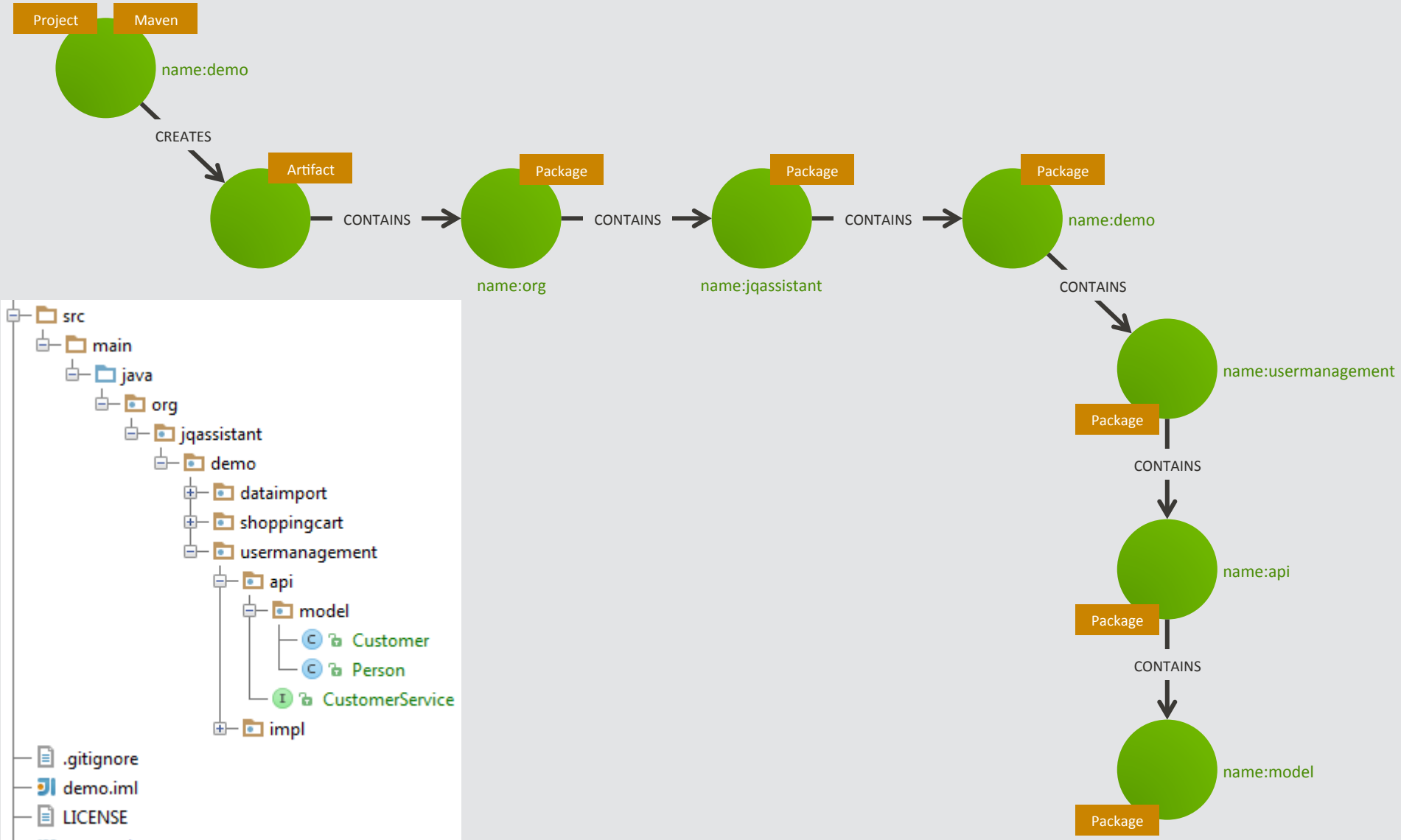
- Visibility of concepts and constraints
  - Explicit
    - @Test
  - Implicit
    - Every test method must contain at least one assertion.
  - Invisible
    - java.util.Calendar must not be used directly.

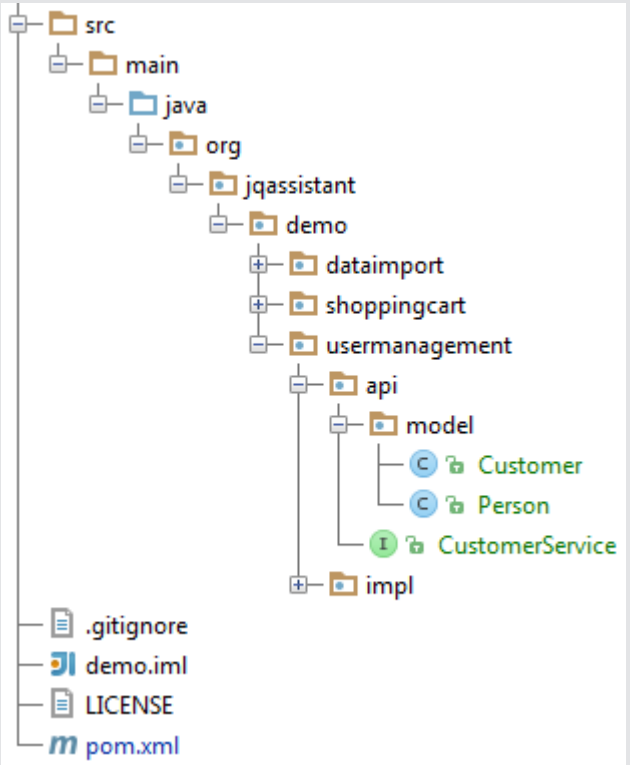
- Concepts and Constraints
  - How to make them visible?
  - Is there a language to describe them?

