ulm university universität ulm

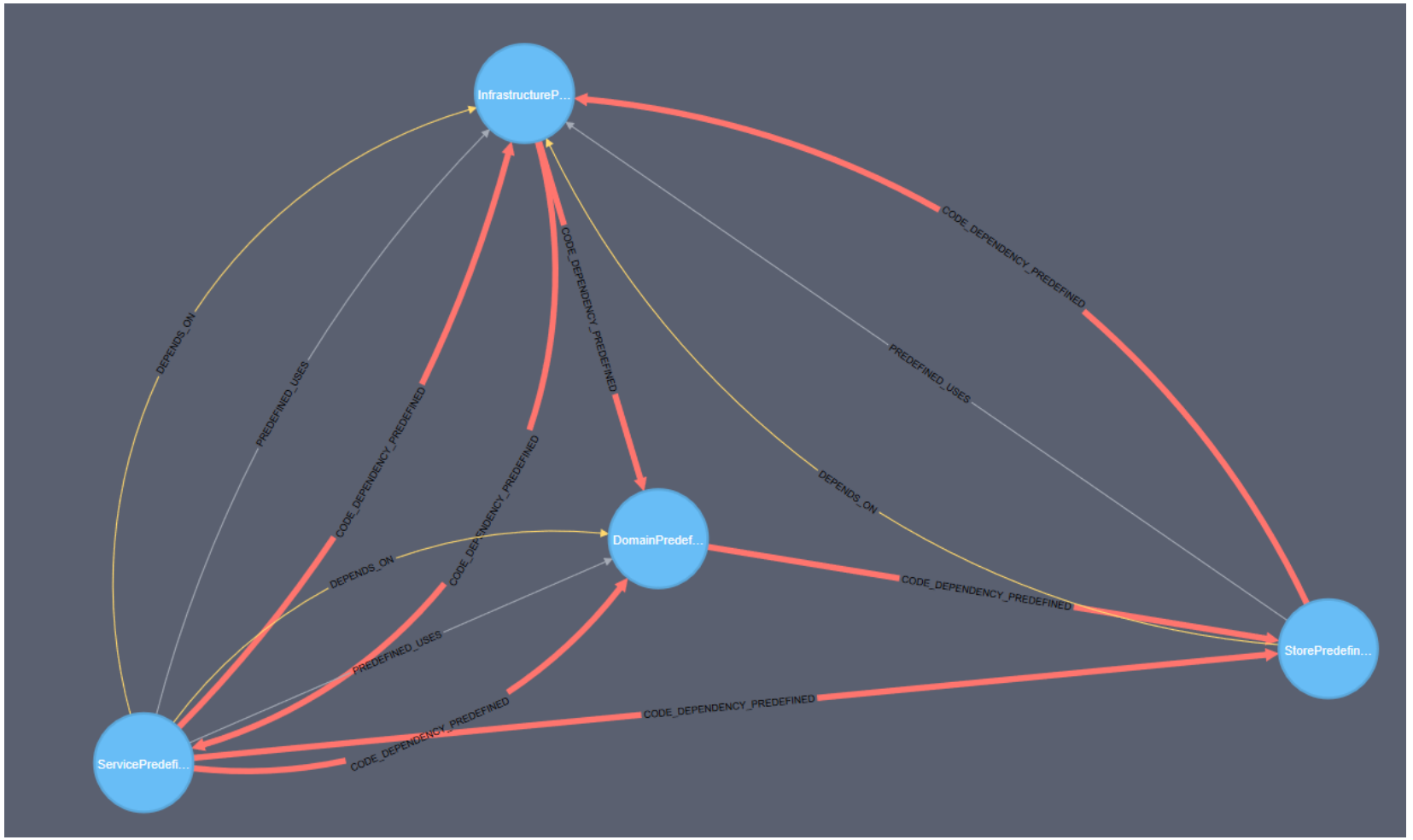
Laura Robien Baldrich,
Institut für Datenbanken und Informationssysteme
Universität Ulm
robien.baldrich@uni-ulm.de
18.12.2018

jQAssistant:
A QA Tool for Definition and Validation of Software Architecture Rules

