

Understanding Large Codebases



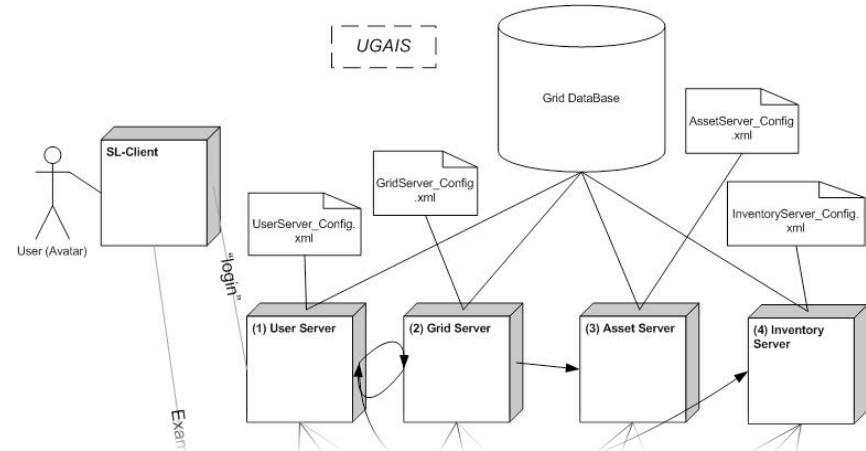
```

public void assemblePlugin(BuildDetail build, Map parameters) {
    Iterator<String> iterator = parameters.keySet().iterator();
    while (iterator.hasNext()) {
        String key = iterator.next();
        try {
            parameters.put(key, (String) iterator.next());
        } catch (Exception e) {
            log.error("Error merging parameter " + key);
        }
    }
}

void assemblePlugin(BuildDetail build, Map parameters, String className) {
    String line = null;
    if (className.startsWith("#")) {
        return;
    }
    Class clazz = Class.forName(className);
    Widget digesterService = (Widget) clazz.newInstance();
    mergeParameters(build, parameters);
    build.addPluginOutput(digesterService.getDisplayName(), c
        .getOutput(parameters));
}

private void mergeParameters(BuildDetail build, Map parameter
    parameters.put(Widget.PARAM_CC_ROOT, configuration.getCC
    parameters.put(CC_PARAM_NAME, buildId);
}

```



micro <=> macro



A photograph of a stack of papers with a calculator resting on top. The calculator screen displays the number '123'.

metrics

cyclomatic complexity

provides a numeric value representing the complexity of a function or method

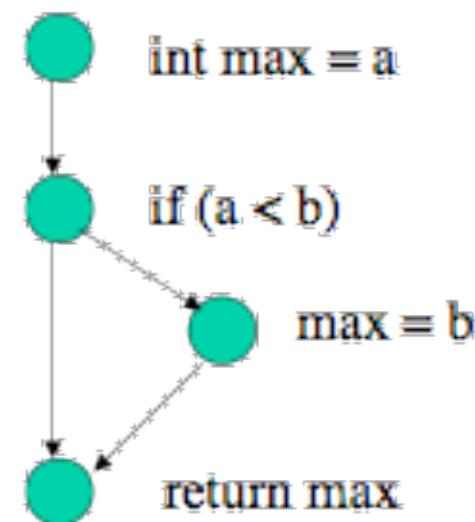
$$V(G) = e - n + 2$$

$V(G)$ = cyclomatic complexity of G

e = # edges

n = # of nodes

```
int max (int a, int b) {  
    int max = a;  
    if (a < b) {  
        max = b;  
    }  
    return max;  
}
```

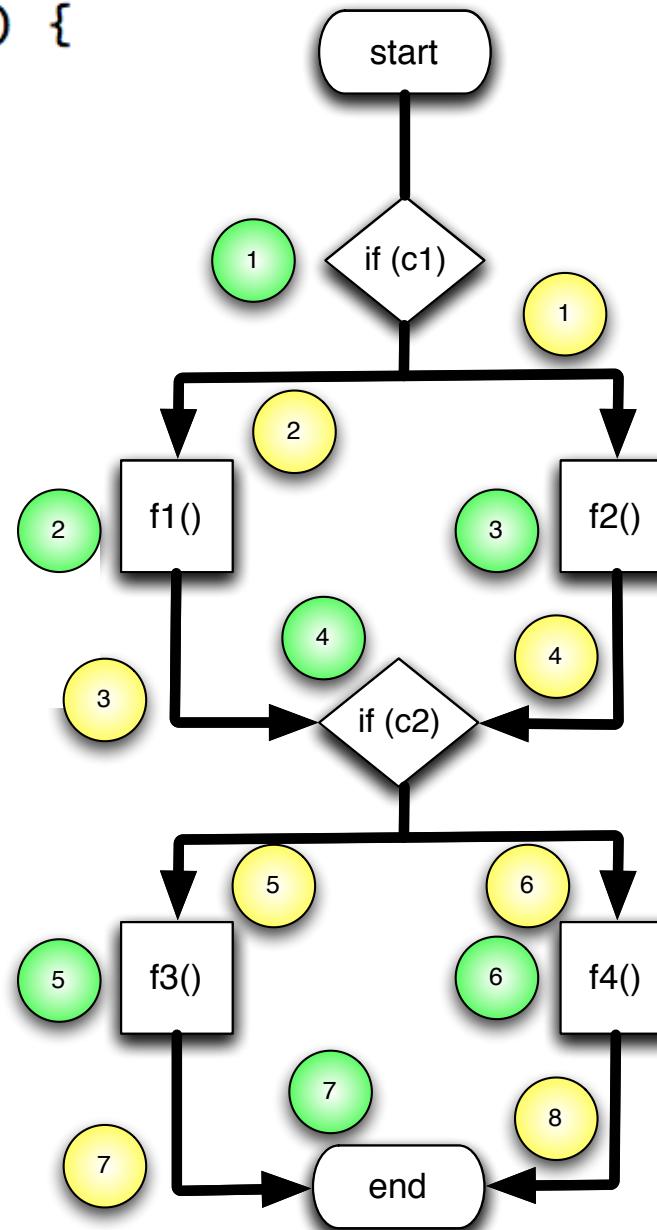


```

public void doIt() {
    if (c1) {
        f1();
    } else {
        f2();
    }
    if (c2) {
        f3();
    } else {
        f4();
    }
}

```

nodes
 edges



Chidamber & Kemerer object-oriented metrics

Shyam R. Chidamber
Chris F. Kemerer

[http://faculty.salisbury.edu/~stlauterburg/cosc425/
metricforood_chidamberkemerer94.pdf](http://faculty.salisbury.edu/~stlauterburg/cosc425/metricforood_chidamberkemerer94.pdf)

easy (but trivial)

DIT	depth of inheritance tree
NOC	number of children
NPM	number of public methods

quite useful

WMC	weighted methods/class	Σ of cyclomatic complexity
RFC	response for class	# of methods executed due to method call
LCOM	lack of cohesion	Σ of sets of methods not shared via sharing fields
CE	efferent coupling	Σ of other classes this class uses (outgoing calls)
CA	afferent coupling	Σ of how many other classes use this class (incoming calls)