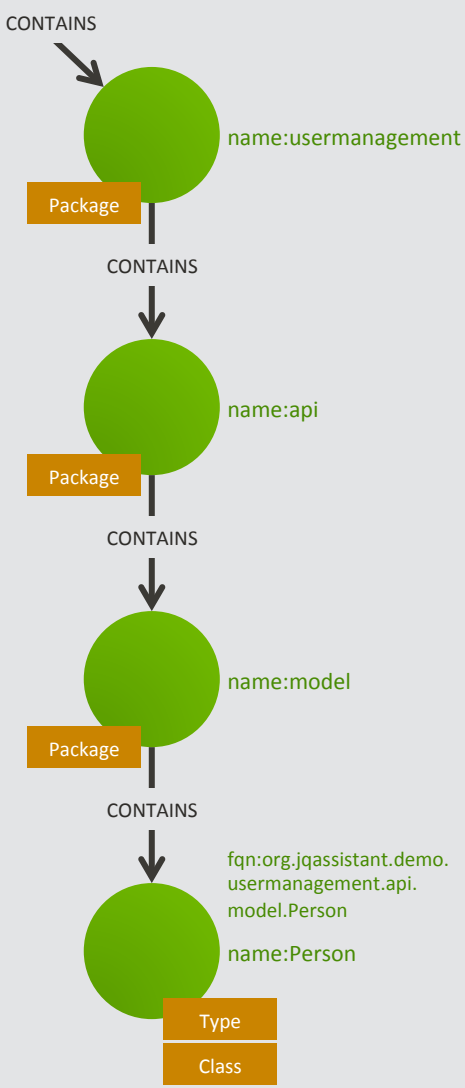
Analysis of software systems using jQAssistant and Neo4j

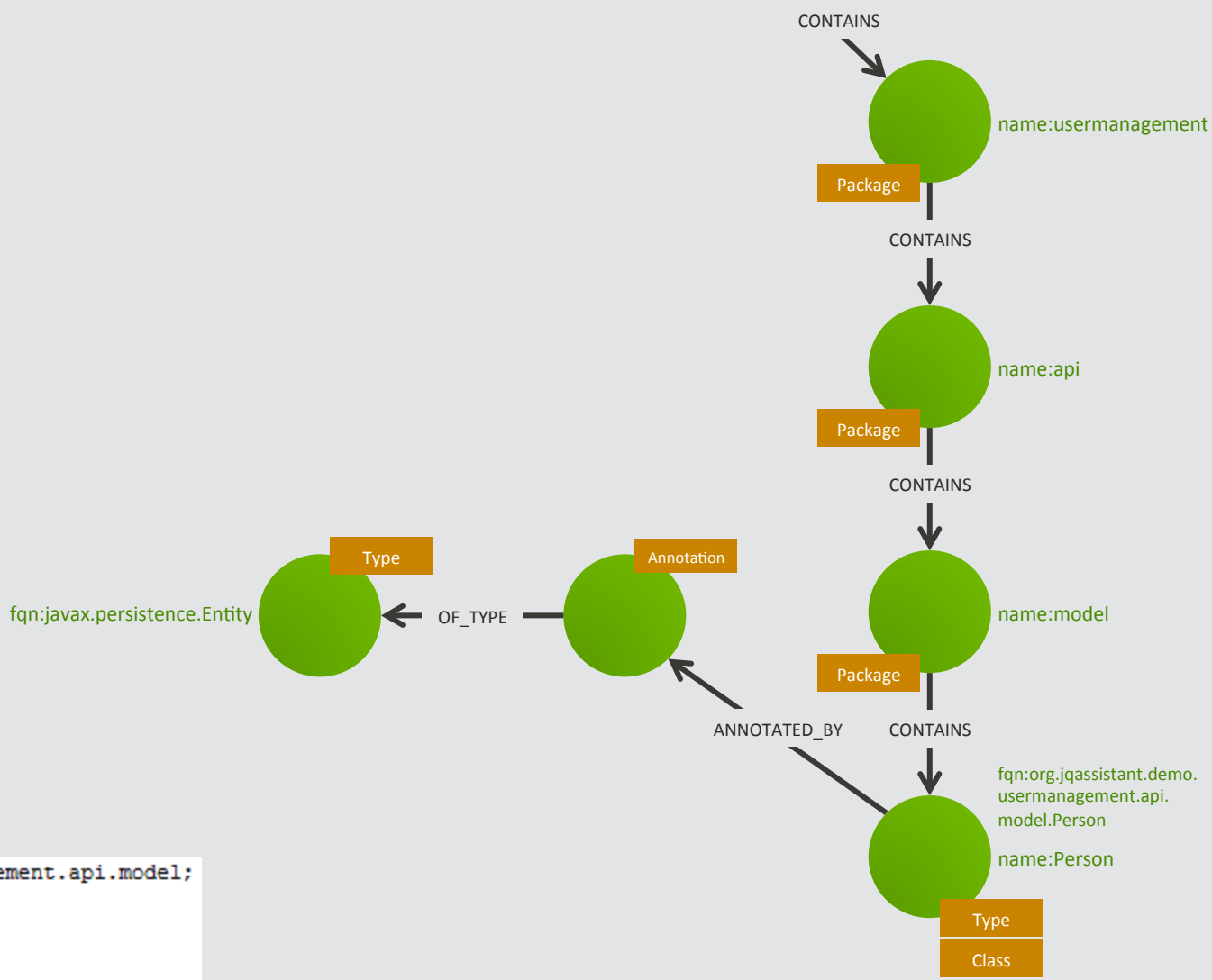
Software As A Graph

- All we need is…
  - Nodes
  - Labels
  - Properties
  - Relationships
- Modeling is just…
  - Taking a pen
  - Drawing the structures on a whiteboard (i.e. the database)
- We don‘t need…
  - Foreign keys
  - Tables and schemas
  - Knowledge in graph theory