Agenda

- Objective
- Preliminaries
- Method
- Prototype
- Conclusion

Objective

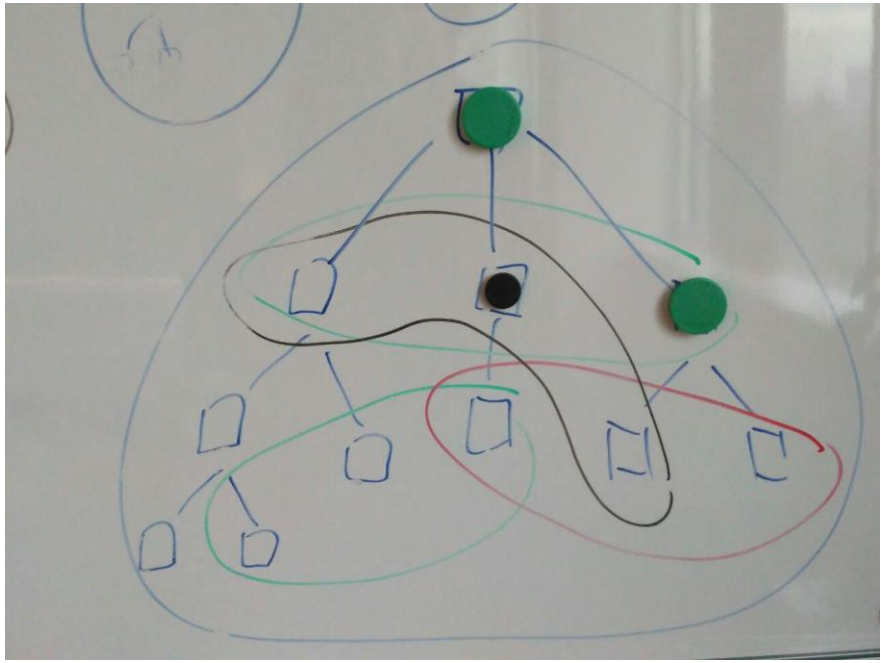

InfrastructureP...
DomainPredef...
StorePredefin...
ServicePredefi...
DEPENDS_ON
PREDEFINED_USES
CODE_DEPENDENCY_PREDEFINED

Objective

- Automation of the static software architecture compliance check
  - What is (static) software architecture?
  - How to automate the process?
- Usage of jQAssistant as source of static code analysis information
  - What is the jQAssistant?
  - How to use the jQAssistant?

"(System) fundamental concepts or properties of a system in its environment embodied in its elements, relationships, and in the principles of its design and evolution.“

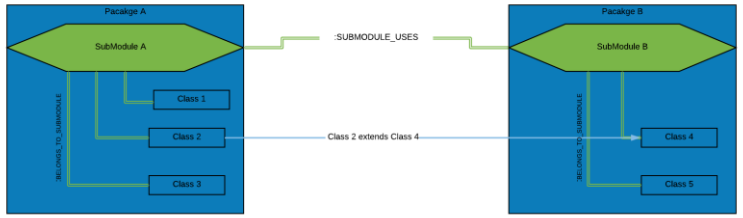
(Medvidovic, N., Taylor, R.N.: A classification and comparison framework for software architecture description languages. IEEE Transactions on software engineering **26** (2000) 70–93)

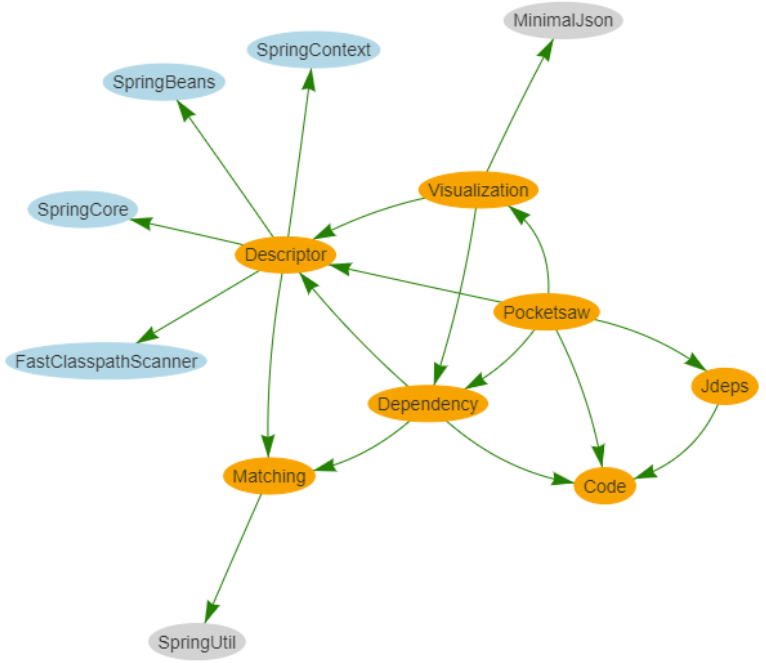


- Elements
- Relationships
- Responsibilities

How to document static architecture?

```java
package com.scheible.pocketsaw.api;

import java.lang.annotation.ElementType;
import java.lang.annotation.Retention;
import java.lang.annotation.RetentionPolicy;
import java.lang.annotation.Target;

@Retention(RetentionPolicy.RUNTIME)
@Target(ElementType.TYPE)
public @interface SubModule {

	/**
	 * Alias for uses();
	 */
	Class<?>[] value() default {};

	/**
	 * Alias for value().
	 */
	Class<?>[] uses() default {};

	boolean includeSubPackages() default true;

	String color() default "orange";
}
```



