Reporting & Dokumentation

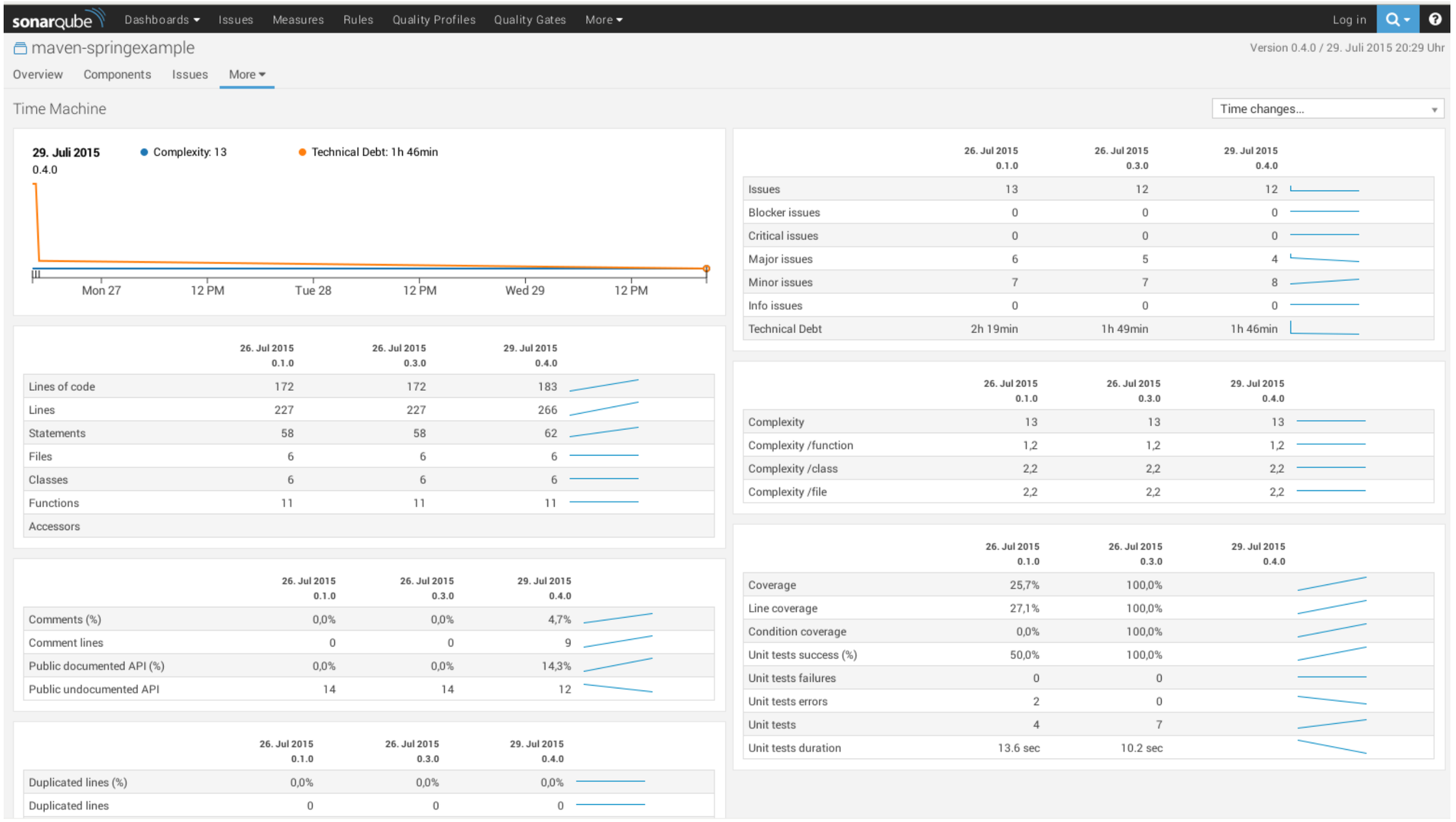
Worum geht es?

- Im Rahmen des Builds können automatisiert Dokumente zum Projekt erzeugt werden
- Reports zur Codequalität
  - Erstellung von Diagrammen, die Qualitätsmetriken (über die Zeit) visualisieren
- Diagramme zu Datenbankschemas
- UML-Diagramme
- Reports mit Informationen aus Drittsystemen (z.B. Bugtracker, Versionsverwaltung)
- Dokumente aus Kommentaren im Quelltext

Codequalität - Erstellung von Reports

- IDE
  - Plugins PMD, EclEmma (JaCoCo), Findbugs, Checkstyle
- Maven
  - Schnelle Ausführung und Rückmeldung im normalen Maven Build
  - Reporterstellung dauert relativ lange
  - meist nur bei neuen Projekten möglich, alte Projekt haben oft "Debts"
  - Vorteil: Ist unabhängig von SonarQube
  - teilweise Integration in Jenkins
- SonarQube
  - Umfangreiche Analysen und Historie
  - "Sonar Way" vs. PMD/Findbugs/Checkstyle (deprecated)