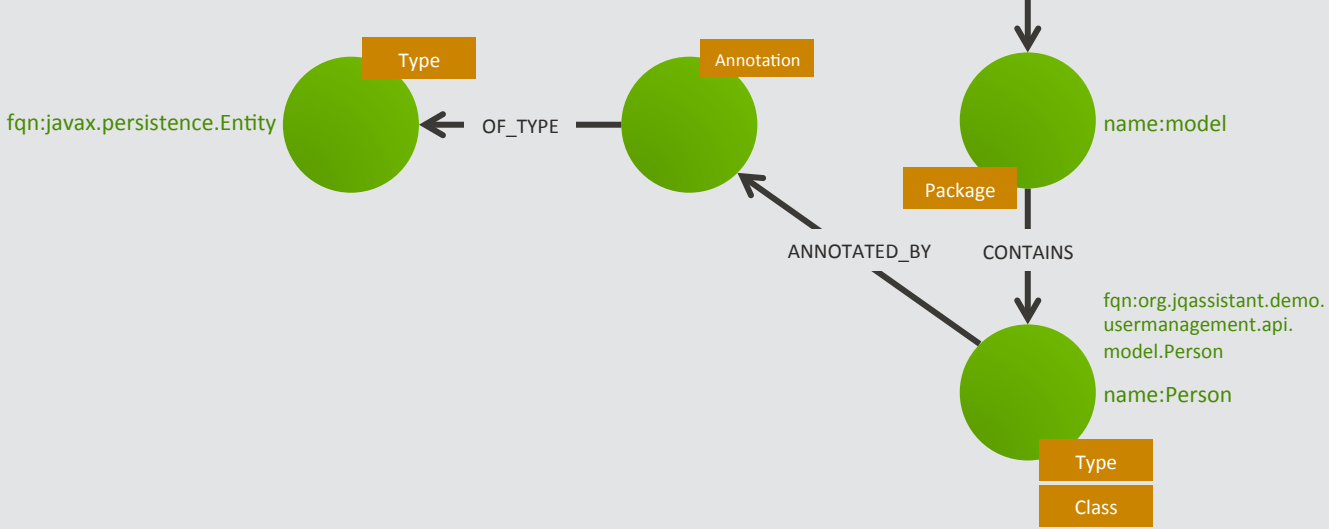

Project
Maven
name:demo
CREATES
Artifact
CONTAINS
Package
name:org
CONTAINS
Package
name:jqassistant
CONTAINS
Package
name:demo
CONTAINS
name:usermanagement
Package
CONTAINS
name:api
Package
CONTAINS
name:model
Package
src
main
java
org
jqassistant
demo
dataimport
shoppingcart
usermanagement
api
model
Customer
Person
CustomerService
impl
.gitignore
demo.iml
LICENSE
pom.xml


src
main
java
org
jqassistant
demo
dataimport
shoppingcart
usermanagement
api
model
Customer
Person
CustomerService
impl
.gitignore
demo.iml
LICENSE
pom.xml

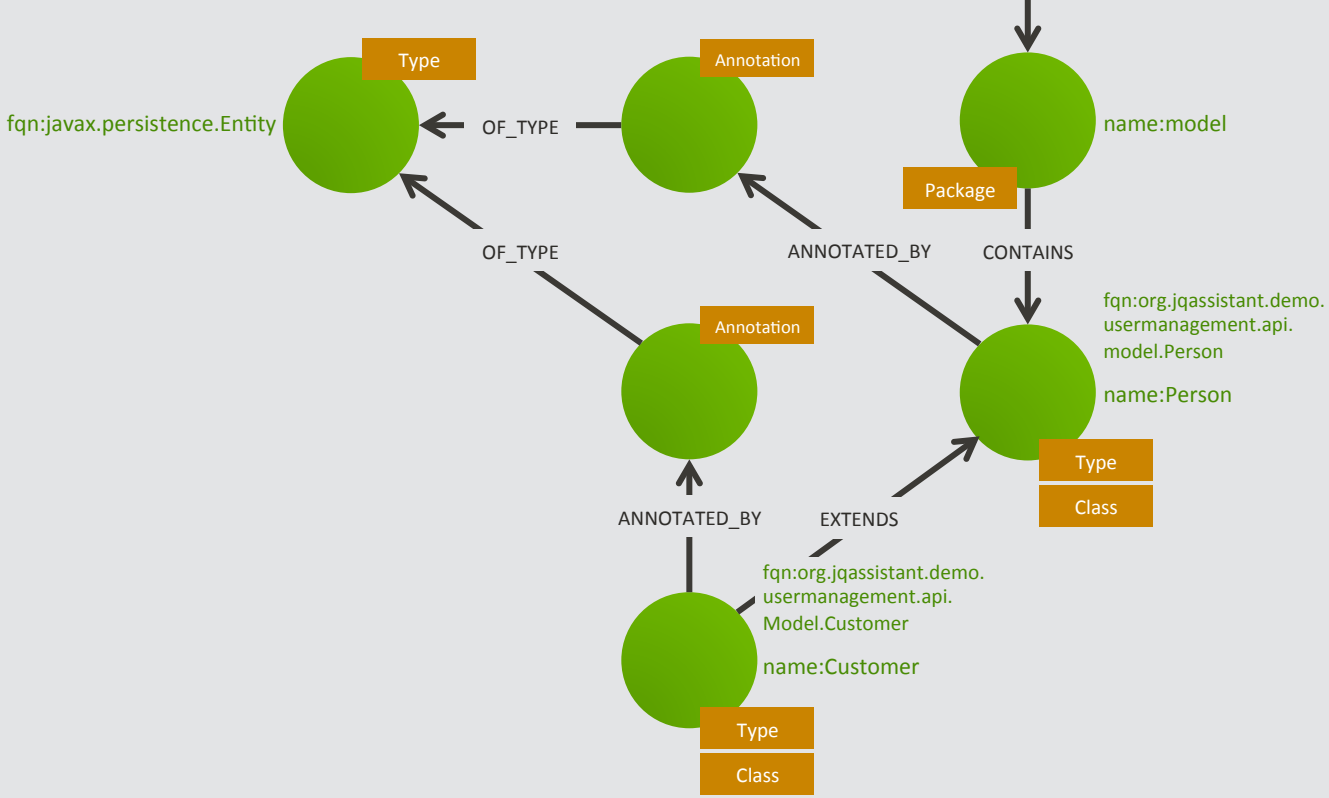

CONTAINS
name:usermanagement
Package
CONTAINS
name:api
Package
CONTAINS
name:model
Package
CONTAINS
fqn:org.jqassistant.demo.
usermanagement.api.
model.Person
name:Person
Type
Class