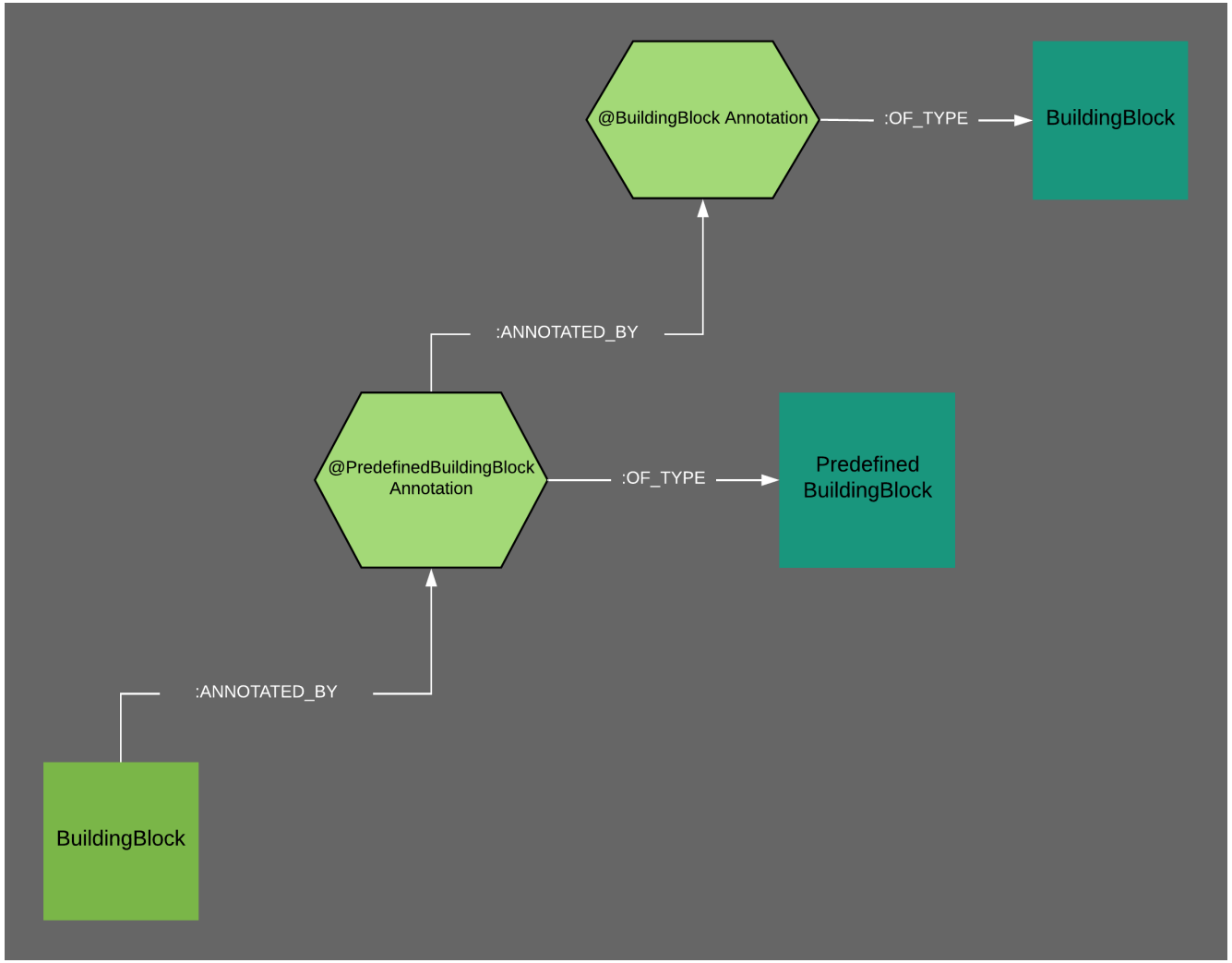


Pocketsaw

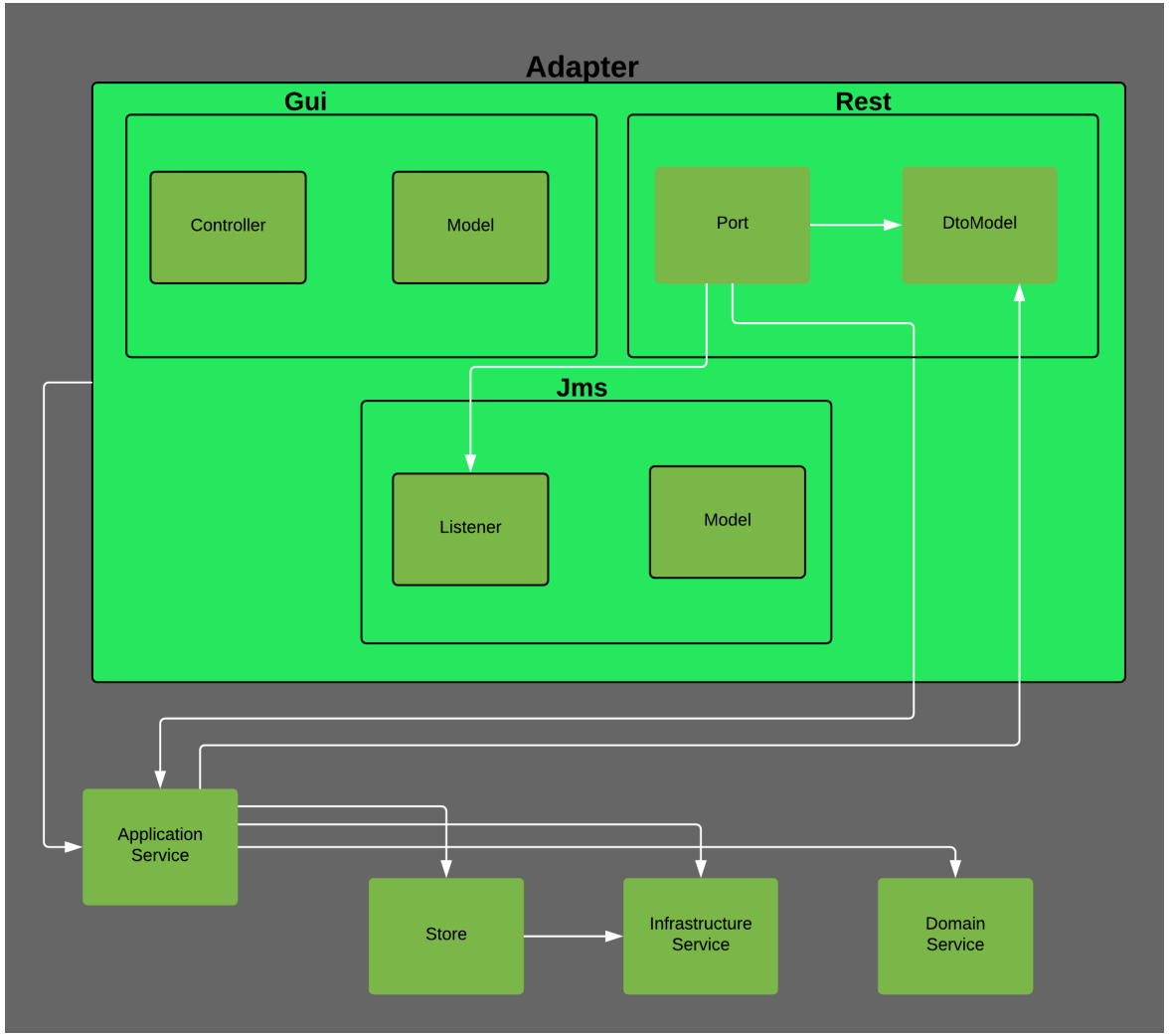
Source:Scheible, Jan: Pocketsaw (2018) https://github.com/janScheible/pocketsaw, last accessed on 2018-10-17.