visualizations

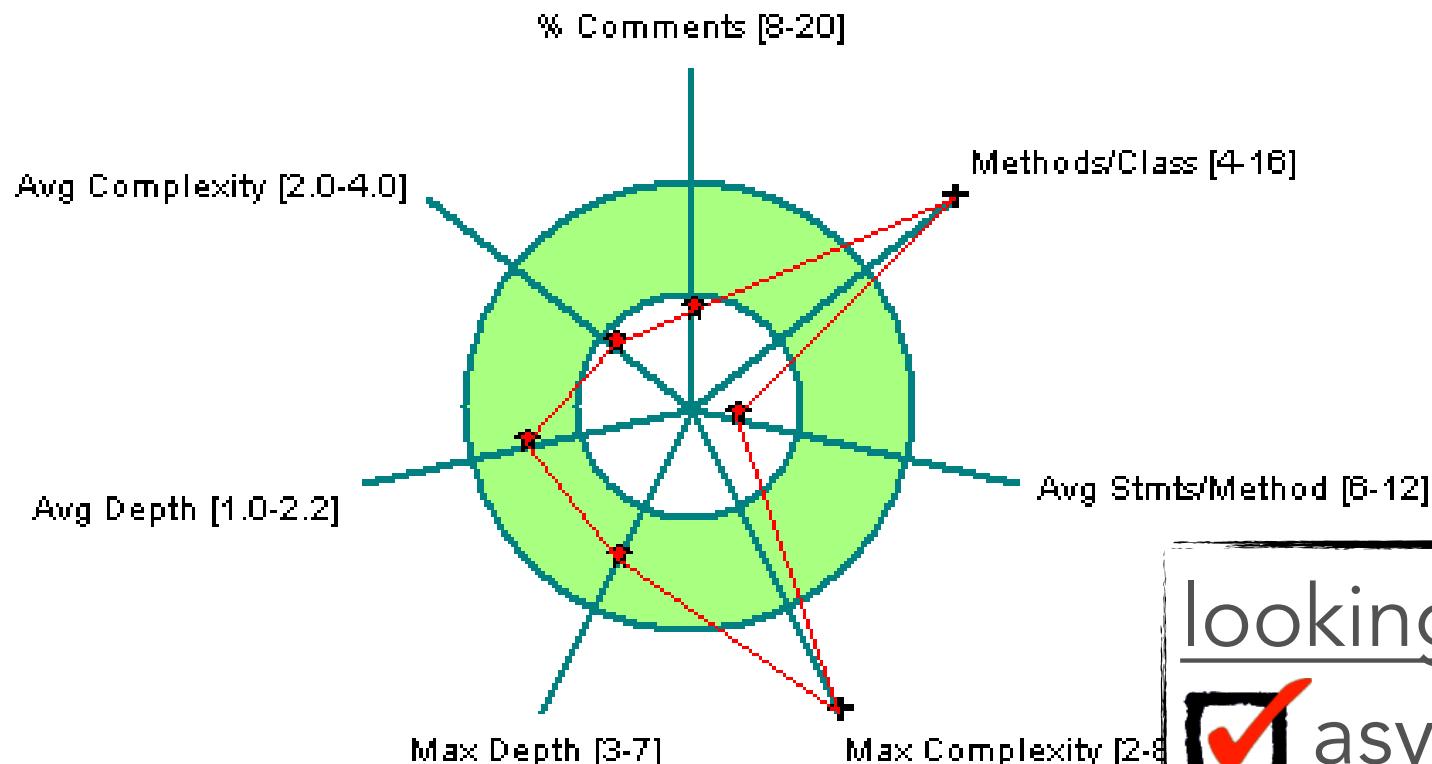


Source Monitor

<http://www.campwoodsw.com/sourcemonitor.html>

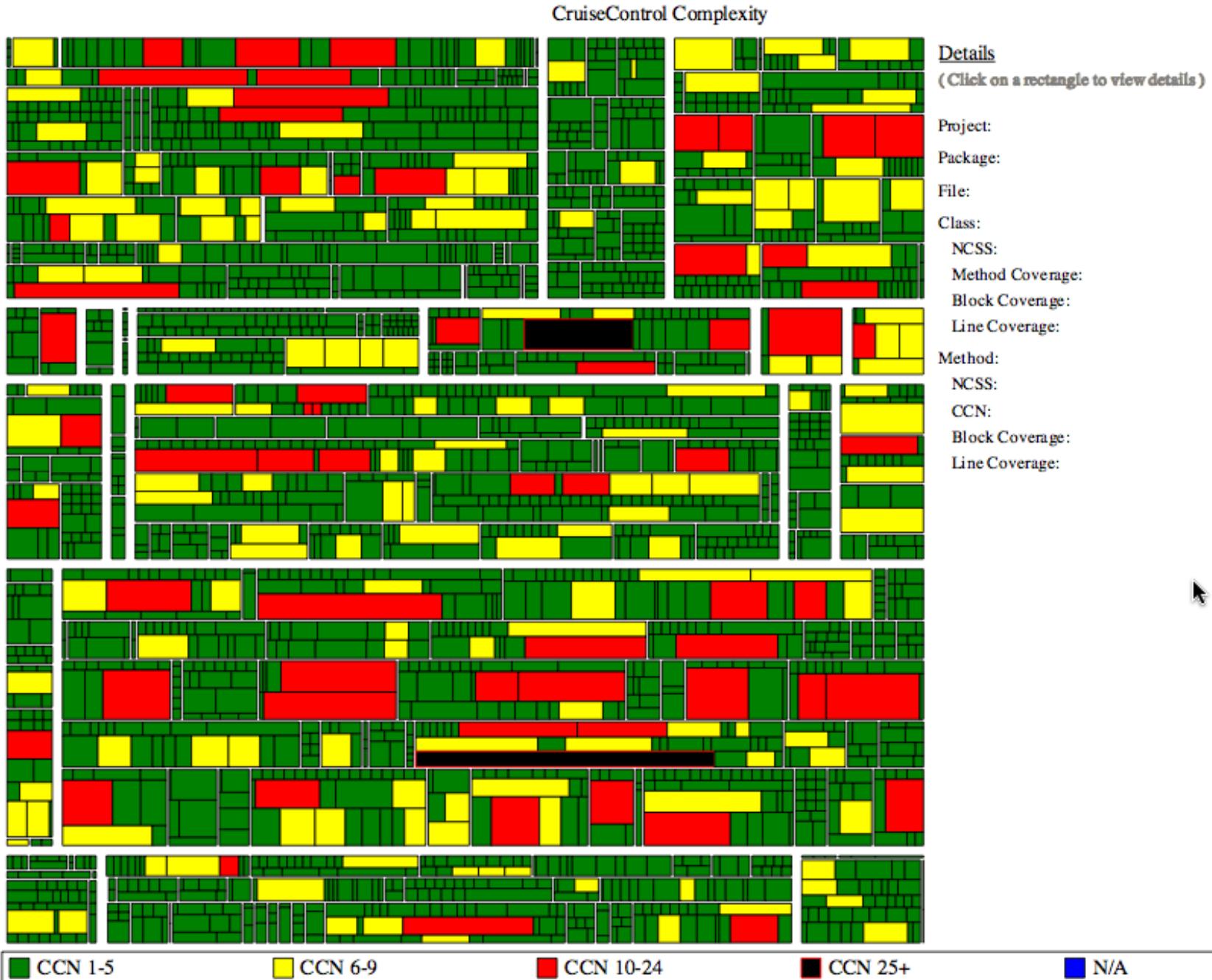


Kiviat Metrics Graph: Project 'core'
Checkpoint 'Baseline'
File 'src\main\java\org\apache\struts2\components\DoubleListUIBean.java'

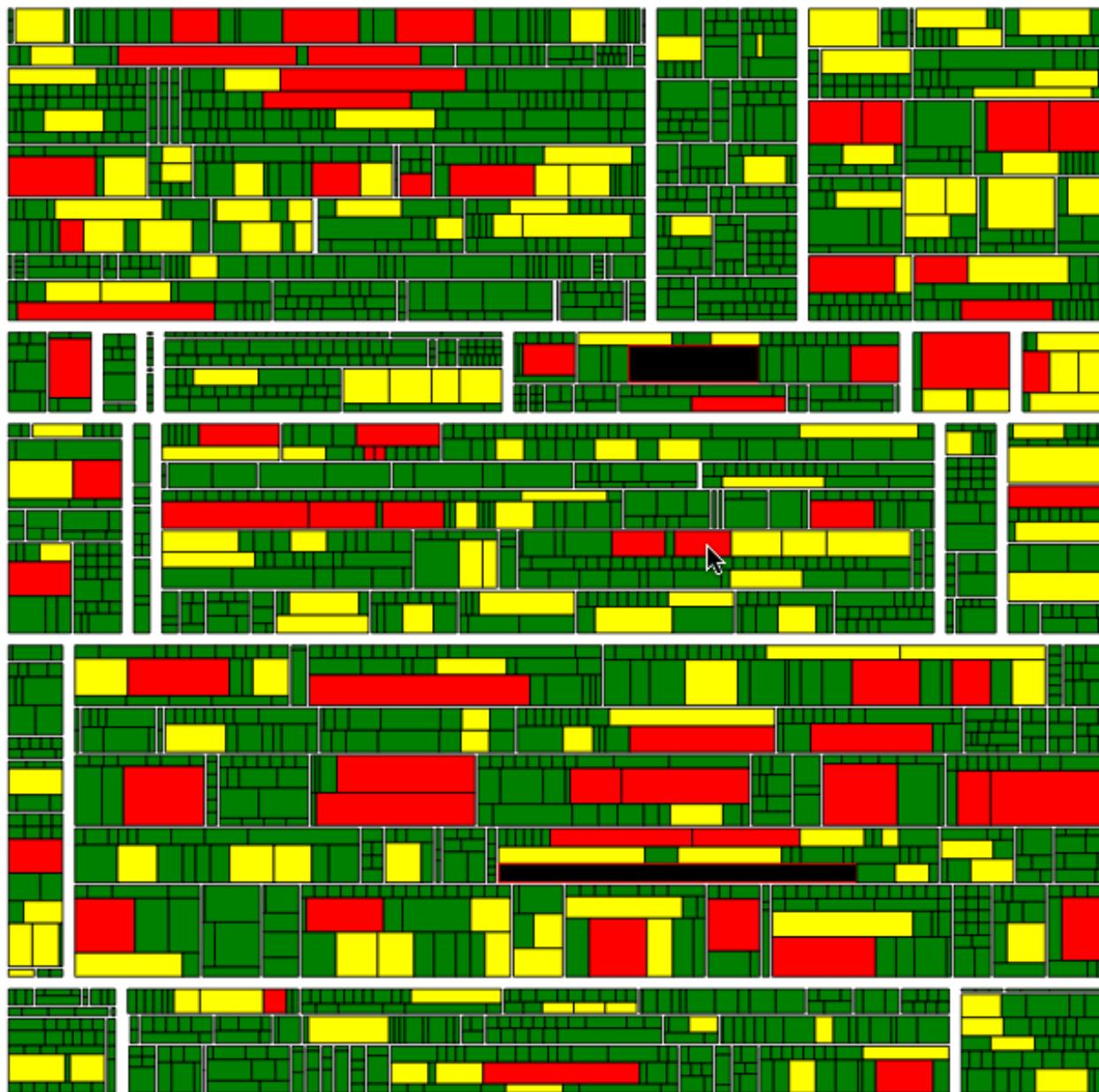


looking for:

- asymmetries
- outliers



CruiseControl Complexity



Details

(Click on a rectangle to view details)

Project: CruiseControl

Package: net.sourceforge.cruisecontrol.publishers

File: EmailPublisher.java

Class: EmailPublisher

NCSS: 345

Method Coverage: 72.0% (36.0/50.0)

Block Coverage: 67.4% (819.0/1215.0)

Line Coverage: 68.2% (187.6/275.0)

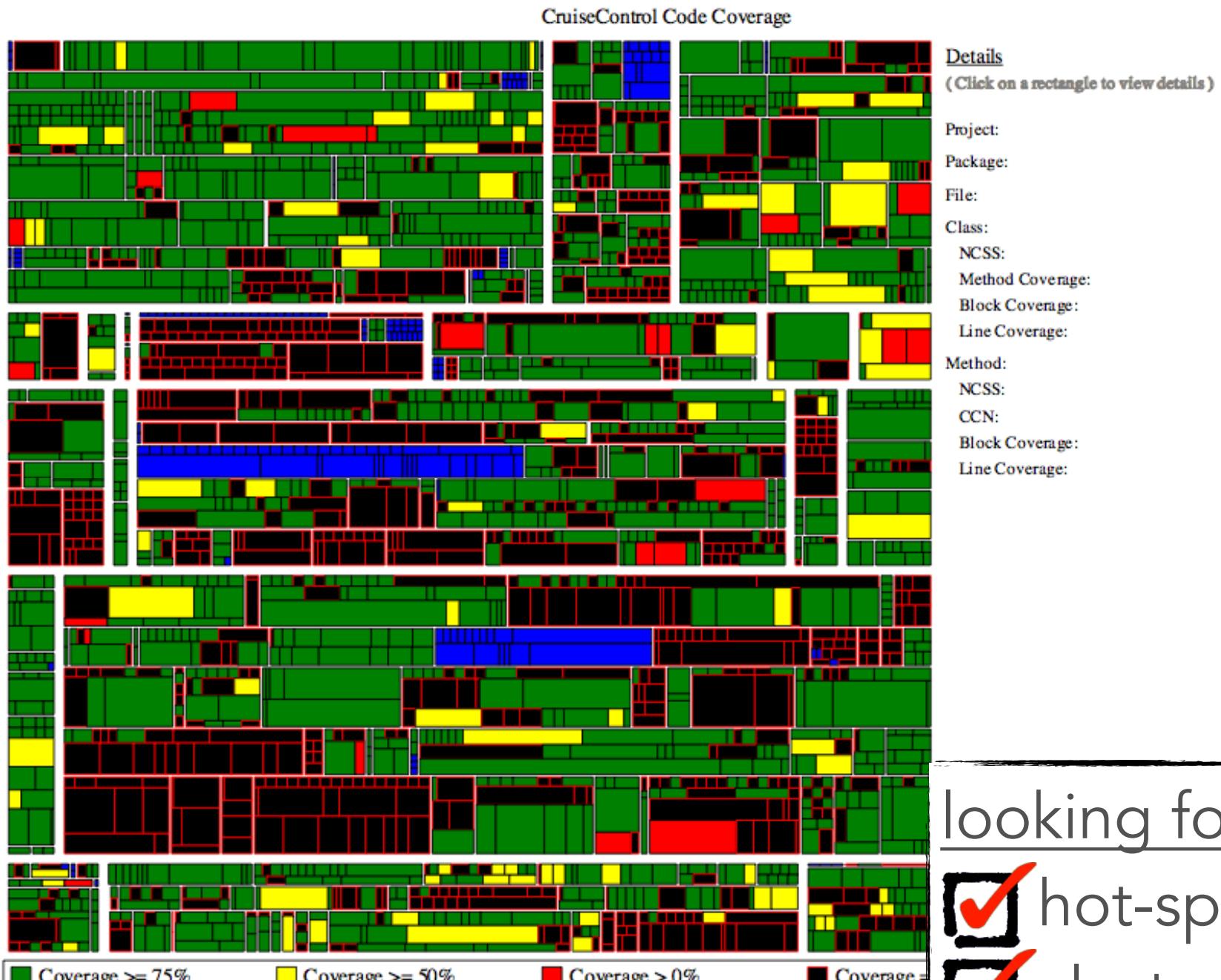
Method: createUserSet(XMLLogHelper)

