Clone Detection & Management für Architekten

CQSE | Dr. Elmar Jürgens

http://de.wikipedia.org/w/index.php?title=Datei:Fred_Brooks.jpg

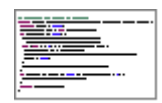
#1 Code Smell (1999)

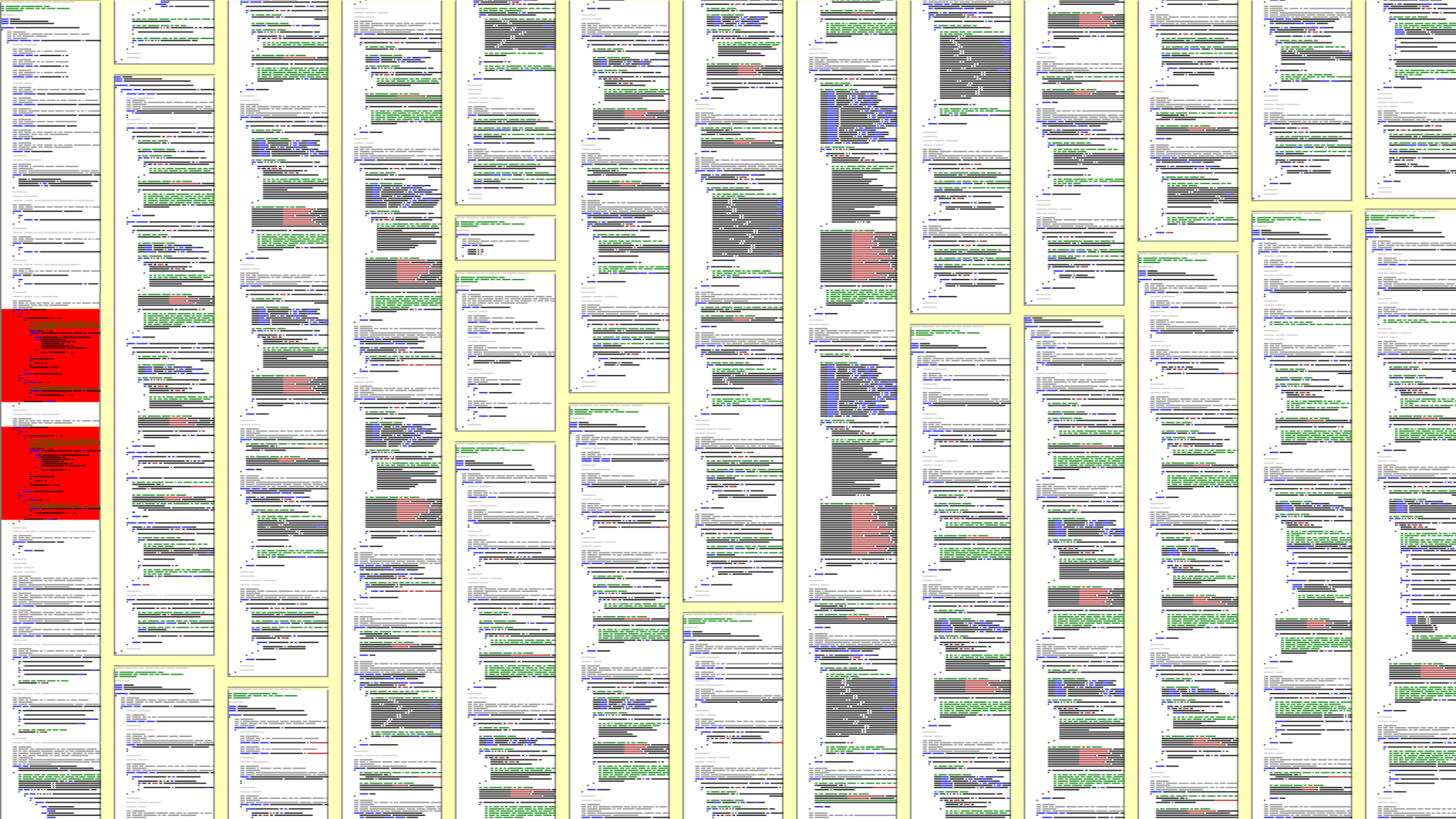
http://www.flickr.com/photos/pragdave/173640462/