Building Blocks

Predefined Building Blocks



Building Blocks Java Web Application



```java
package com.tpgroup.buildingblocks.annotation;

import java.lang.annotation.ElementType;
import java.lang.annotation.Retention;
import java.lang.annotation.RetentionPolicy;
import java.lang.annotation.Target;

@Retention(RetentionPolicy.RUNTIME)
@Target(ElementType.TYPE)
public @interface BuildingBlock {

    /**
     * Alias for uses();
     */
    Class<?>[] value() default {};

    /**
     * Alias for value().
     */
    Class<?>[] uses() default {};

    Class<?>[] allowedAnnotations() default {};

    boolean allowAcyclic() default false;

    boolean includeSubPackages() default true;
}
```

Annotated_by

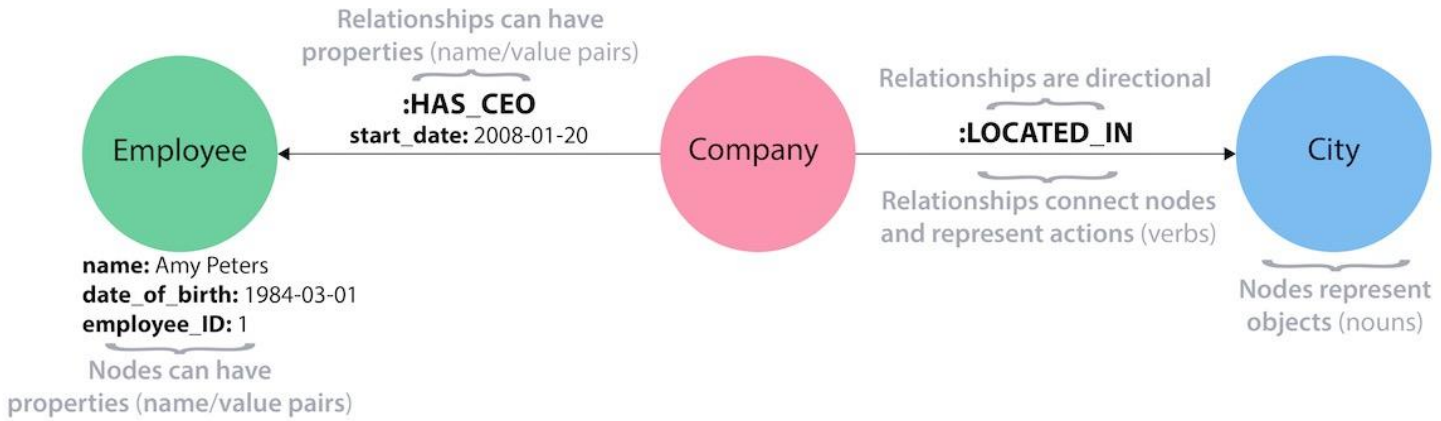
```java
package com.tpgroup.buildingblocks.annotation;


import org.springframework.stereotype.Repository;

import javax.swing.text.html.parser.Entity;
import java.lang.annotation.ElementType;
import java.lang.annotation.Retention;
import java.lang.annotation.RetentionPolicy;
import java.lang.annotation.Target;

@Retention(RetentionPolicy.RUNTIME)
@Target(ElementType.TYPE)
@BuildingBlock(allowedAnnotations = {Entity.class, Repository.class}, uses= InfrastructurePredefinedBuildingBlock.class)
public @interface StorePredefinedBuildingBlock {

    Class<?>[] allowedAnnotations() default {};

    boolean includeSubPackages() default true;

}
```