NCSS: 19

CCN: 11

Block Coverage: 100.0% (122.0/122.0)

Line Coverage: 100.0% (18.0/18.0)



looking for:

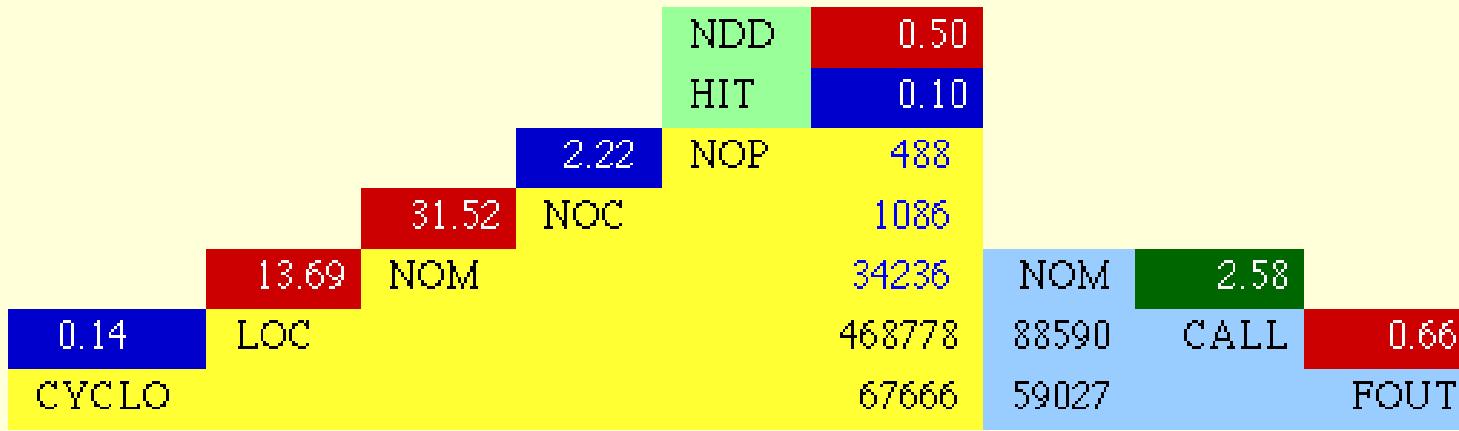
hot-spots
 clusters

size & complexity pyramid

0.26	LOC	5.69	NOM	6.13	NOC	5.75	NOP	224	1289	7905	7905	NOM	2.79	CALLS	0.40	FANOUT
CYCLO								44988	22039	8798						

	Low	Medium	High
CYCLO / Line	0.16	0.20	0.24
LOC / method	7	10	13
NOM / class	4	7	10
NOC / package	6	17	26
CALLS / method	2.01	2.62	3.20
FANOUT / call	0.56	0.62	0.68

vuze



Interpretation of the Overview Pyramid for module
[/Users/nealford/Downloads/Applications/Vuze_4.2.0.2_source.zip Folder](#)

Class Hierarchies tend to be **shallow** and **wide**

(i.e. inheritance trees tend to have only few depth-level(s) and base-classes with many directly derived sub-classes)

Classes tend to:

- be rather **large** (i.e. they define many methods);
- be organized in rather **fine-grained packages** (i.e. few classes per package);

Methods tend to:

- tend to be rather **long** yet having a rather **simple logic** (i.e. few conditional branches);
- tend to call an **several methods** from **many other classes** (high coupling dispersion);

Design Flaws Overview

	NDD	0.6
HIT	1.6	
14.13	NOP	30
6.72	NOC	42
8.17	NOM	285
0.21	LOC	2332
CYCLO		506

Interpretation

Class hierarchies tend to be **tall** and **wide**, classes with many directly derived sub-classes.

Classes tend to be:

- contain an **average** number of methods
- be organized in **average-sized** packages

Methods tend to:

- be rather **short** and having an **average** number of parameters
- call **several methods** from **few components**

Legend:

- URLTag
- AnchorTag
- DoubleSelectTag
- Component

Loading the model took: 1m 1s

Package Map

Coupling perspective on system main

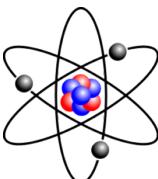
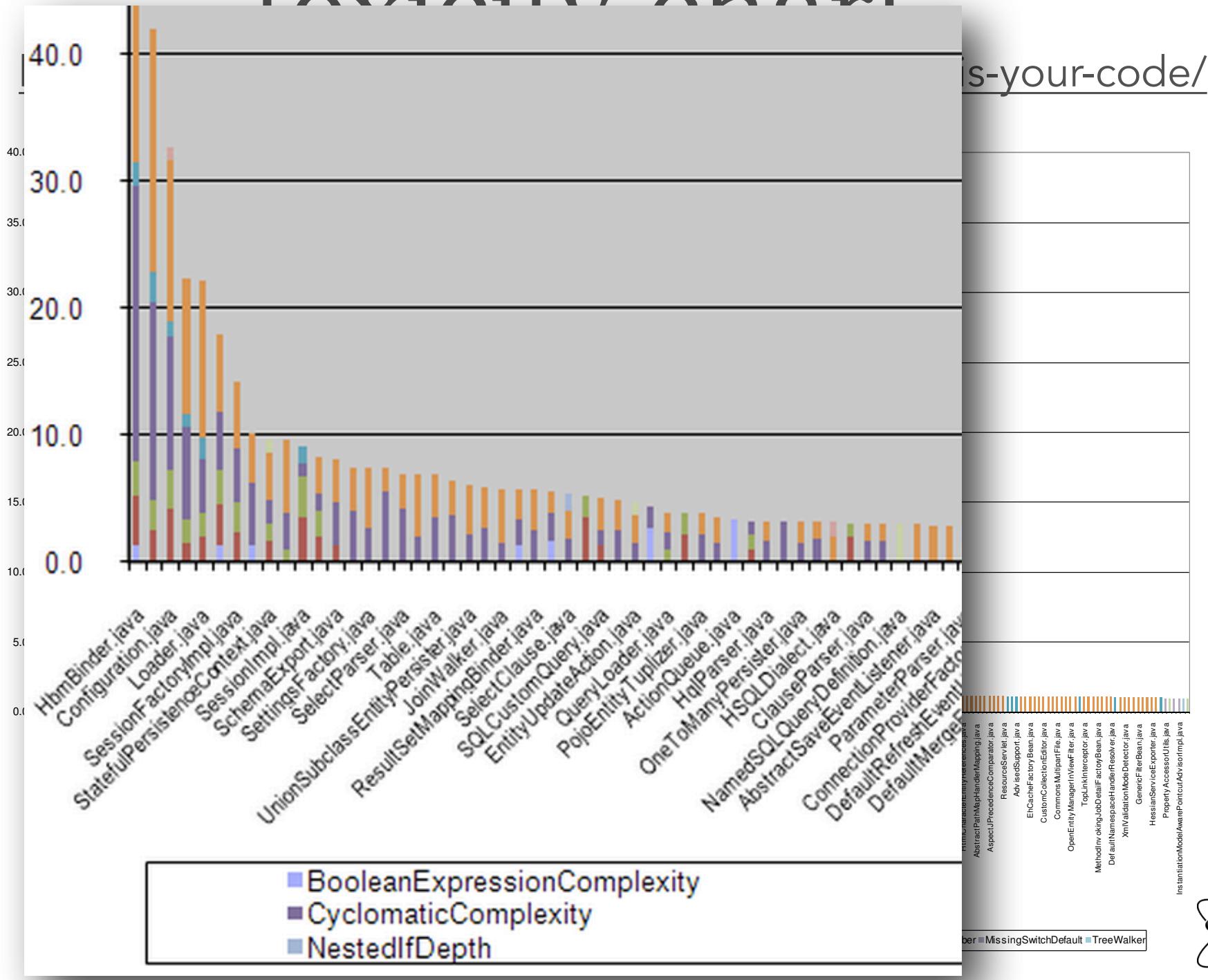
looking for:

holistic compliance

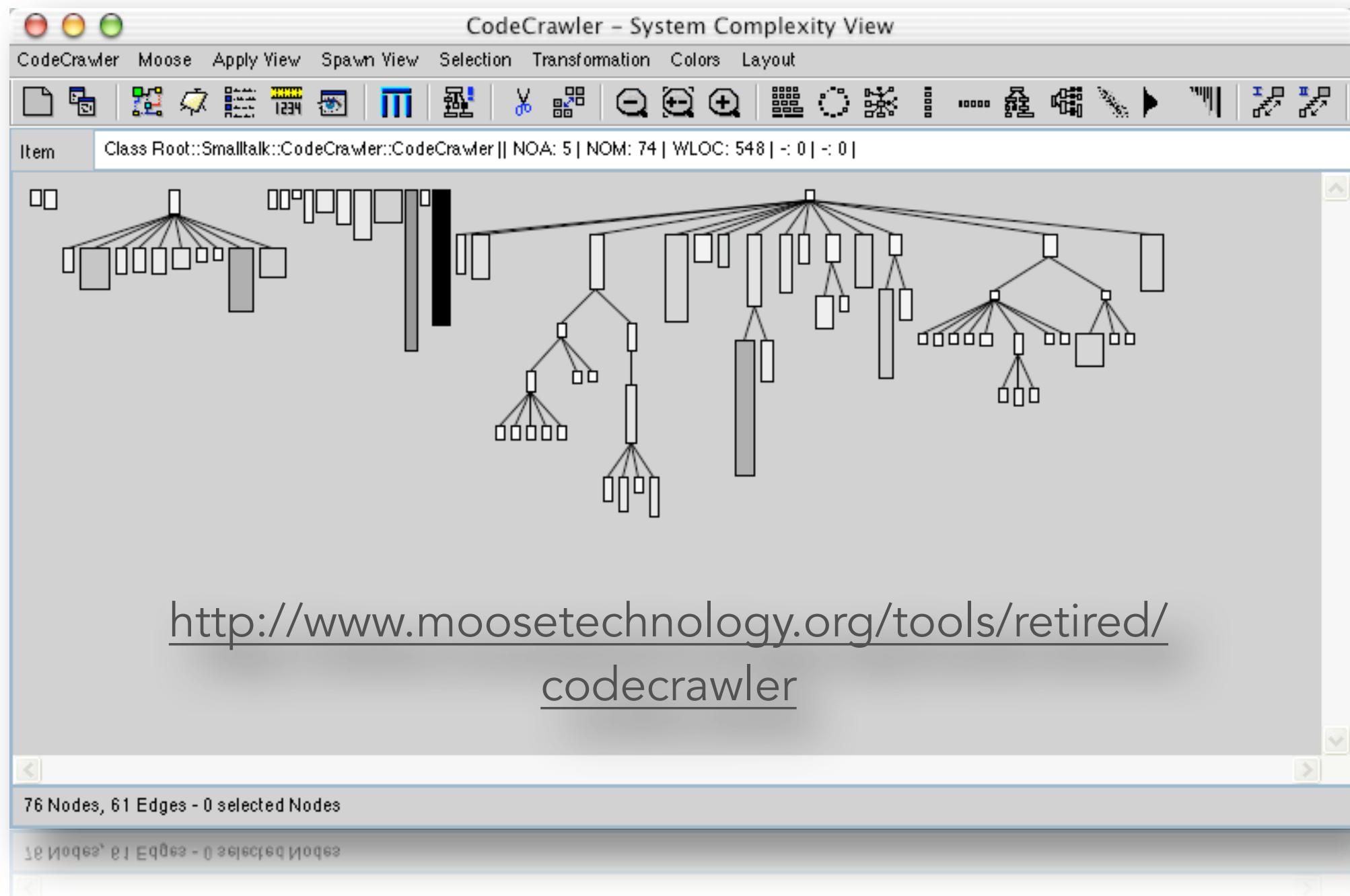
outliers

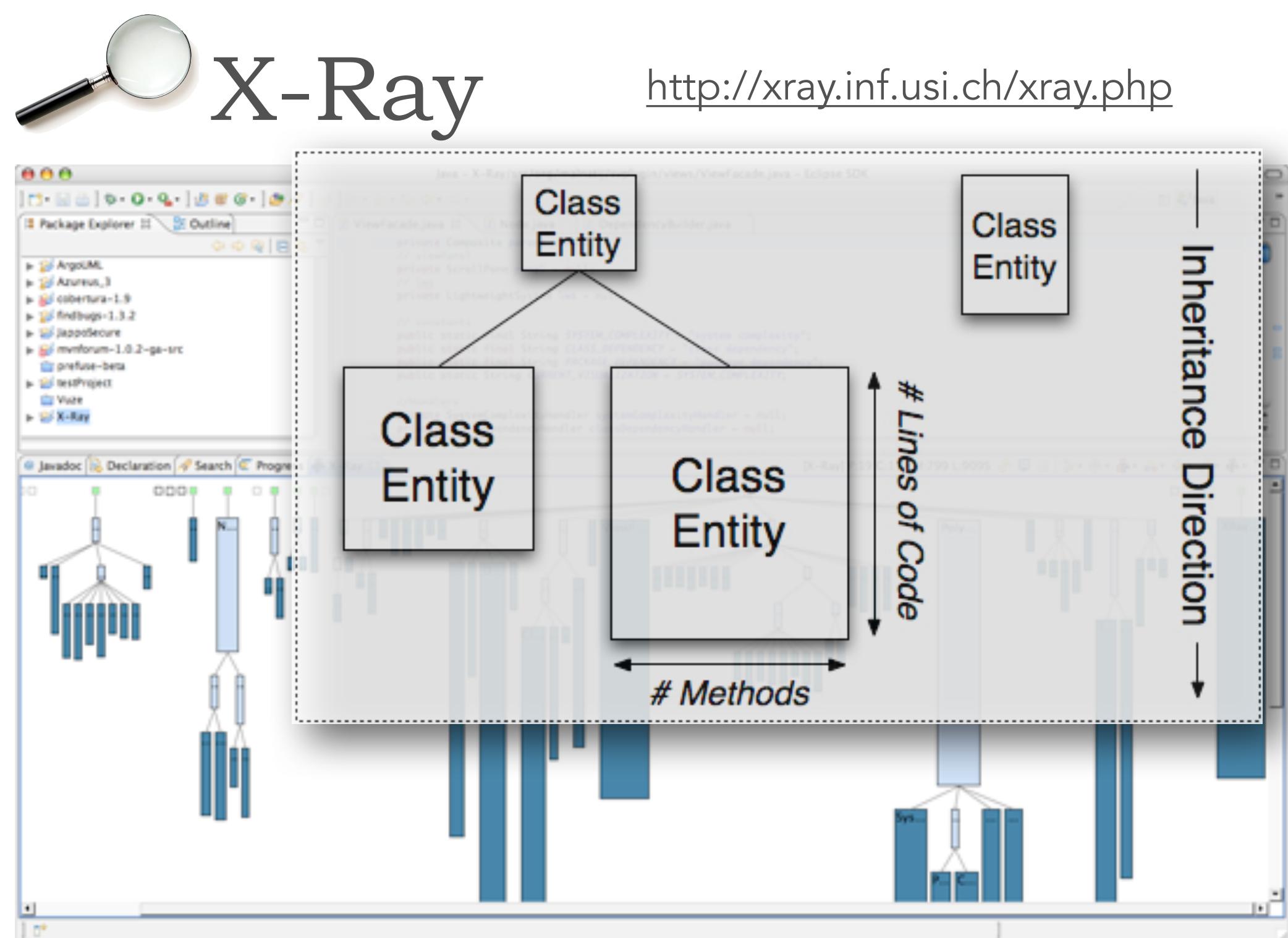
<http://www.intooitus.com>

Toxicity about

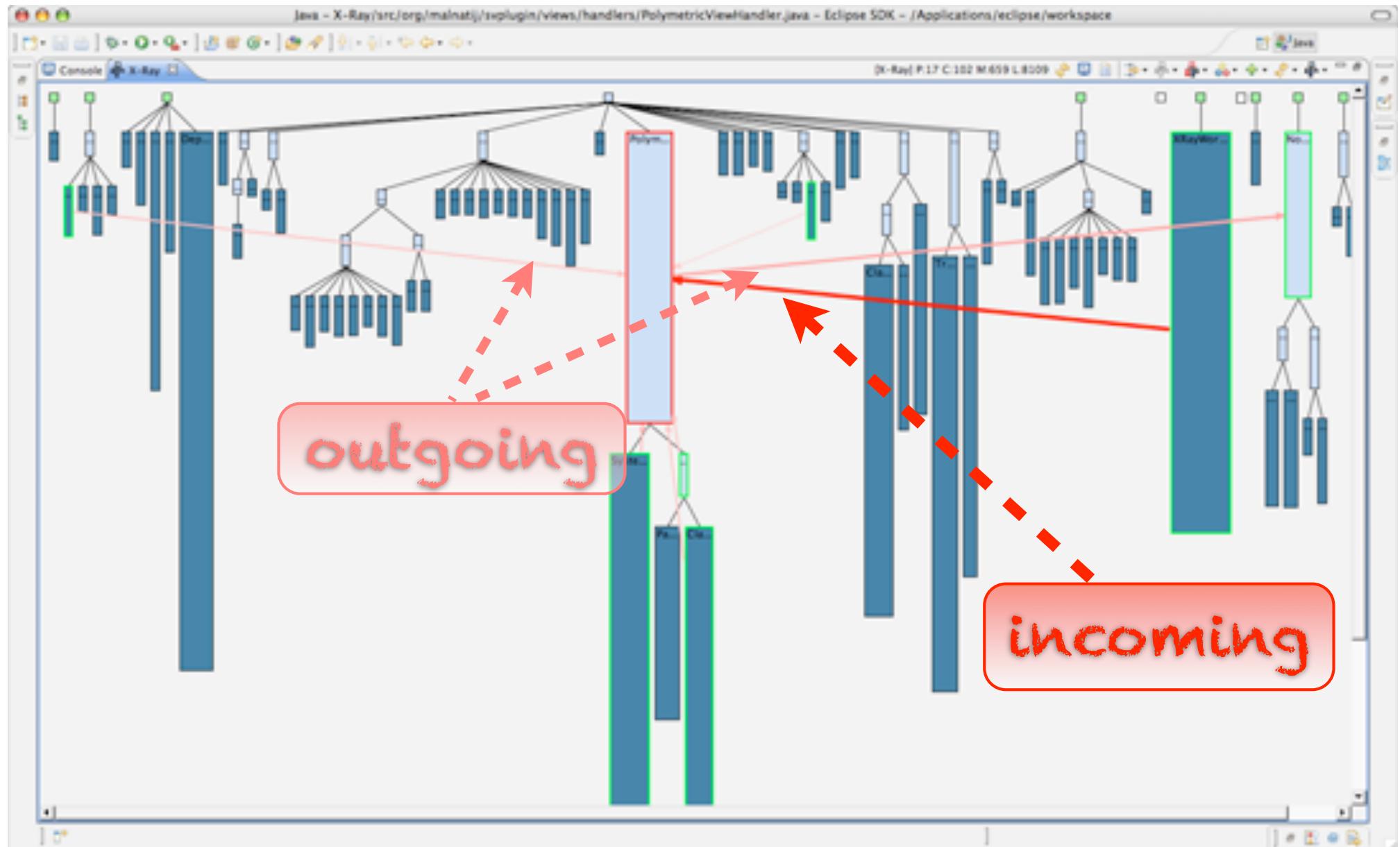


code crawler

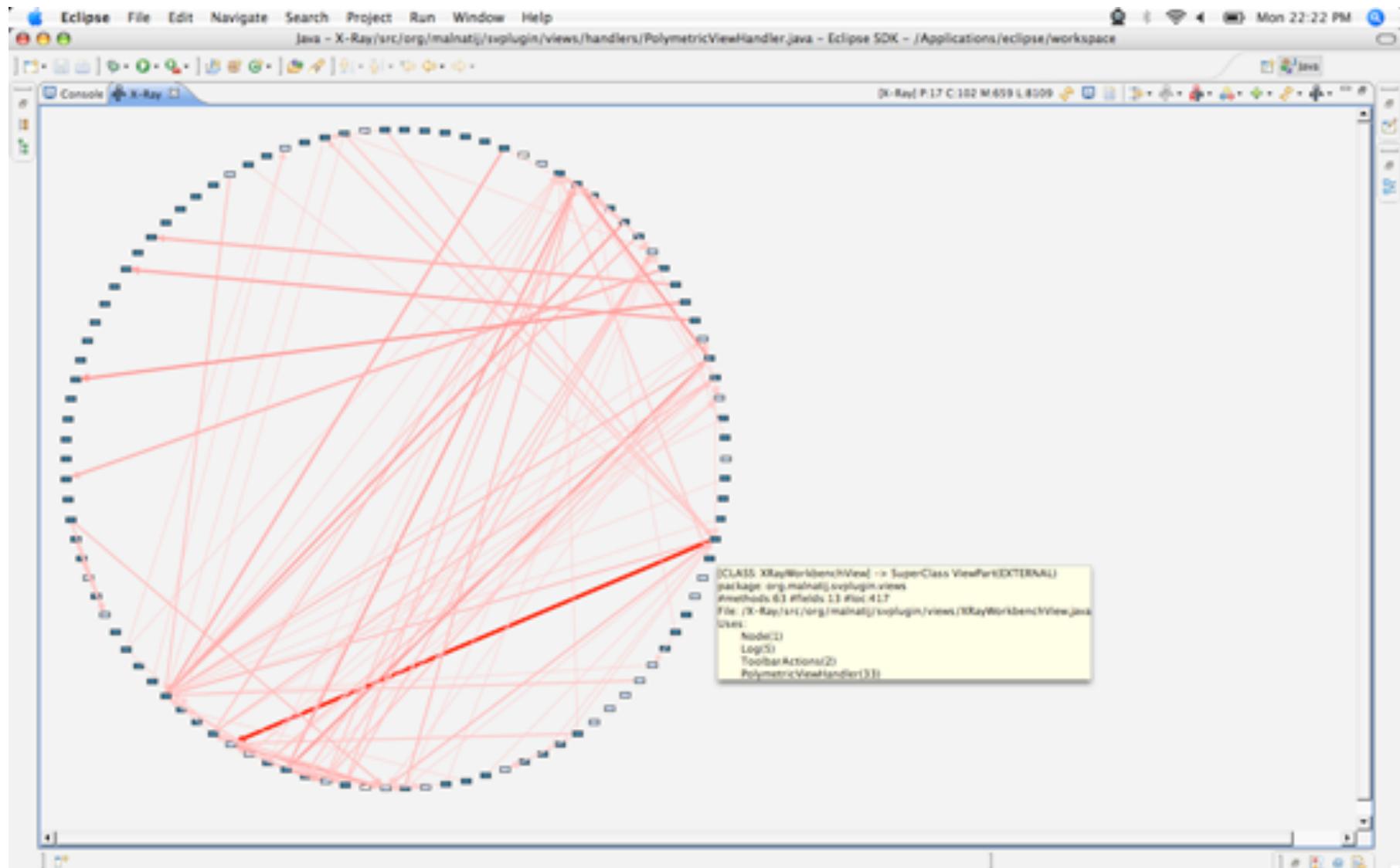




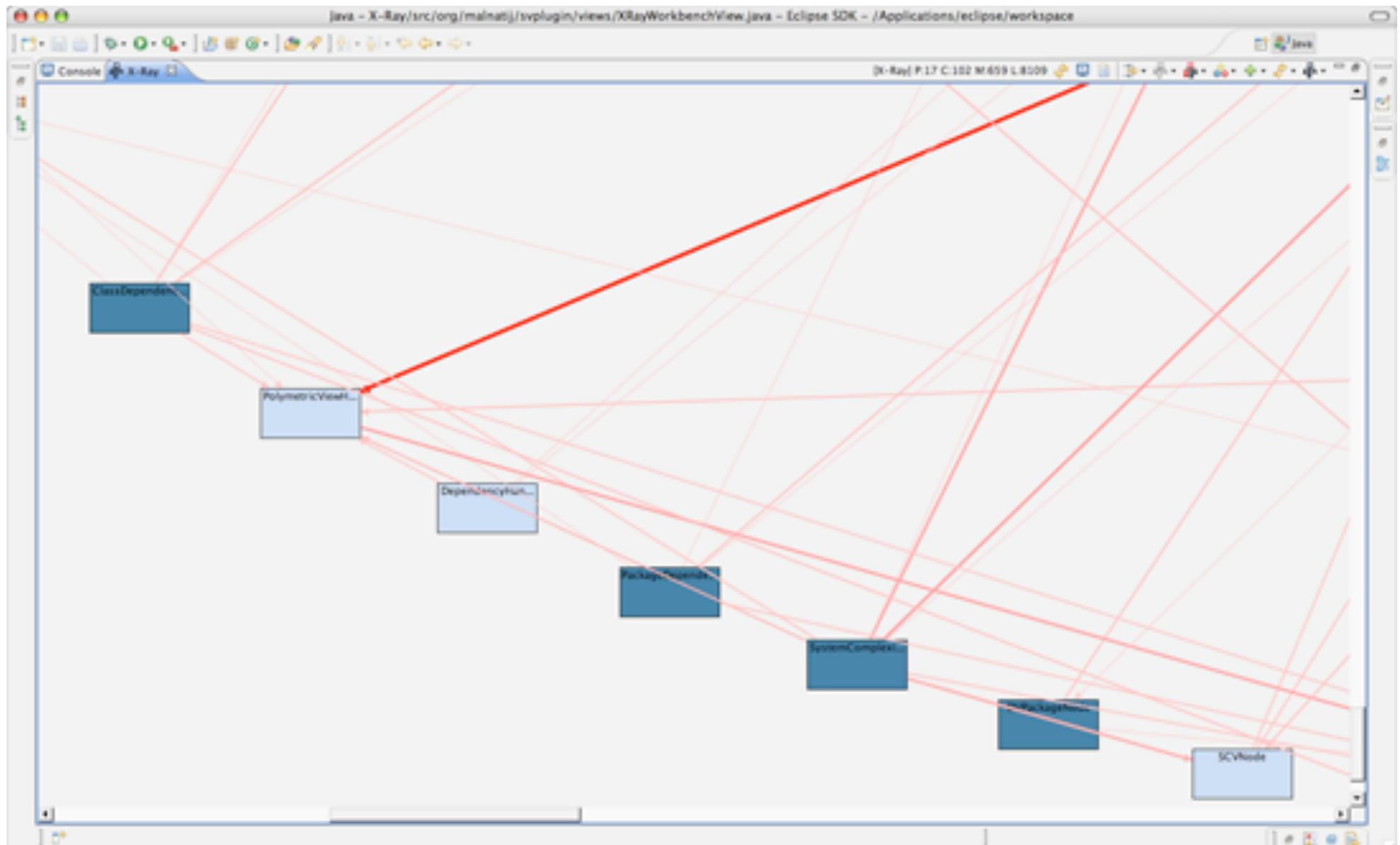
XRay dependencies



class & package dependency views



class dependency view



package dependency

Java - X-Ray/src/org/malnatij/xplugin/SVPluginActivator.java - Eclipse SDK

Package Explorer Hierarchy

SVPluginActivator.java

```
package org.malnatij.xplugin;

import org.eclipse.jface.resource.ImageDescriptor;
// ...