SonarQube


sonarqube
Dashboards
Issues
Measures
Rules
Quality Profiles
Quality Gates
More
Log in
maven-springexample
Version 0.4.0 / 29. Juli 2015 20:29 Uhr
Overview
Components
Issues
More
Time Machine
Time changes...
29. Juli 2015
0.4.0
Complexity: 13
Technical Debt: 1h 46min
Mon 27
12 PM
Tue 28
12 PM
Wed 29
12 PM
26. Jul 2015 0.1.0
26. Jul 2015 0.3.0
29. Jul 2015 0.4.0
Issues 13 12 12
Blocker issues 0 0 0
Critical issues 0 0 0
Major issues 6 5 4
Minor issues 7 7 8
Info issues 0 0 0
Technical Debt 2h 19min 1h 49min 1h 46min
Lines of code 172 172 183
Lines 227 227 266
Statements 58 58 62
Files 6 6 6
Classes 6 6 6
Functions 11 11 11
Accessors
Complexity 13 13 13
Complexity /function 1,2 1,2 1,2
Complexity /class 2,2 2,2 2,2
Complexity /file 2,2 2,2 2,2
Comments (%) 0,0% 0,0% 4,7%
Comment lines 0 0 9
Public documented API (%) 0,0% 0,0% 14,3%
Public undocumented API 14 14 12
Coverage 25,7% 100,0%
Line coverage 27,1% 100,0%
Condition coverage 0,0% 100,0%
Unit tests success (%) 50,0% 100,0%
Unit tests failures 0 0
Unit tests errors 2 0
Unit tests 4 7
Unit tests duration 13.6 sec 10.2 sec
Duplicated lines (%) 0,0% 0,0% 0,0%
Duplicated lines 0 0 0

Empfehlung Codequalität

- Standardregeln vs. angepasste Regeln
- Performance - Abwägung zwischen schnellem Build
- Ist nicht der "heilige Gral"
  - Nicht alle Metriken sind sinnvoll
  - False Positives
  - Aber: Code sauber halten

Javadoc

- Dokumentationswerkzeug, des JDK
  - generiert aus Java-Quellcode HTML-Dateien
  - enthalten sind alle Kommentare die einer bestimmten Struktur entsprechen
  - zur Strukturierung der Dokumentation existieren eigene Annotations
- wird hauptsächlich für API-Dokumentation verwendet
- Konfiguration mittels maven-javadoc-plugin
  - https://maven.apache.org/plugins/maven-javadoc-plugin/

Beispiel generierte Doku

Constructor Summary

Constructors

Constructor and Description
`ArrayList()` Constructs an empty list with an initial capacity of ten.
`ArrayList(Collection<? extends E> c)` Constructs a list containing the elements of the specified collection, in the order they are returned by the collection's iterator.
`ArrayList(int initialCapacity)` Constructs an empty list with the specified initial capacity.

Method Summary

All Methods | **Instance Methods** | **Concrete Methods**

Modifier and Type	Method and Description
boolean	`add(E e)` Appends the specified element to the end of this list.
void	`add(int index, E element)` Inserts the specified element at the specified position in this list.
boolean	`addAll(Collection<? extends E> c)` Appends all of the elements in the specified collection to the end of this list, in the order that they are returned by the specified collection's Iterator.
boolean	`addAll(int index, Collection<? extends E> c)` Inserts all of the elements in the specified collection into this list, starting at the specified position.
void	`clear()` Removes all of the elements from this list.
`Object`	`clone()` Returns a shallow copy of this `ArrayList` instance.
boolean	`contains(Object o)` Returns `true` if this list contains the specified element.
void	`ensureCapacity(int minCapacity)` Increases the capacity of this `ArrayList` instance, if necessary, to ensure that it can hold at least the number of elements specified by the

Beispiel Java-Quellcode

```java
/**
 * Constructs an empty list with the specified initial capacity.
 *
 * @param  initialCapacity  the initial capacity of the list
 * @throws IllegalArgumentException if the specified initial capacity
 *         is negative
 */
public ArrayList(int initialCapacity) {
    super();
    if (initialCapacity < 0)
        throw new IllegalArgumentException("Illegal Capacity: "+
                                           initialCapacity);
    this.elementData = new Object[initialCapacity];
}

/**
 * Constructs an empty list with an initial capacity of ten.
 */
public ArrayList() {
    super();
    this.elementData = EMPTY_ELEMENTDATA;
}

/**
 * Constructs a list containing the elements of the specified
 * collection, in the order they are returned by the collection's
 * iterator.
 *
 * @param c the collection whose elements are to be placed into this list
 * @throws NullPointerException if the specified collection is null
 */
public ArrayList(Collection<? extends E> c) {
    elementData = c.toArray();
    size = elementData.length;
    // c.toArray might (incorrectly) not return Object[] (see 6260652)
    if (elementData.getClass() != Object[].class)
        elementData = Arrays.copyOf(elementData, size, Object[].class);
}

/**
 * Trims the capacity of this <tt>ArrayList</tt> instance to be the
 * list's current size.  An application can use this operation to minimize
 * the storage of an <tt>ArrayList</tt> instance.
 */
public void trimToSize() {
```