Annotated_by

```java
package com.tpgroup.refarch.cononical.store.user;

import com.tpgroup.buildingblocks.annotation.StorePredefinedBuildingBlock;

@StorePredefinedBuildingBlock
public class UserStoreBuildingBlock {

}
```

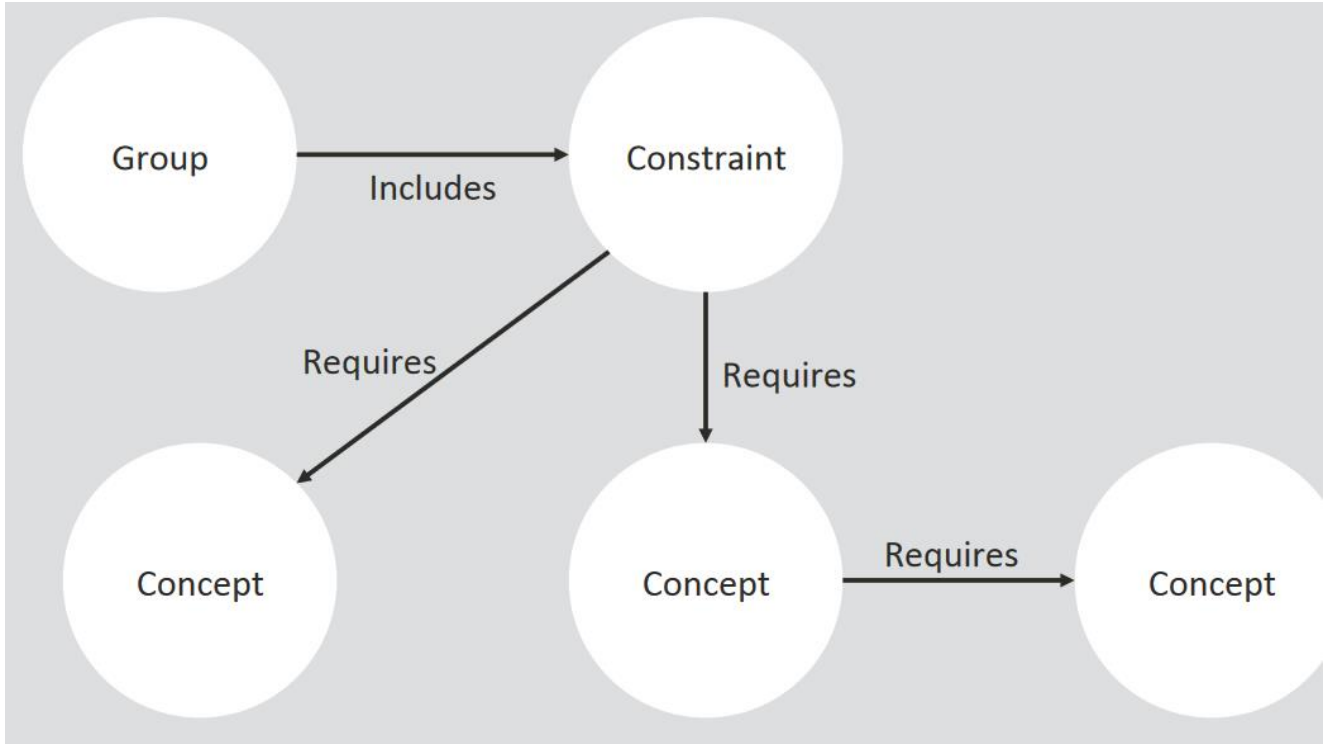
What is jQAssistant?

jQAssistant



- QA tool for definition and validation of software architecture rules
- Can be used via Command line or Maven plugin
- Uses Neo4j graph database and Cypher

Neo4j, Inc.: What is a Graph Database? (2018) https://neo4j.com/developer/graphdatabase/, last accessed on 2018-10-17.