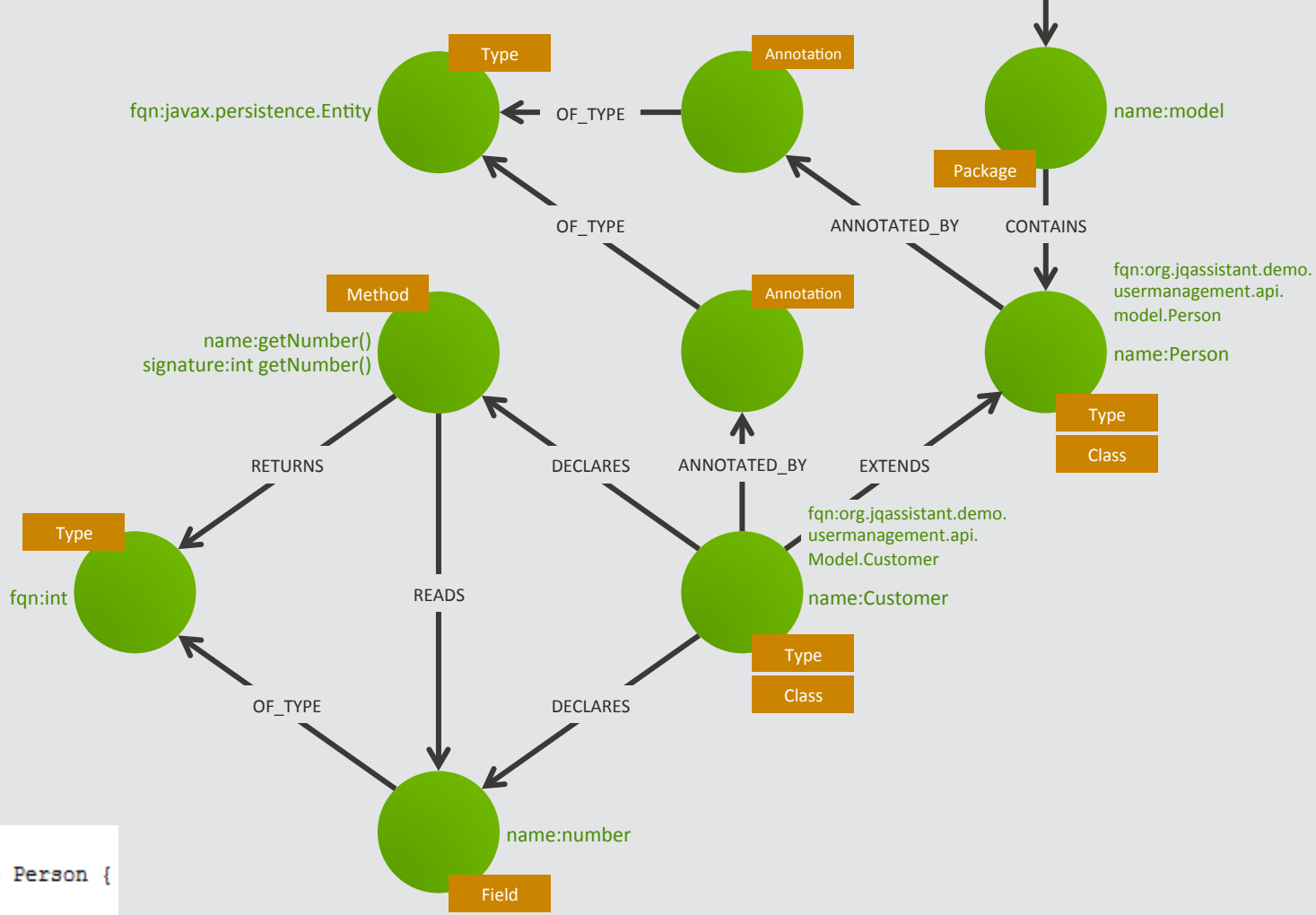

```java
package org.jqassistant.demo.usermanagement.api.model;

import javax.persistence.Entity;
import javax.persistence.Id;

@Entity
public class Person {
```




```java
package org.jqassistant.demo.usermanagement.api.model;

import javax.persistence.Entity;
import javax.persistence.Id;

@Entity
public class Person {
```




```java
@Entity
public class Customer extends Person {

    private int number;

    public int getNumber() {
        return number;
    }
```




```java
@Entity
public class Customer extends Person {

    private int number;

    public int getNumber() {
        return number;
    }
```

- Explore an application using queries
  - Which class extends from another class?

- Let‘s convert this to ASCII art…
  - () as nodes
  - -[]-> as directed relationships

- Explore an application using queries
  - Which class extends from another class?

- Let‘s convert this to ASCII art…
  - () as nodes
  - -[]-> as directed relationships

```
()-[]->()
```

- Explore an application using queries
  - Which class extends from another class?

- Let‘s convert this to ASCII art…
  - () as nodes
  - -[]-> as directed relationships

```
(c1)-[]->(c2)
```

- Explore an application using queries
  - Which class extends from another class?

- Let‘s convert this to ASCII art…
  - () as nodes
  - -[]-> as directed relationships

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

- Explore an application using queries
  - Which class extends from another class?

- Let‘s convert this to ASCII art…
  - () as nodes
  - -[]-> as directed relationships

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

- Explore an application using queries
  - Which class extends from another class?

- Pattern matching is the core principle of Cypher

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

- Which classes contain the highest number of methods?

- Which classes contain the highest number of methods?

```
MATCH
  (class:Class)-[:DECLARES]->(method:Method)
```

- Which classes contain the highest number of methods?

```
MATCH
  (class:Class)-[:DECLARES]->(method:Method)
RETURN
  class.fqn, count(method) as Methods
```

- Which classes contain the highest number of methods?

```
MATCH
  (class:Class)-[:DECLARES]->(method:Method)
RETURN
  class.fqn, count(method) as Methods
ORDER BY
  Methods DESC
```

- Which classes contain the highest number of methods?

```
MATCH
  (class:Class)-[:DECLARES]->(method:Method)
RETURN
  class.fqn, count(method) as Methods
ORDER BY
  Methods DESC
LIMIT 20
```

- Which class has the deepest inheritance hierarchy?