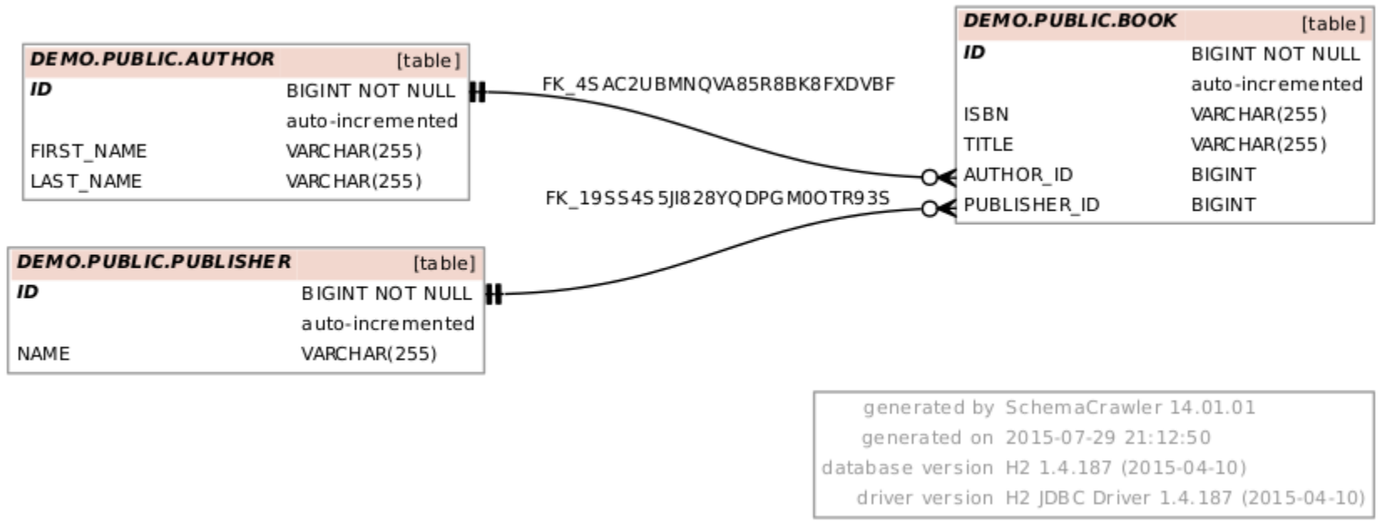
Beispiel Maven Konfiguration

```
<plugin>
  <groupId>org.apache.maven.plugins</groupId>
  <artifactId>maven-javadoc-plugin</artifactId>
  <version>2.10.3</version>
    <configuration>
      ...
    </configuration>
</plugin>
```

Datenbankdiagramm generieren

- Generieren eines ER-Diagramms zu einer bestehenden Datenbank
- Beispielsweise mittels SchemaCrawler (http://sualeh.github.io/SchemaCrawler/)
- Dafür existiert kein fertiges Maven Plugin
- Konfiguration mittels `exec-maven-plugin`

Datenbankdiagramm



mvn site

- Generiert eine HTML Seite aus Informationen in der pom.xml
- Ausserhalb von OpenSource Projekten praktisch keine Verwendung

Übung 5

- Wechsle in das Verzeichnis `if2015-maven` und führe dort `git pull` aus
- Die Sourcen für diese Übung befinden sich im Verzeichnis `uebung-spring3`
- Füge das Repository `http://52.18.220.227:8081/nexus/content/repositories/releases/` zum Downloaden von dependencies hinzu
- Ergänze die Dependency `de.informatica.buildmanagement:documentation:0.5`
- Konfiguriere das JavaDoc Plugin mit dem Custom Doclet `doclets.GlossarDoclet`. Dieses erzeugt ein einfaches Glossar für das Projekt. Setze dabei auch den Konfigurationsparameter `<useStandardDocletOptions>false</useStandardDocletOptions>`
- Generiere mit `mvn javadoc:javadoc` ein Glossar
- Lade SonarQube als ZipFile herunter (http://www.sonarqube.org/downloads/), entpacke und starte es
- Erzeuge mittels `mvn clean install sonar:sonar` einen neuen Report
- Nimm einige Änderungen am Quelltext vor und beobachte wie sich die Metriken