jQAssistant – Concepts & Constraints

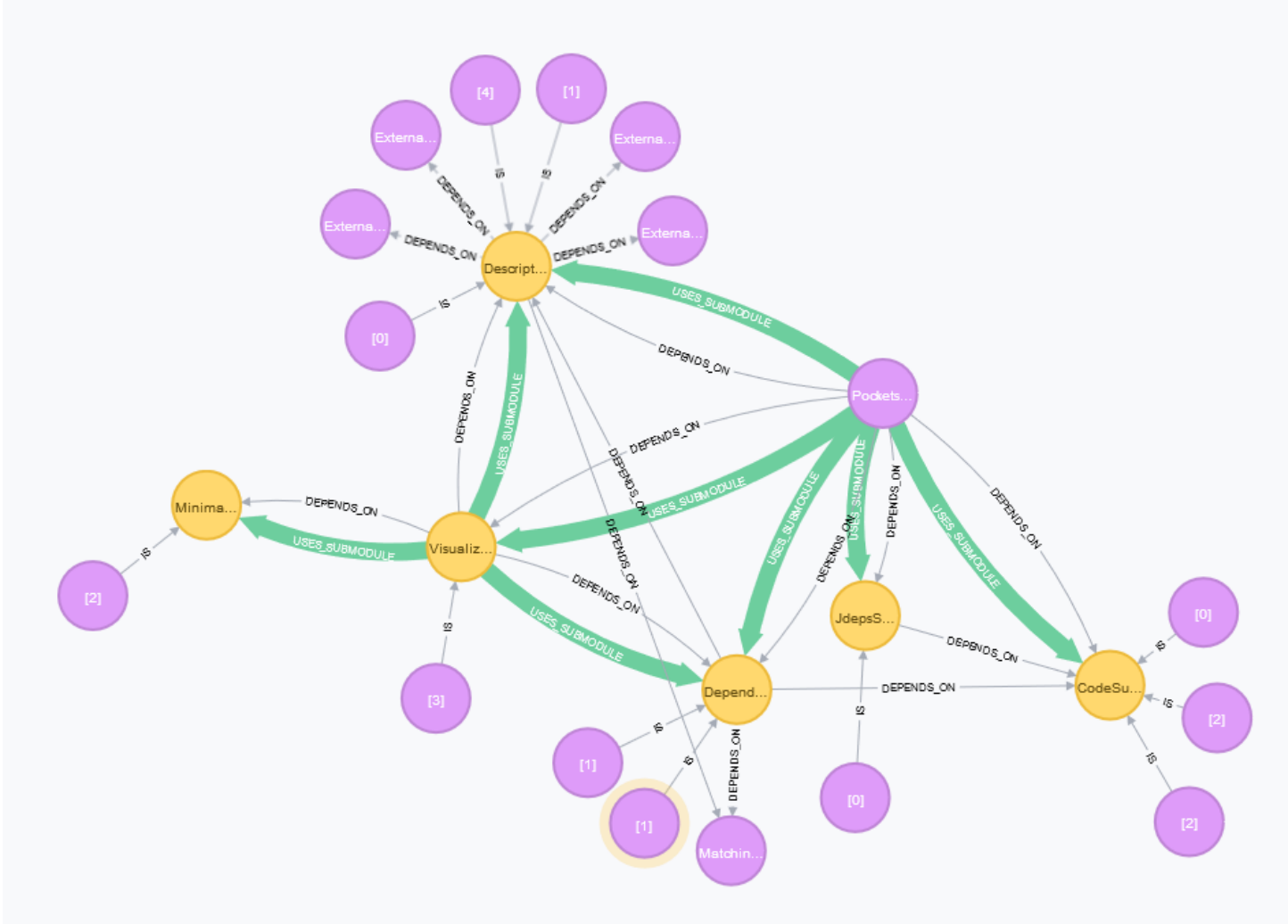
```
<concept id="example:ReturnSuperclasses">
    <requiresConcept refId="example:LabelSuperclass"/>
    <description>Returns all Classes with Superclass Label contained in
        root package.</description>
    <cypher><![CDATA[
        MATCH
            (p:Package)-[:CONTAINS]->(c:Superclass)
        WHERE
                p.name = {rootPackage}
        RETURN
            c
        ORDER BY
            c.fqn
     ]]></cypher>
</concept>
```

```
<constraint id="example:CountSuperClasses">
        <requiresConcept refId="example:LabelSuperclass"/>
        <description>Returns superclasses</description>
        <cypher><![CDATA[
         MATCH (c:Superclass)
         RETURN c
        ]]></cypher>
      <verify>
          <rowCount max="10"/>
      </verify>
</constraint>
```