- Which class has the deepest inheritance hierarchy?

```
MATCH
  h=(class:Class)-[:EXTENDS*]->(super:Type)
```

- Which class has the deepest inheritance hierarchy?

```
MATCH
  h=(class:Class)-[:EXTENDS*]->(super:Type)
RETURN
  class.fqn, length(h) as Depth
```

- Which class has the deepest inheritance hierarchy?

```
MATCH
  h=(class:Class)-[:EXTENDS*]->(super:Type)
RETURN
  class.fqn, length(h) as Depth
ORDER BY
  Depth desc
```

- Which class has the deepest inheritance hierarchy?

```
MATCH
  h=(class:Class)-[:EXTENDS*]->(super:Type)
RETURN
  class.fqn, length(h) as Depth
ORDER BY
  Depth desc
LIMIT 20
```

- Which classes are affected by IOExceptions?

```
MATCH
  (e:Type)-[:DECLARES]->(init:Constructor)
WHERE
  e.fqn="java.io.IOException"
WITH
  e,init
MATCH
  (type:Type)-[:DECLARES]->(method:Method),
  path=(method)-[:INVOKES*]->(init)
RETURN
  e,type,path
LIMIT 10
```

Analysis of software systems using jQAssistant and Neo4j

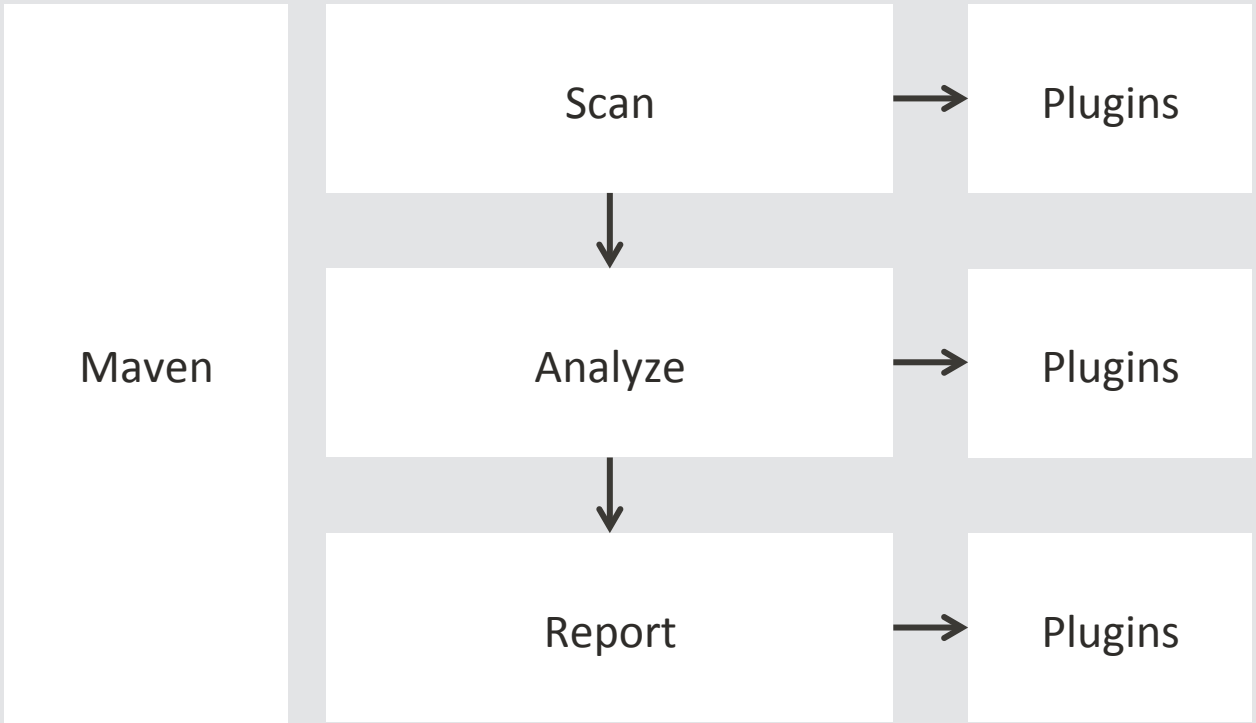
Demo #1

Analysis of software systems using jQAssistant and Neo4j

jQAssistant

- Project Homepage
  - http://jqassistant.org

- Open Source
  - GPL v3
- Milestone
  - 1.0.0-M3 (ASL v2.0), final Release: 12/2014
- Based on Neo4j 2.x
  - Embedded, no installation necessary

