Continuous architecture analysis

with SonarQube in 3D
Software visualization

- Visualization of information about software systems
- Visualization of classes and components
- Simplified risk analysis for all stakeholders
  - reduce risks and costs
  - increase productivity and quality
- Promotes communication about the quality of software between different levels

Why is software visualization so rarely used?
Master Thesis – 2012

City metaphor

○ Structure → Districts

○ Metrics → Building footprint and height
Challenge 1: Context
Challenge 2: Data

JDepend

...?
The software visualization obstacle

How much effort is needed?

○ Platform setup
  - Desktop application
  - Server application

○ Import your project
  - is my language supported?
  - security constraints?

○ Analyse your project
  - which metrics are available
  - integration of specific metrics

○ Report
  - Who can see the result / visualization
Continuous inspection!

- Structure of the software
- Static code analysis
  - Lines of code, complexity
  - Checkstyle, PMD, Findbugs
- Test results (unit and integration tests)
- Management of standards and rules
- Analyse the results on a daily basis over a long period
### Software development cycle at Payback

<table>
<thead>
<tr>
<th>Development environment</th>
<th>Build automation</th>
<th>Deployment</th>
</tr>
</thead>
<tbody>
<tr>
<td>○ Unit tests</td>
<td>○ Unit tests</td>
<td>○ System tests</td>
</tr>
<tr>
<td>○ static code analysis</td>
<td>○ Integration tests</td>
<td>○ Akzeptanz tests</td>
</tr>
<tr>
<td></td>
<td>○ static code analysis</td>
<td>○ Performance tests</td>
</tr>
</tbody>
</table>

**Jenkins**

- Test results
- Errors and warnings of the static code analysis
- Code quality and other metrics
- Risk analysis
Languages
○ Java
○ Web
○ .NET
○ and many more...

Static code analysis
○ Checkstyle
○ PMD, Findbugs
○ Management and export

For everyone!
○ Developer
○ Team lead
○ Project lead

Risk analysis
○ Analyse results
○ Comparable with previous results
○ Tables, diagrams, ...
Software development cycle at Payback

- **Development environment**
- **Build automation**
- **Deployment**

- Export of rules
- Analyse

- Define rules to check
- Store result over a long period
- Show and analyse results

**Deployment**

- Kontinuierliche Architekturanalyse in 3D
## Software development cycle at Payback

<table>
<thead>
<tr>
<th>Development environment</th>
<th>Build automation</th>
<th>Deployment</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="JInva" /></td>
<td><img src="image" alt="maven" /></td>
<td><img src="image" alt="VAA" /></td>
</tr>
</tbody>
</table>

- **Export of rules**
  - Define rules to check
  - Store result over a long period
  - Show and analyse results

---

**Visualization**

- Kontinuierliche Architekturanalyse in 3D
SoftVis3D Sonar plugin - technology

- Webpage plugin for SonarQube
  - Java backend connected via webservice
- AngularJS for the frontend (menu / navigation / interaction)
- GraphViz for the layout
- threeJS as 3D framework
DEMO SoftVis3D
Software architecture = structure?

Package "1"
- Package "2"
  - C1class.java
- Package "3"
  - C2class.java
  - C3class.java
Software architecture = structure + dependencies!

Package "1"
- Package "2"
  - C1class.java
- Package "3"
  - C2class.java
  - C3class.java
Overflow!

<table>
<thead>
<tr>
<th>Style</th>
<th>access</th>
<th>annotation</th>
<th>xml</th>
<th>parsing</th>
<th>support</th>
<th>wiring</th>
<th>service</th>
<th>config</th>
<th>support</th>
<th>annotation</th>
<th>factory</th>
<th>beans</th>
<th>property editors</th>
<th>style</th>
<th>task</th>
<th>enums</th>
<th>type</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>io</td>
<td>4</td>
<td>18</td>
<td>7</td>
<td>11</td>
<td>2</td>
<td>5</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>annotation</td>
<td>2</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>core</td>
<td>6</td>
<td>4</td>
<td>10</td>
<td>10</td>
<td>1</td>
<td>17</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>util</td>
<td>1</td>
<td>12</td>
<td>22</td>
<td>8</td>
<td>44</td>
<td>5</td>
<td>2</td>
<td>34</td>
<td>5</td>
<td>1</td>
<td>3</td>
<td>25</td>
<td>24 1  8  34  120</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>4</td>
</tr>
</tbody>
</table>
Transformation of dependencies I
Transformation of dependencies II
Transformation of dependencies III
Visualization of dependencies I
Visualization of dependencies II
Visualization of dependencies III
Wrap up dependencies

○ Inner dependencies
  - direct connection

○ Dependencies to outside packages
  - via "shortest path" transformed and aggregated

○ Districts are build to the bottom and will be represented in the upper layer

○ "Elevator-Buildings" are the connections between the layers / platforms
  - Includes all incoming and outgoing dependencies of the package

○ Aggregated / abstract view on the dependencies

○ Explorative analysis enabled
Kontinuierliche Architekturanalyse in 3D

DEMO
Wrap up

Code quality management

Continuous inspection of the source code

Consolidate and analyse results

Visualization as additional tool!
SoftVis3d Sonarqube plugin

○ Easy installationn as SonarQube Plugin
○ Open source: https://github.com/stefanrinderle/sonar-softvis3d-plugin

○ Visualization of all existing Metrics in SonarQube
○ City view / Dependency view
○ Explorative user interface

○ Details: http://softvis3d.com