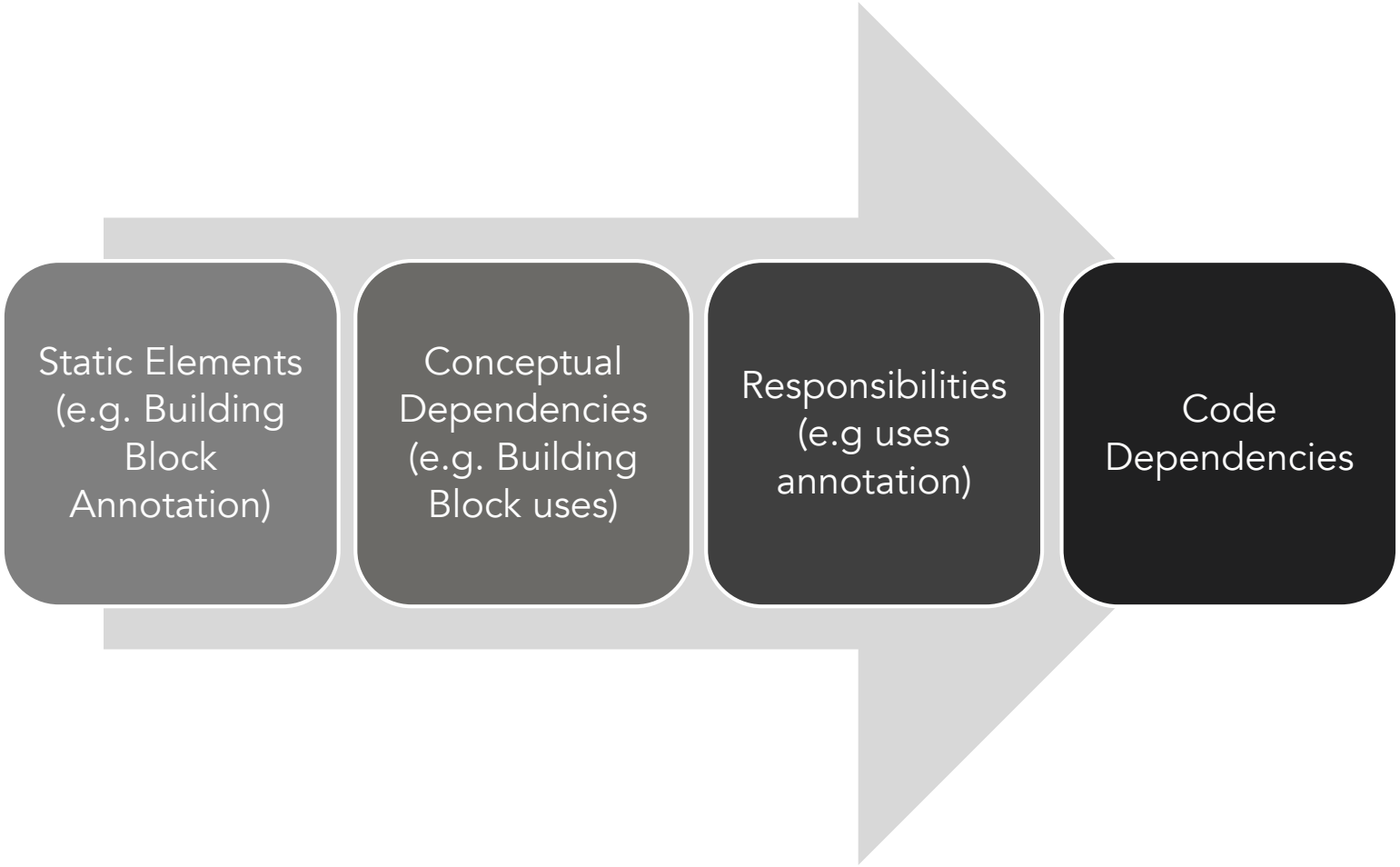
Concepts and Constraints

How to automate the process?