public class SVPluginActivator extends AbstractUIPlugin {
    // ...
}
```

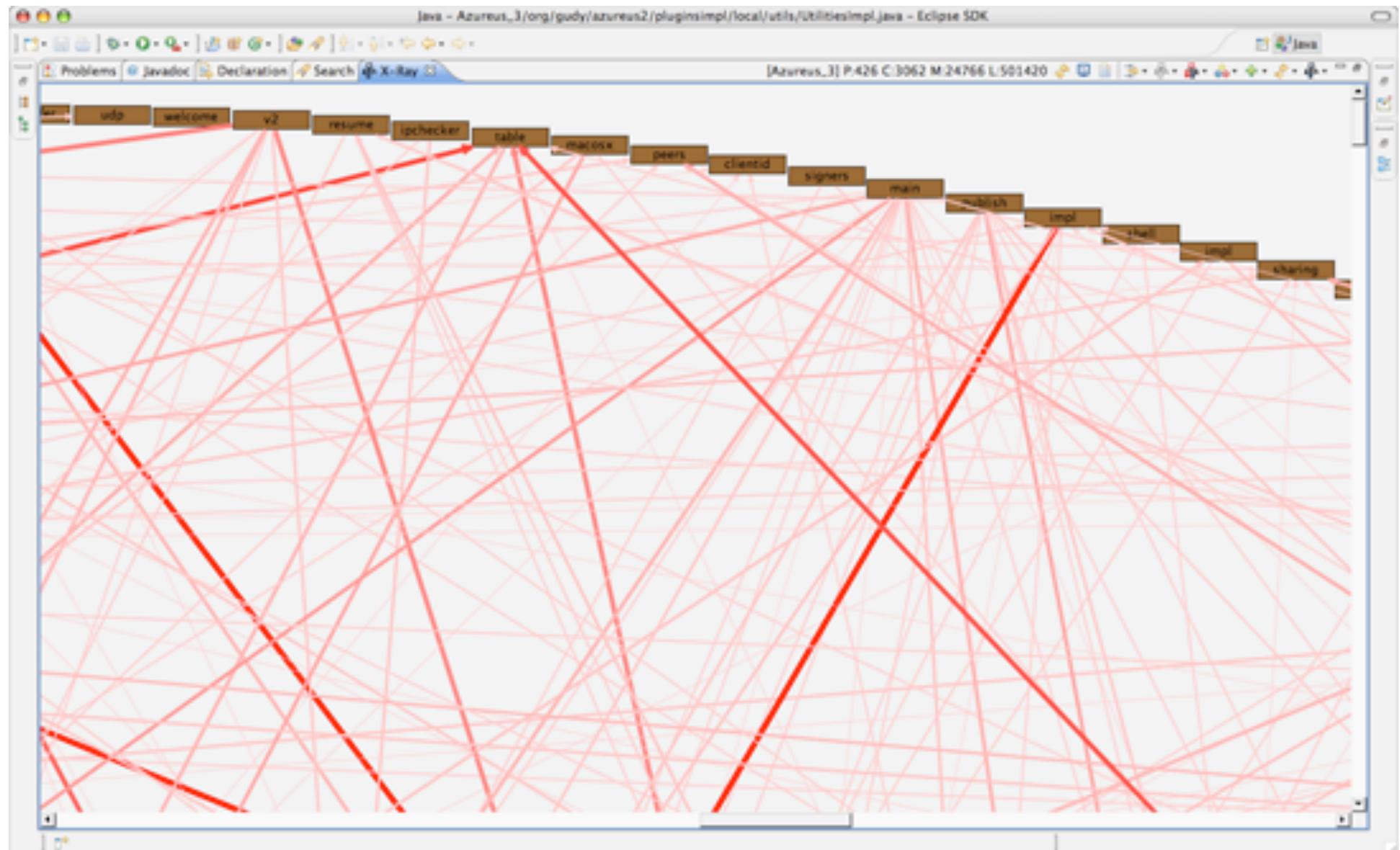
Problems Javadoc Declaration Search X-Ray [X-Ray] P:17 C:102 M:659 L:8109

Outline

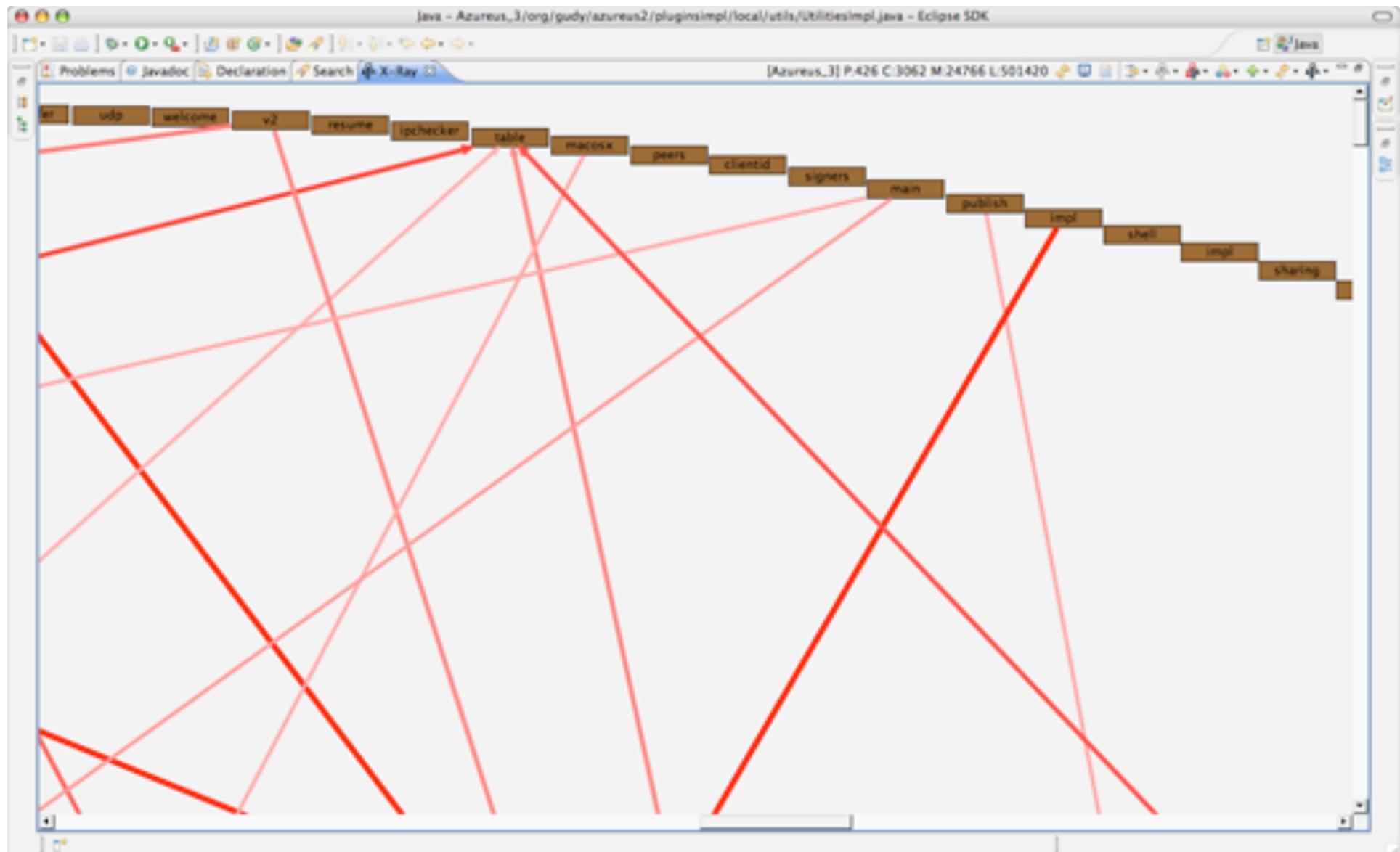
- org.malnatij.xplugin
- Import declarations
 - org.eclipse.jface.resource.ImageDescriptor
 - org.eclipse.ui.plugin.AbstractUIPlugin
 - org.malnatij.xplugin.util.Log
 - org.osgi.framework.BundleContext
- SVPluginActivator
 - PLUGIN_ID : String
 - plugin : SVPluginActivator
 - SVPluginActivator()
 - start(BundleContext)
 - stop(BundleContext)
 - getDefault()
 - getImageDescriptor(String)