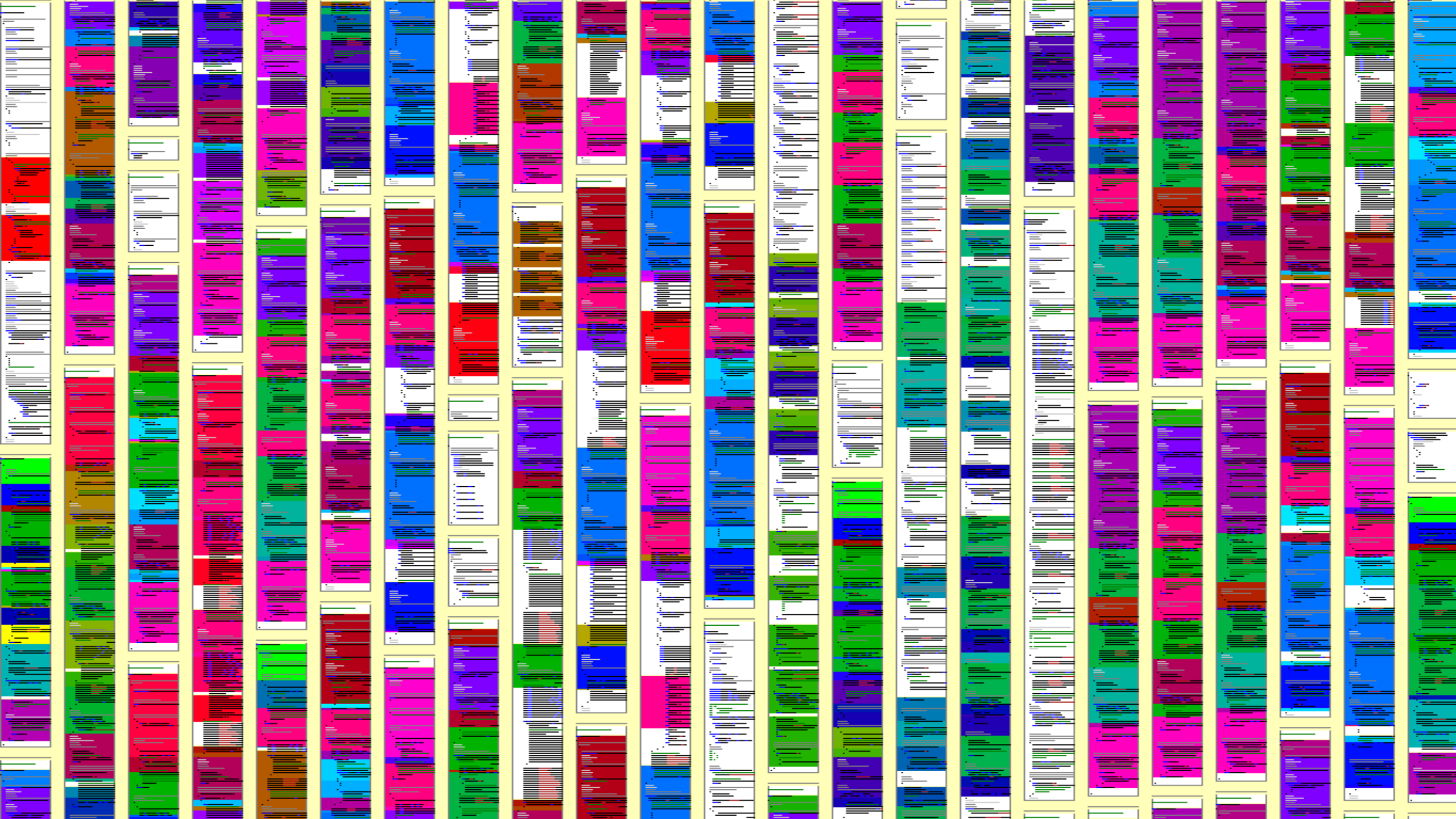
```java
// Utilities for arrays of elements
public String showElements(ModelElement[] elements, String nomsg) {
   boolean found = false;
   StringBuffer res = new StringBuffer();
   if (elements != null) {
      Index.getInstance().setCurrentRenderer(
            FlatReferenceRenderer.getInstance());
      for (int i = 0; i < elements.length; i++) {
         ModelElement el = elements[i];
         res.append(showElementLink(el)).append(HTML.LINE_BREAK);
         found = true;
      }
      Index.getInstance().resetCurrentRenderer();
   }
   if (!found && nomsg != null && nomsg.length() > 0) {
      res.append(HTML.italics(nomsg));
   }
   return res.toString();
}
```