Dependency graph

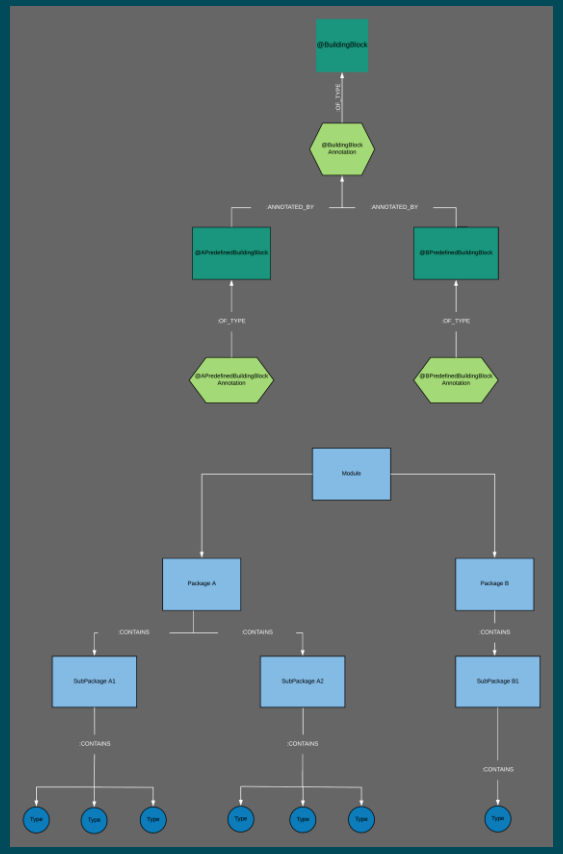


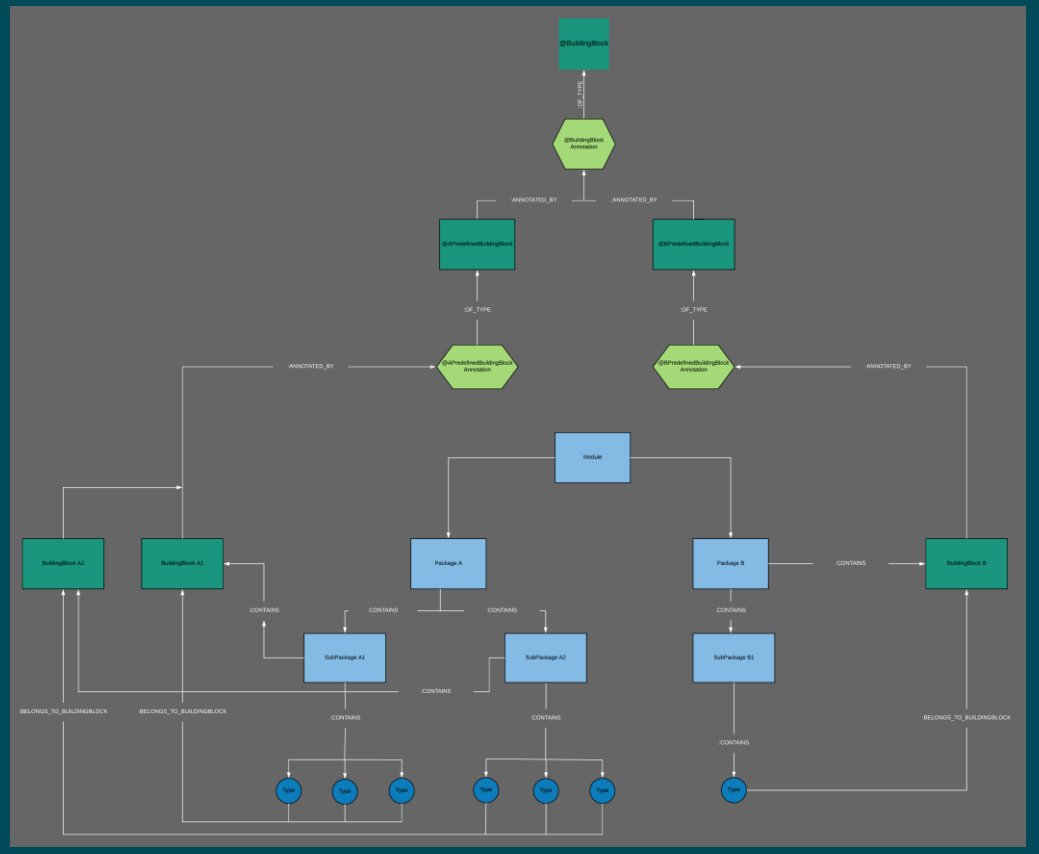
The creation of a dependency graph



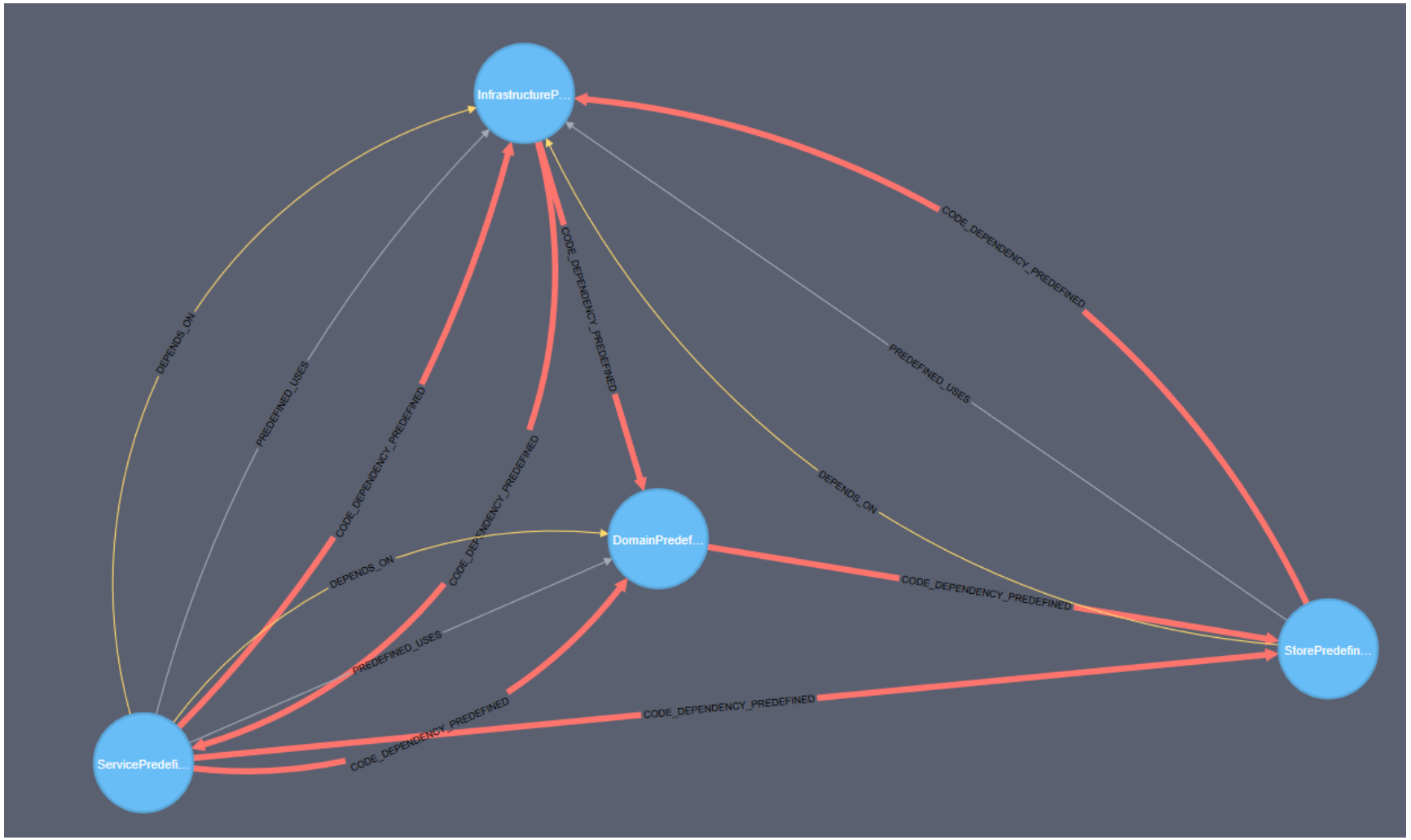
Using the Prototype with jQAssistant

Before and After





Results


InfrastructureP...
DomainPredef...
StorePredefin...
ServicePredefi...
DEPENDS_ON
PREDEFINED_USES
CODE_DEPENDENCY_PREDEFINED

Conclusion