org.malnatij.xplugin.SVPluginActivator.java / org.malnatij.xplugin - X-Ray / src

Azureus packages



filtering (< 30 weight)



proximity alert

The screenshot shows the Eclipse IDE interface with the X-Ray plugin installed. The central part of the screen displays the code for `ClassRepresentation.java` from the `org.malnati.jsvplugin.model` package. The code implements `UniqueIdentifiableObject` and contains annotations for interfaces and fields.

```
java - X-Ray/src/org/malnati/jsvplugin/model/ClassRepresentation.java - Eclipse SDK - /Applications/eclipse/workspace
package org.malnati.jsvplugin.model;

import java.util.ArrayList;

/*
 * This class represents a "class entity" found in the project that the
 * plugin is analyzing. It holds information about the name and type
 * of class
 * Author: malnati
 */
public class ClassRepresentation
    extends ContainedEntityRepresentation
    implements UniqueIdentifiableObject {
    // every class may implement several INTERFACES
    private ArrayList<ClassRepresentation> interfaces =
        new ArrayList<ClassRepresentation>();
    // every class may extend one class, setting that class as SUPERCLASS
    private ClassRepresentation superClass = null;
}
```

To the right of the code editor, the **Outline** and **Configuration** view are visible. The **Outline** view shows general information for the project, including:

Project	X-Ray
Packages	19 of 19
Classes	134 of 134
Interfaces	4 of 4
Fields	213 of 213
Methods	761 of 761
Dependencies	234 of 234
Lines of code	8876 of 8876

The **Configuration** view includes sections for **Class Analysis Settings**, **Package Analysis Settings**, **BoxPlot Settings**, and **Filters**.

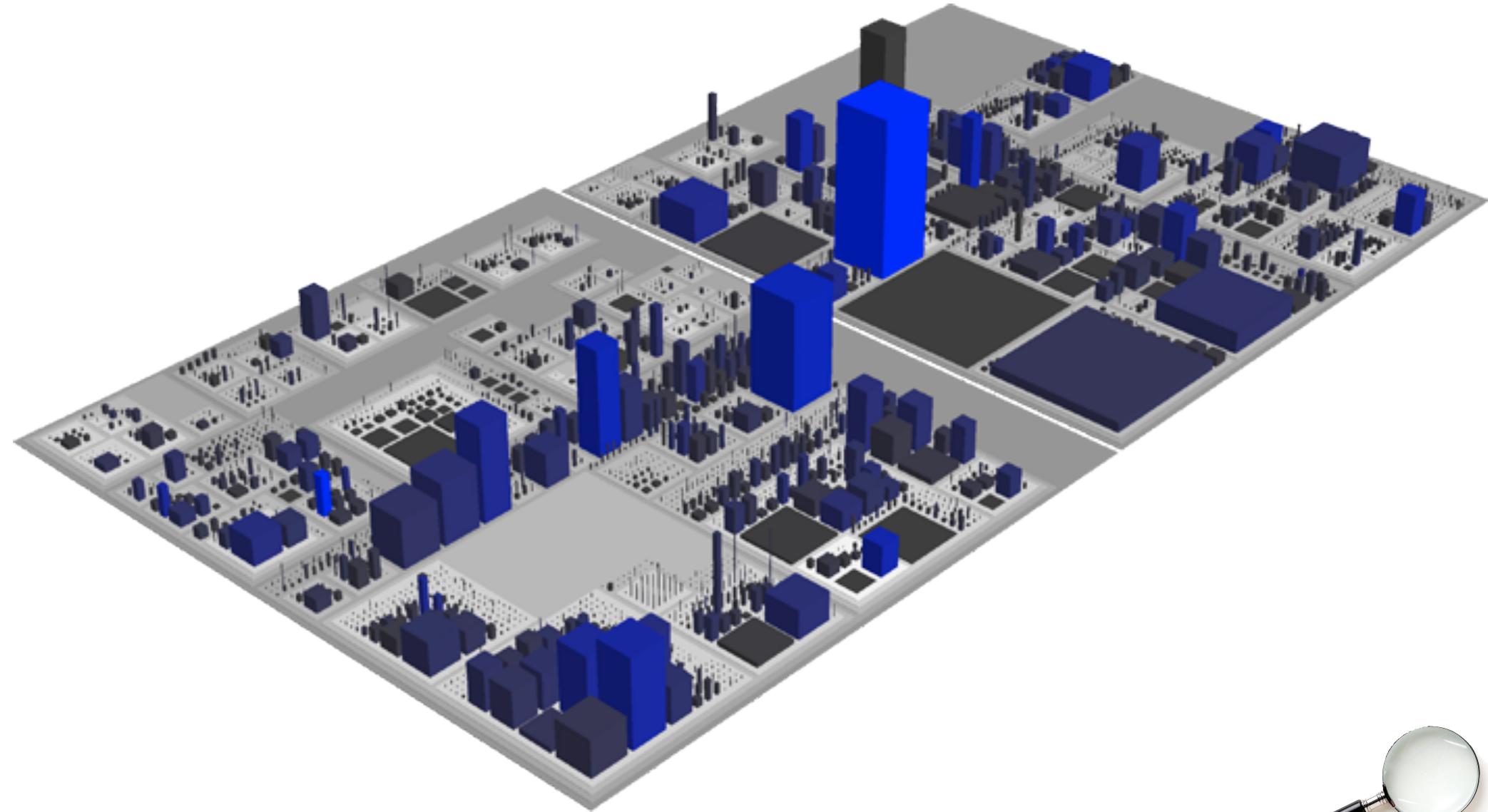
At the bottom of the interface, the **Proximity Alert** view is active, displaying a table of classes and their proximity alert scores. The table includes columns for Class Name, Package, Proximity Alert, Fields, i-Fields, Methods, i-M_ods, Const., i-Const., Used, MisIdent, Used, MisRec, and LoC.

Class Name	Package	Proximity Alert	Fields	i-Fields	Methods	i-M_ods	Const.	i-Const.	Used	MisIdent	Used	MisRec	LoC
ZoomitemMenuProvider	org.malnati.jsvplugin.views.actions.menuitems.providers	27.0	0	21	2	0	1	1	0	0	0	0	38
ZoomAction	org.malnati.jsvplugin.views.actions	21.0	0	0	2	1	1	1	0	0	0	0	41
XRayWorkbenchView	org.malnati.jsvplugin.views	299.0	6	6	61	0	1	1	57	12	18	18	337
XRay	org.malnati.jsvplugin	7.0	1	0	0	0	1	0	0	0	0	0	12
ViewFiller	org.malnati.jsvplugin.model.viewcommunication	14.0	2	1	2	0	1	1	1	1	0	0	20
ViewFacade	org.malnati.jsvplugin.views	166.0	18	1	33	0	1	1	6	12	1	22	246
ViewActionDelegate	org.malnati.jsvplugin.actions	23.0	1	1	0	0	1	1	1	3	0	0	38
UpAnchor	org.malnati.jsvplugin.graph.links	11.0	0	0	2	1	1	1	0	0	0	0	18
UntagMenuItemProvider	org.malnati.jsvplugin.views.actions.menuitems.providers	17.0	0	2	2	2	1	1	0	0	0	0	27
UniqueIdentifiableObject	org.malnati.jsvplugin.model	5.0	0	0	1	0	1	0	0	0	0	0	8
TreeLayout	org.malnati.jsvplugin.layouts	268.0	5	7	27	6	1	1	5	21	1	10	437
ToolbarActions	org.malnati.jsvplugin.views.actions	42.0	3	1	5	0	1	1	1	2	3	0	66
TicksPredictor	org.malnati.jsvplugin.model.core	31.0	1	0	3	0	1	1	0	0	1	1	62
SystemHotspotHandler	org.malnati.jsvplugin.views.handlers	381.0	0	9	23	49	1	1	0	0	0	0	143
SystemComplexityHandler	org.malnati.jsvplugin.views.handlers	337.0	4	9	42	49	1	1	8	31	1	1	547
StringValidator	org.malnati.jsvplugin.views.actions	13.0	0	1	0	0	1	1	0	0	0	0	22
SnapshotWarning	org.malnati.jsvplugin.views.actions.dialogs	18.0	0	0	1	0	1	1	1	1	0	0	31
SnapshotAction	org.malnati.jsvplugin.views.actions	53.0	1	1	7	1	1	1	2	3	0	0	91
SaveArrowLink	org.malnati.jsvplugin.graph.links	17.0	0	6	2	5	1	1	2	0	0	0	21