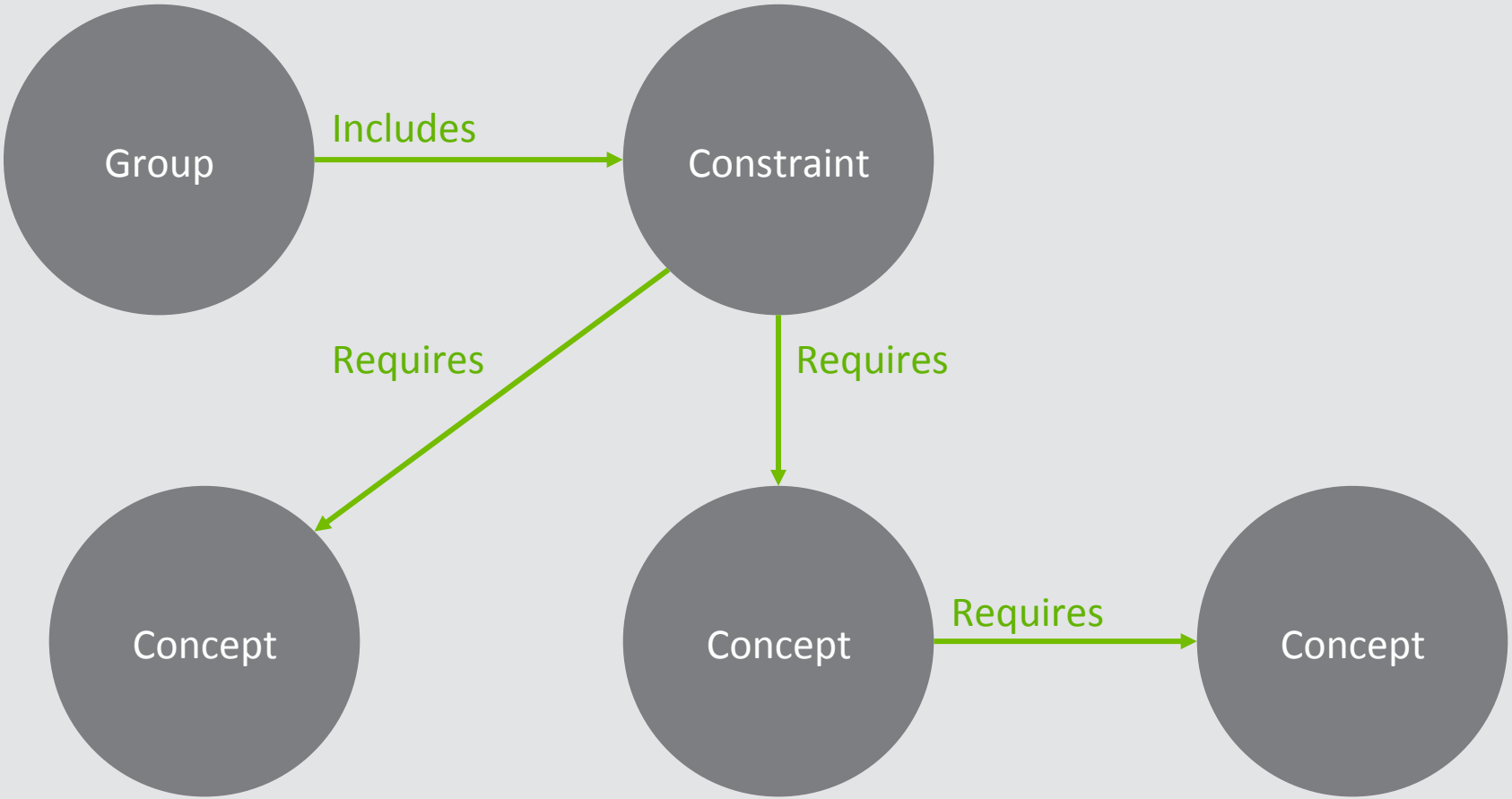

Maven
Scan
Plugins
Analyze
Plugins
Report
Plugins

- Getting started

```
<build>
  <plugins>
    <plugin>
      <groupId>com.buschmais.jqassistant.scm</groupId>
      <artifactId>jqassistant-maven-plugin</artifactId>
    </plugin>
  </plugins>
</build>
```

- mvn jqassistant:scan
- mvn jqassistant:server
- mvn jqassistant:available-rules
- mvn jqassistant:analyze

- Analyze
  - Execution of rules
  - Concepts
    - Enrich data model
  - Constraints
    - Detect violations
  - Group
    - Allow different execution profiles


Group
Includes
Constraint
Requires
Requires
Concept
Concept
Requires
Concept

- Concept

```
<jqa:jqassistant-rules xmlns:jqa="...">

  <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 Entity
    ]]></cypher>
  </concept>

</jqa:jqassistant-rules>
```

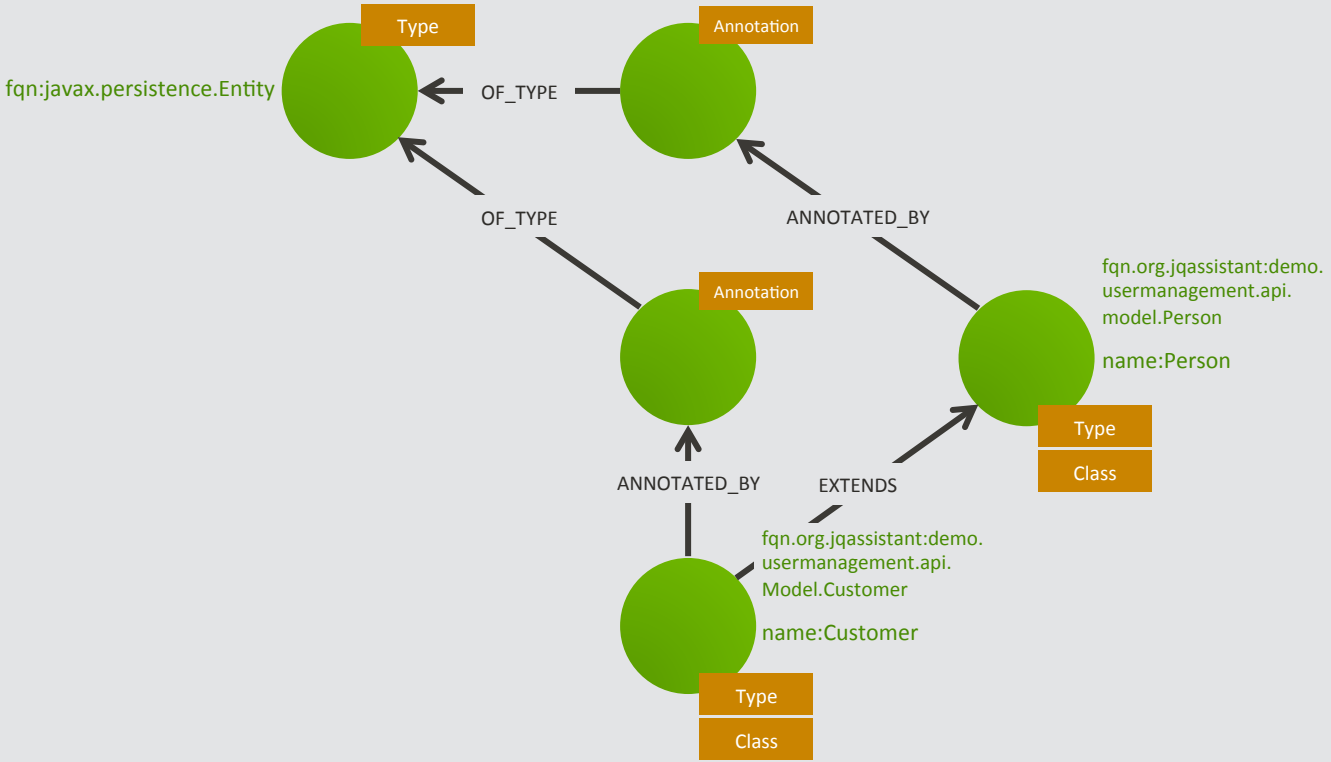
- Concept

```
<jqa:jqassistant-rules xmlns:jqa="...">

  <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 Entity
    ]]></cypher>
  </concept>

</jqa:jqassistant-rules>
```