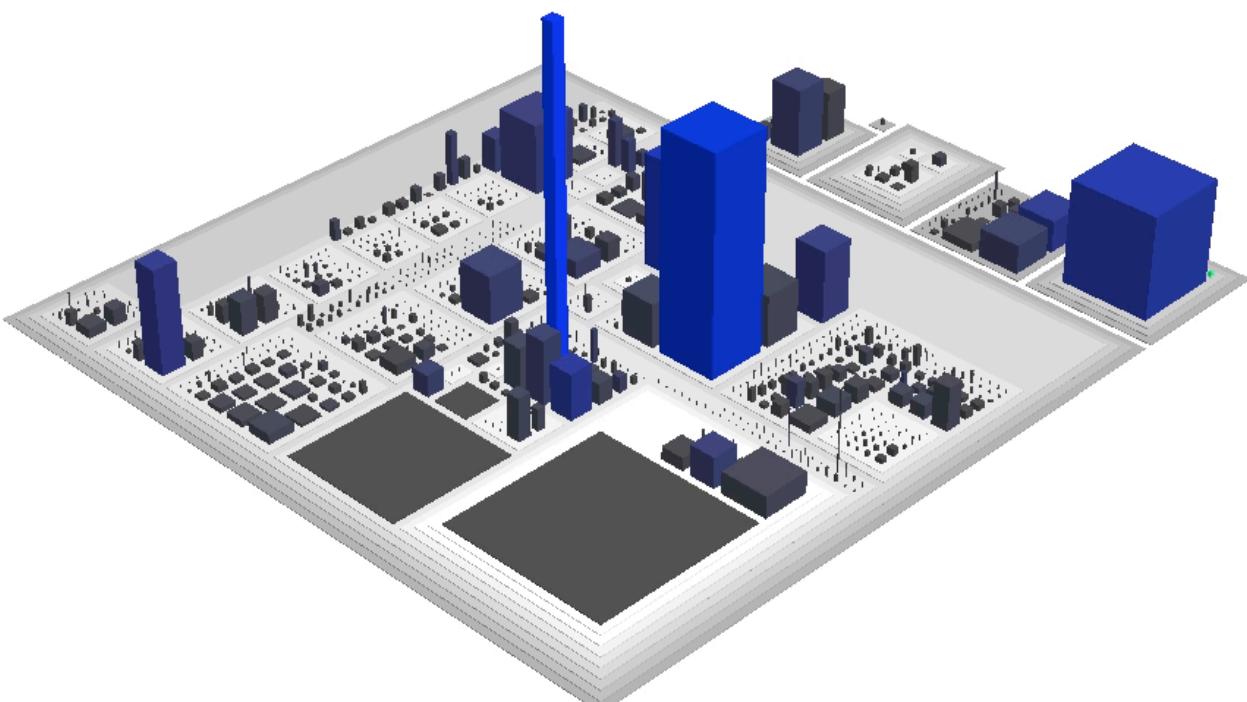
Code City



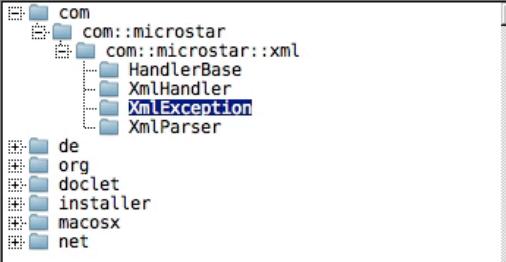
<http://www.inf.usi.ch/phd/wettel/codecity.html>