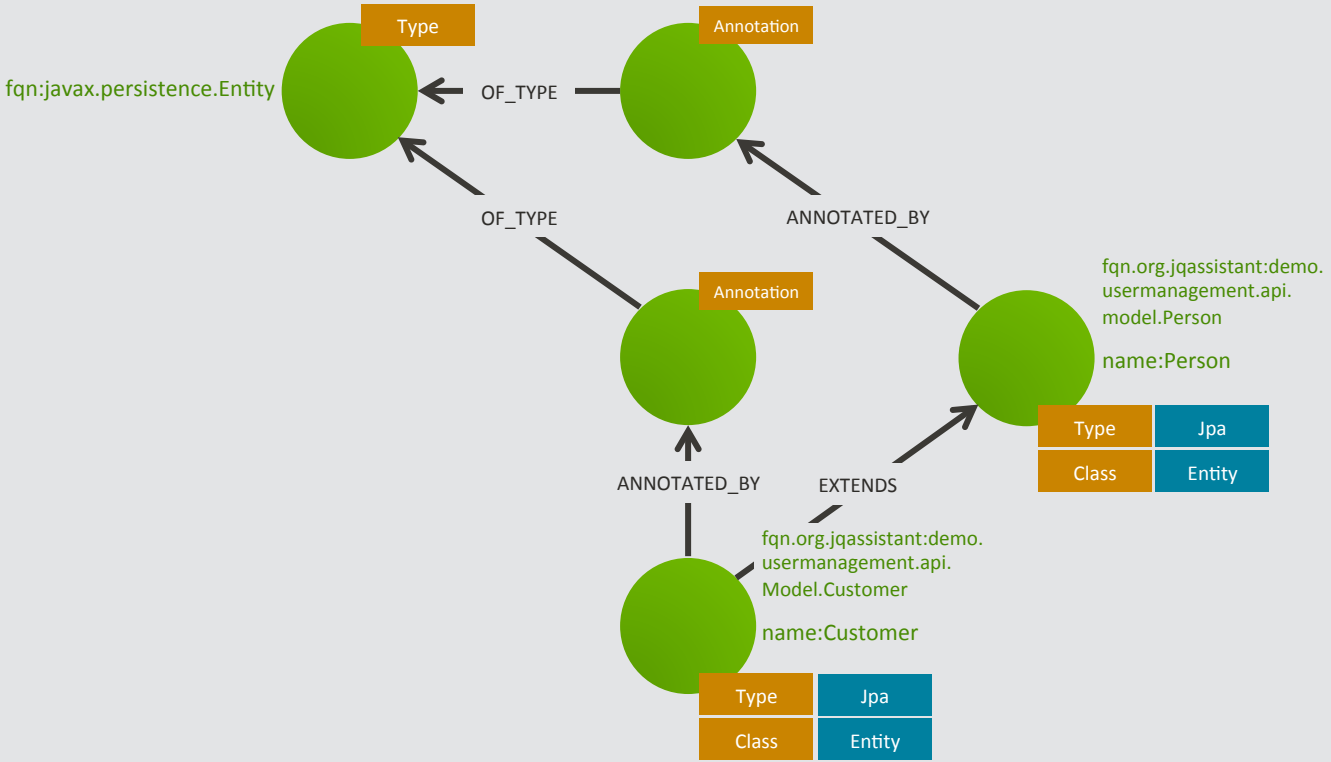
Concept

```
<jqa:jqassistant-rules xmlns:jqa="...">

  <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 Entity
    ]]></cypher>
  </concept>

</jqa:jqassistant-rules>
```

Concept


Type
fqn:javax.persistence.Entity
Annotation
OF_TYPE
OF_TYPE
ANNOTATED_BY
Annotation
fqn.org.jqassistant:demo.
usermanagement.api.
model.Person
name:Person
Type
Class
ANNOTATED_BY
EXTENDS
fqn.org.jqassistant:demo.
usermanagement.api.
Model.Customer
name:Customer
Type
Class

Concept


Type
fqn:javax.persistence.Entity
Annotation
OF_TYPE
OF_TYPE
ANNOTATED_BY
Annotation
fqn.org.jqassistant:demo.
usermanagement.api.
model.Person
name:Person
Type
Jpa
Class
Entity
ANNOTATED_BY
EXTENDS
fqn.org.jqassistant:demo.
usermanagement.api.
Model.Customer
name:Customer
Type
Jpa
Class
Entity

Constraint

```
<jqa:jqassistant-rules xmlns:jqa="...">

  <constraint id="JpaEntitiesInModelPackage">
    <requiresConcept refId="jpa2:Entity"/>
    <description>All JPA entities must be located in
      packages named "model".
    </description>
    <cypher><![CDATA[
      MATCH (p:Package)-[:CONTAINS]->(e)
      WHERE e:Jpa AND e:Entity AND NOT(p.name = "model")
      RETURN
        e AS EntityInWrongPackage
    ]]></cypher>

  </constraint>

</jqa:jqassistant-rules>
```

Constraint

```
<jqa:jqassistant-rules xmlns:jqa="...">

  <constraint id="JpaEntitiesInModelPackage">
    <requiresConcept refId="jpa2:Entity"/>
    <description>All JPA entities must be located in
      packages named "model".
    </description>
    <cypher><![CDATA[
      MATCH (p:Package)-[:CONTAINS]->(e)
      WHERE e:Jpa AND e:Entity AND NOT(p.name = "model")
      RETURN
        e AS EntityInWrongPackage
    ]]></cypher>

  </constraint>

</jqa:jqassistant-rules>
```

Constraint

```
<jqa:jqassistant-rules xmlns:jqa="...">

  <constraint id="JpaEntitiesInModelPackage">
    <requiresConcept refId="jpa2:Entity"/>
    <description>All JPA entities must be located in
      packages named "model".
    </description>
    <cypher><![CDATA[
      MATCH (p:Package)-[:CONTAINS]->(e)
      WHERE e:Jpa AND e:Entity AND NOT(p.name = "model")
      RETURN
        e AS EntityInWrongPackage
    ]]></cypher>

  </constraint>

</jqa:jqassistant-rules>
```

Constraint

```
<jqa:jqassistant-rules xmlns:jqa="...">

  <constraint id="JpaEntitiesInModelPackage">
    <requiresConcept refId="jpa2:Entity"/>
    <description>All JPA entities must be located in
      packages named "model".
    </description>
    <cypher><![CDATA[
      MATCH (p:Package)-[:CONTAINS]->(e)
      WHERE e:Jpa AND e:Entity AND NOT(p.name = "model")
      RETURN
        e AS EntityInWrongPackage
    ]]></cypher>

  </constraint>

</jqa:jqassistant-rules>
```

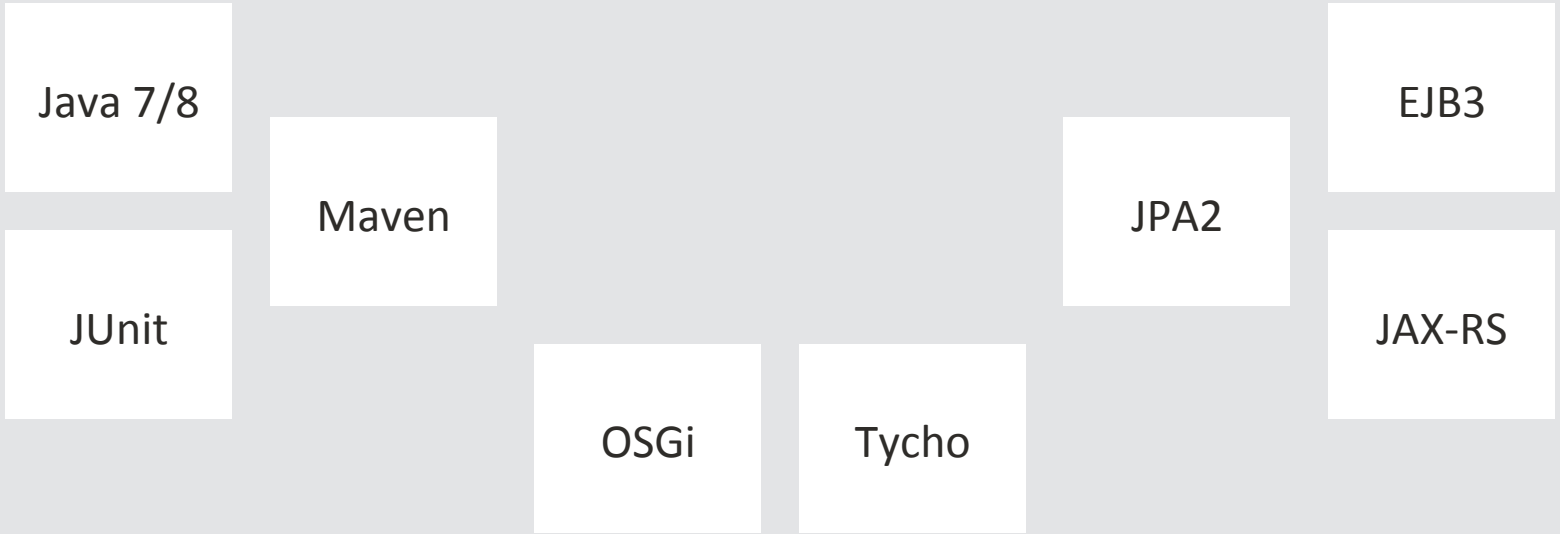
- Group

```
<jqa:jqassistant-rules xmlns:jqa="...">

  <group id="default">
    <includeConstraint
      refId="abstractness:ApiMustNotDependOnImplementation"/>
    <includeConstraint
      refId="JpaEntitiesInModelPackage"/>
    <includeConstraint
      refId="EjbLocatedInImplementationPackage"/>
    <includeConstraint refId="TestClassNameHasTestSuffix"/>
    <includeConstraint refId="dependency:TypeCycles"/>
    <includeConstraint refId="dependency:ArtifactCycles"/>
  </group>

</jqa:jqassistant-rules>
```

- Group

```
<jqa:jqassistant-rules xmlns:jqa="...">

  <group id="default">
    <includeConstraint
      refId="abstractness:ApiMustNotDependOnImplementation"/>
    <includeConstraint
      refId="JpaEntitiesInModelPackage"/>
    <includeConstraint
      refId="EjbLocatedInImplementationPackage"/>
    <includeConstraint refId="TestClassNameHasTestSuffix"/>
    <includeConstraint refId="dependency:TypeCycles"/>
    <includeConstraint refId="dependency:ArtifactCycles"/>
  </group>

</jqa:jqassistant-rules>
```

- Plugins
  - Scanner, Rules, Report



  - Where‘s yours? Contributions welcome!

Analysis of software systems using jQAssistant and Neo4j

Demo #2

- All packages in a Maven module must be prefixed with ${groupId}.${artifactId}

```
MATCH
  (module:Maven)-[:CREATES]->(artifact:Artifact)
WITH
  artifact, artifact.group + "\\." + artifact.name + "\
\..*" as prefixPattern

MATCH
  (artifact)-[:CONTAINS]->(type:Type)
WHERE NOT
  type.fqn =~ prefixPattern
RETURN
  type as TypeInInvalidPackage
```

- Message Driven Beans must have a suffix „MDB“.

```
MATCH
  (mdb:MessageDrivenBean)
WHERE NOT
  mdb.name =~ ".*MDB"
RETURN
  mdb as InvalidBean
```

- JPA entities must be located in „api.model“ packages.

```
MATCH
  (p:Package)-[:CONTAINS]->(e:Jpa:Entity)
WHERE NOT
  p.fqn =~ '.*\\.api\\.model'
RETURN
  e.fqn as Entity
```

- Every test method must contain at least one assertion.

```
MATCH
  (testType:Type)-[:DECLARES]->(testMethod:Test:Method)
WHERE NOT
  (testMethod)-[:INVOKES*..3]->(:Method:Assert)
RETURN
  testType AS DeclaringType,
  testMethod AS Method
```

- Each assertion must provide a human readable message.

```
MATCH
  (testType:Type)-[:DECLARES]->(testMethod:Method),
  (testMethod)-[i:INVOKES]->(assertMethod:Assert:Method)
WHERE NOT
  assertMethod.signature =~
    'void assert.*\\(java.lang.String,.*\\)'
RETURN
  i AS Invocation, // provides lineNumber
  testType AS DeclaringType,
  testMethod AS Method
```

- Remote APIs must be interfaces declaring only primitives or immutables as parameter or return types.

```
MATCH
  (api:Package:API:Remote),
  (api)-[:CONTAINS*]->(interface:Interface),
  (interface)-[:DECLARES]->(method:Method),
  (method)-[:HAS|RETURNS]->(type:Type)
WHERE NOT
  (type:Immutable OR type:Primitive)
RETURN
  interface AS InvalidInterface, type AS InvalidType
```

- OSGi-Bundles must only export dedicated API packages.

```
MATCH
  (bundle:Artifact:Bundle)-[:EXPORTS]->(package:Package)
WHERE NOT
  (package:API)
RETURN
  package AS InvalidExport
```

Thank You! – Questions?

Mail: info@jqassistant.org

Web: jqassistant.org

Twitter: @jqassistant