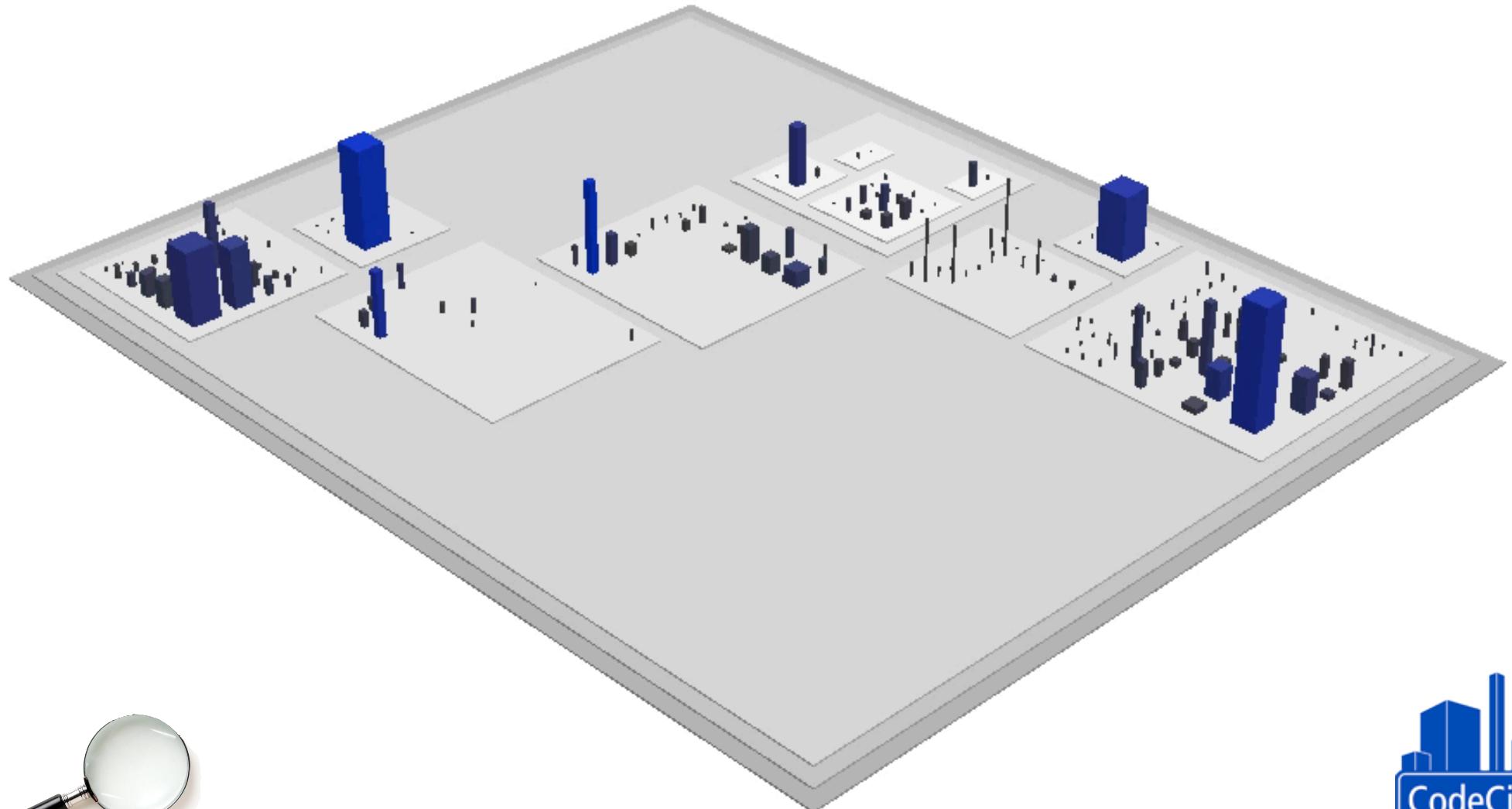
Select... On selection... Help



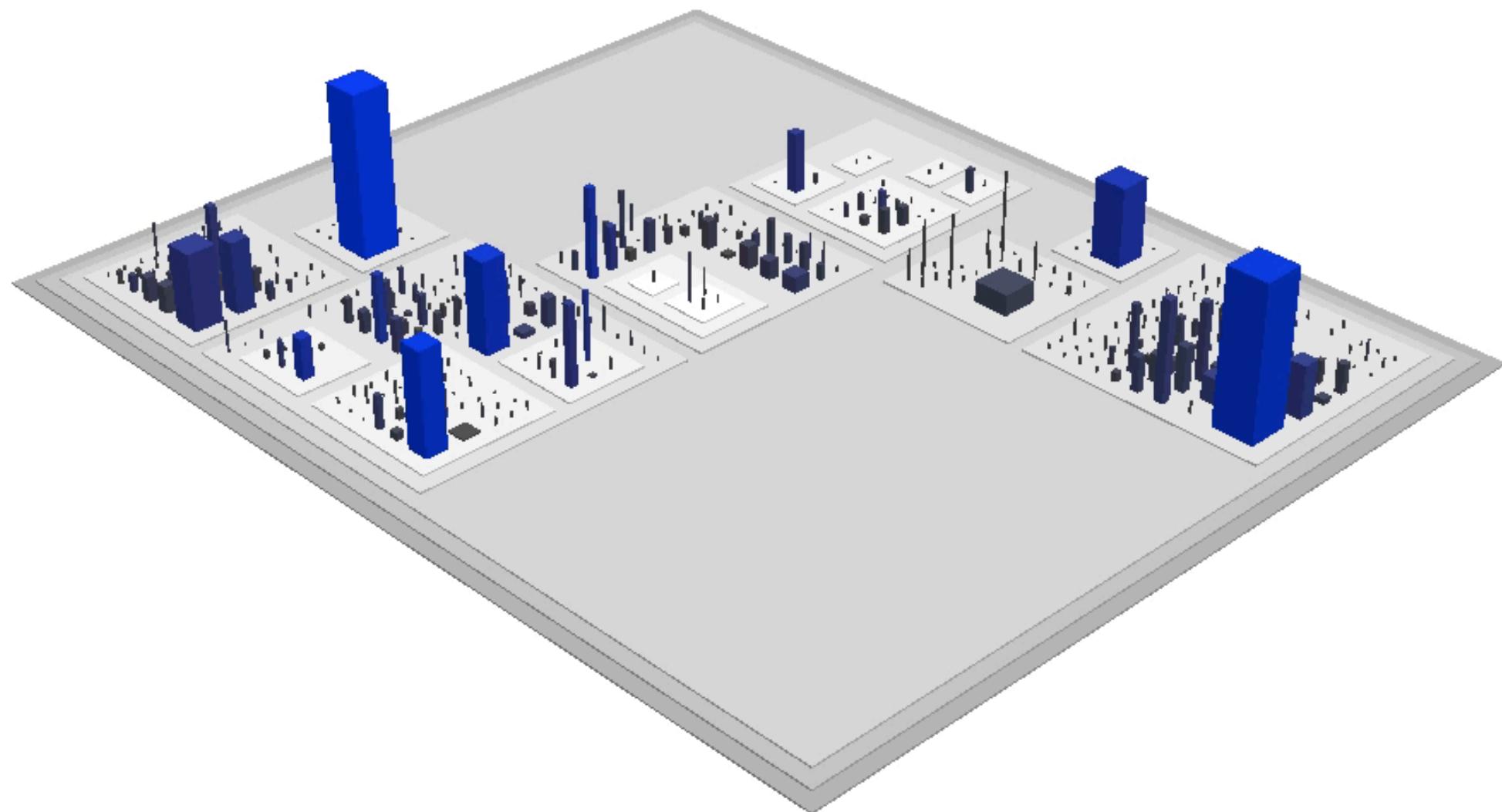
Selected: class XmlException



time sequence



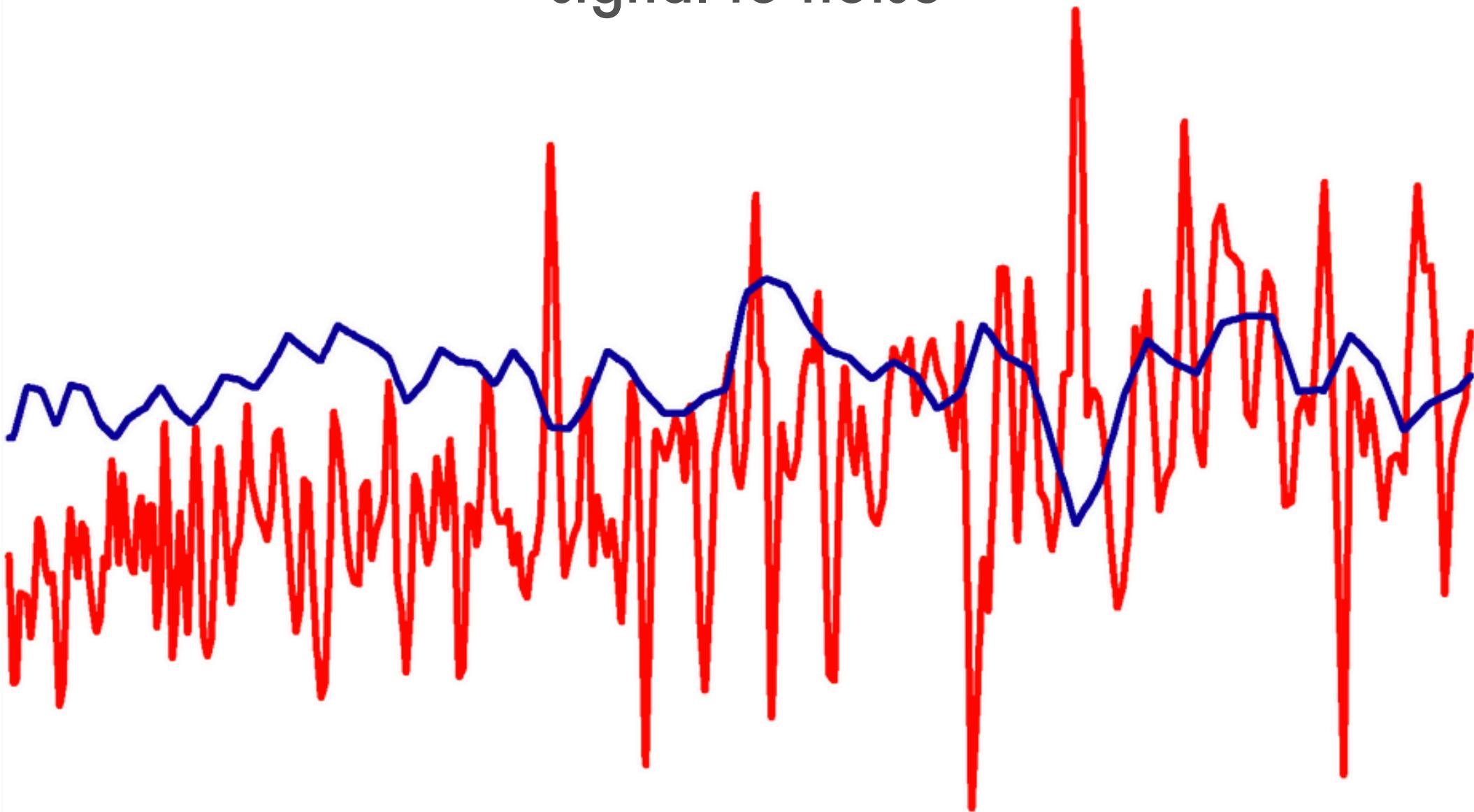
Select... On selection... Evolution Select classes where Help



limitations of metrics

signal to noise

signal to noise



limitations of metrics

signal to noise

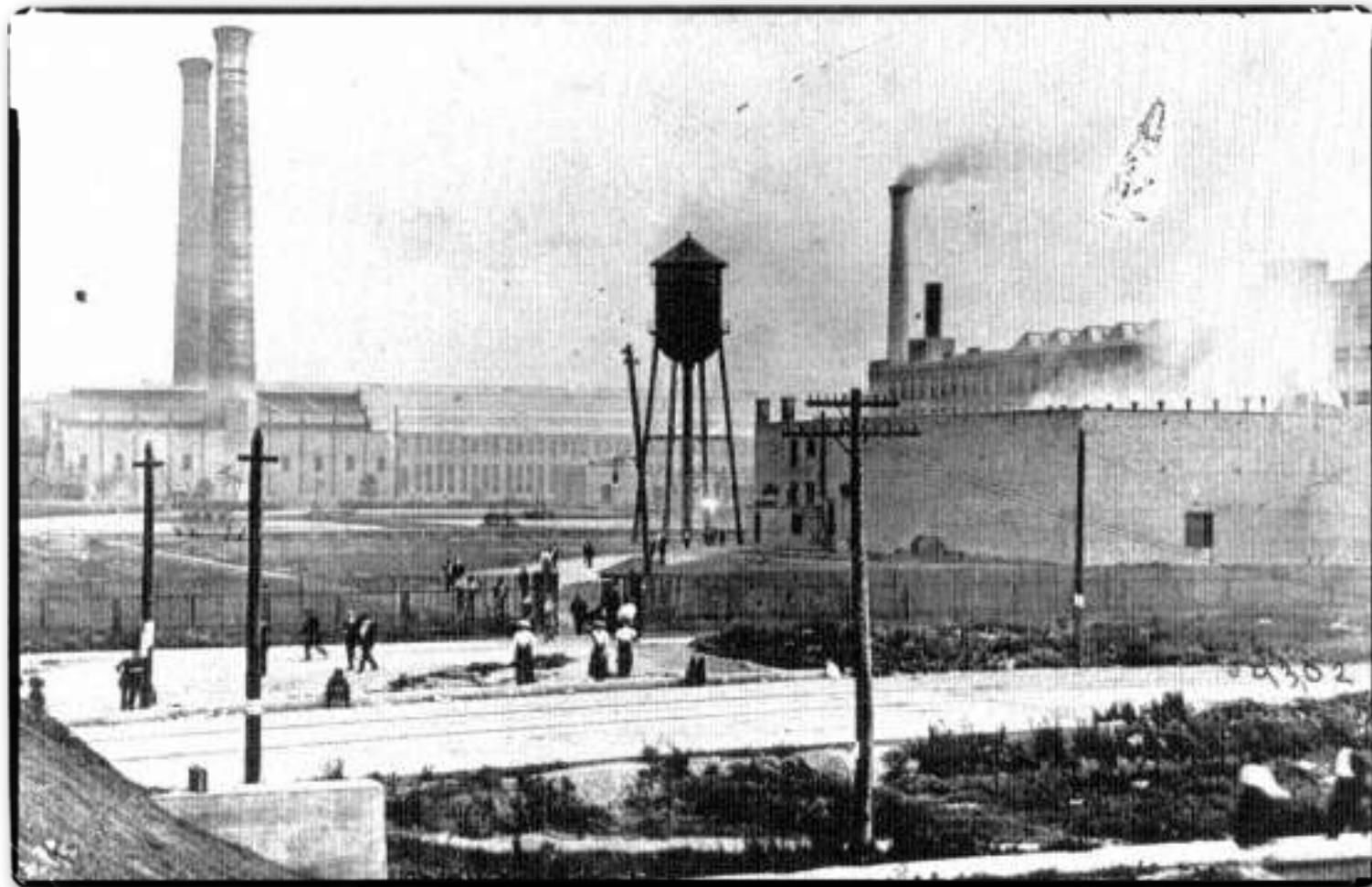
no “1 True Metric”

gathered but ignored

inaction / overaction

prefer trends to discrete values

the Hawthorne Effect



measure & let it be known that you are measuring

limitations of metrics

signal to noise

no “1 True Metric”

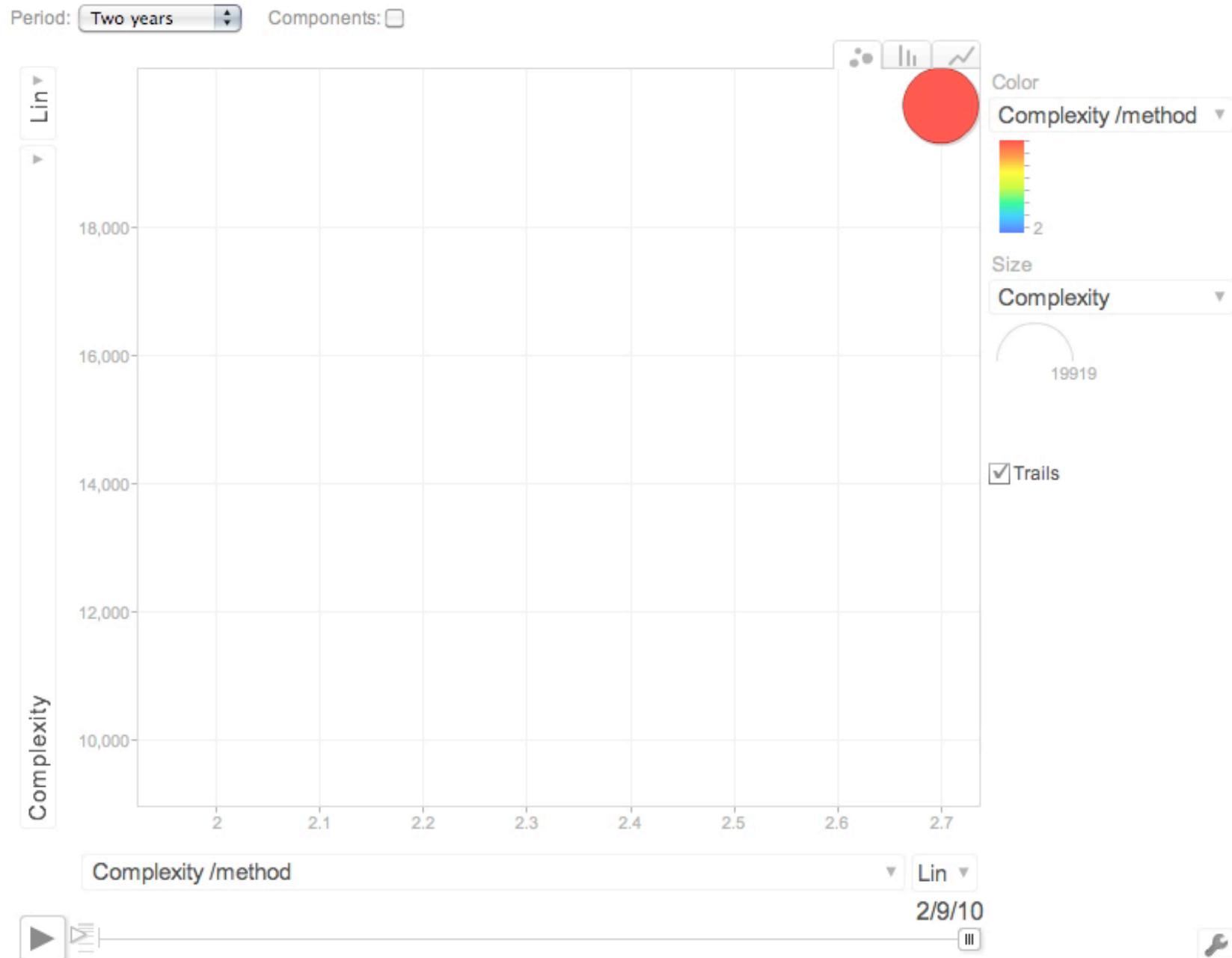
gathered but ignored

inaction / overaction

prefer trends to discrete values

Sonar trends

<http://www.sonarsource.org/>



2 typical uses for metrics



radiators



probes



<https://github.com/thehammer/jukebox.rb>

? ' S



Mark Richards

Independent Consultant

Hands-on Enterprise / Integration Architect

Published Author / Conference Speaker

<http://www.wmrichards.com>

<http://www.linkedin.com/pub/mark-richards/0/121/5b9>

Published Books:

Java Message Service, 2nd Edition

97 Things Every Software Architect Should Know

Java Transaction Design Strategies



Neal Ford

Director / Software Architect /

Meme Wrangler

ThoughtWorks®

2002 Summit Blvd, Level 3, Atlanta, GA 30319, USA

T: +1 404 242 9929 Twitter: @neal4d

E: nford@thoughtworks.com W: thoughtworks.com