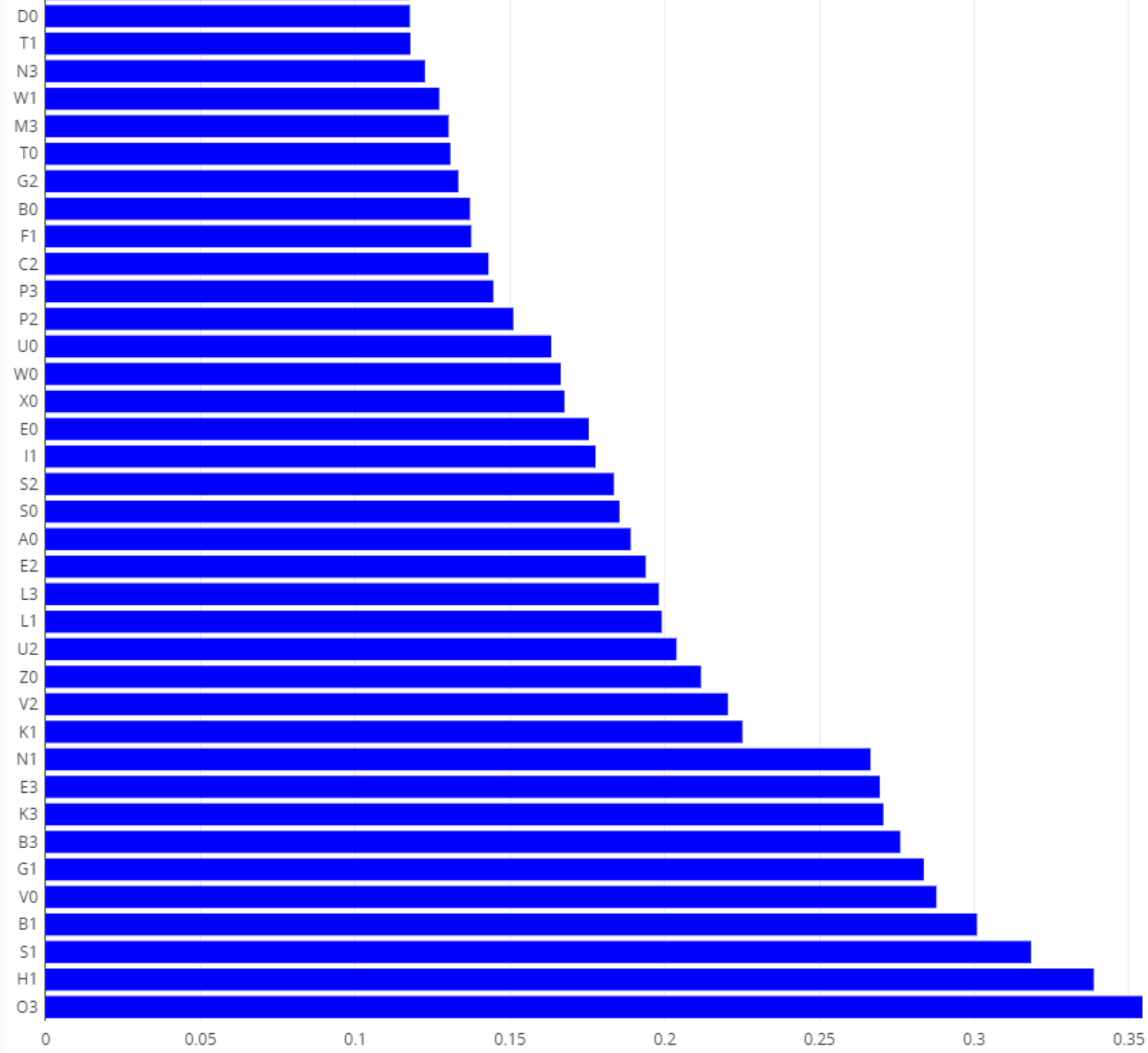
```java
// Utilities for arrays of elements
public String showElements(ModelElement[] elements, String nomsg) {
   boolean found = false;
   StringBuffer res = new StringBuffer();
   if (elements != null) {
      Index.getInstance().setCurrentRenderer(
            FlatReferenceRenderer.getInstance());
      for (int i = 0; i < elements.length; i++) {
         ModelElement el = elements[i];
         res.append(showElementLink(el)).append(HTML.LINE_BREAK);
         found = true;
      }
      Index.getInstance().resetCurrentRenderer();
   }
   if (!found && nomsg.length() > 0) {
      res.append(HTML.italics(nomsg));
   }
   return res.toString();
}
```

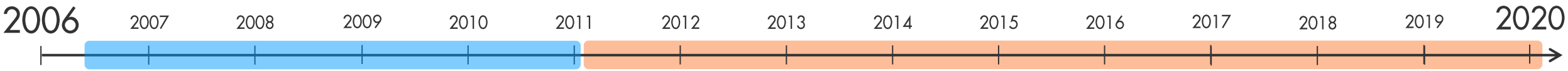
```java
// Utilities for arrays of elements
public String showElements(ModelElement[] elements, String nomsg) {
    boolean found = false;
    StringBuffer res = new StringBuffer();
    if (elements != null) {
        Index.getInstance().setCurrentRenderer(
                FlatReferenceRenderer.getInstance());
        for (int i = 0; i < elements.length; i++) {
            ModelElement el = elements[i];
            res.append(showElementLink(el)).append(HTML.LINE_BREAK);
            found = true;
        }
        Index.getInstance().resetCurrentRenderer();
    }
    if (!found && nomsg != null && nomsg.length() > 0) {
        res.append(HTML.italics(nomsg));
    }
    return res.toString();
}
```



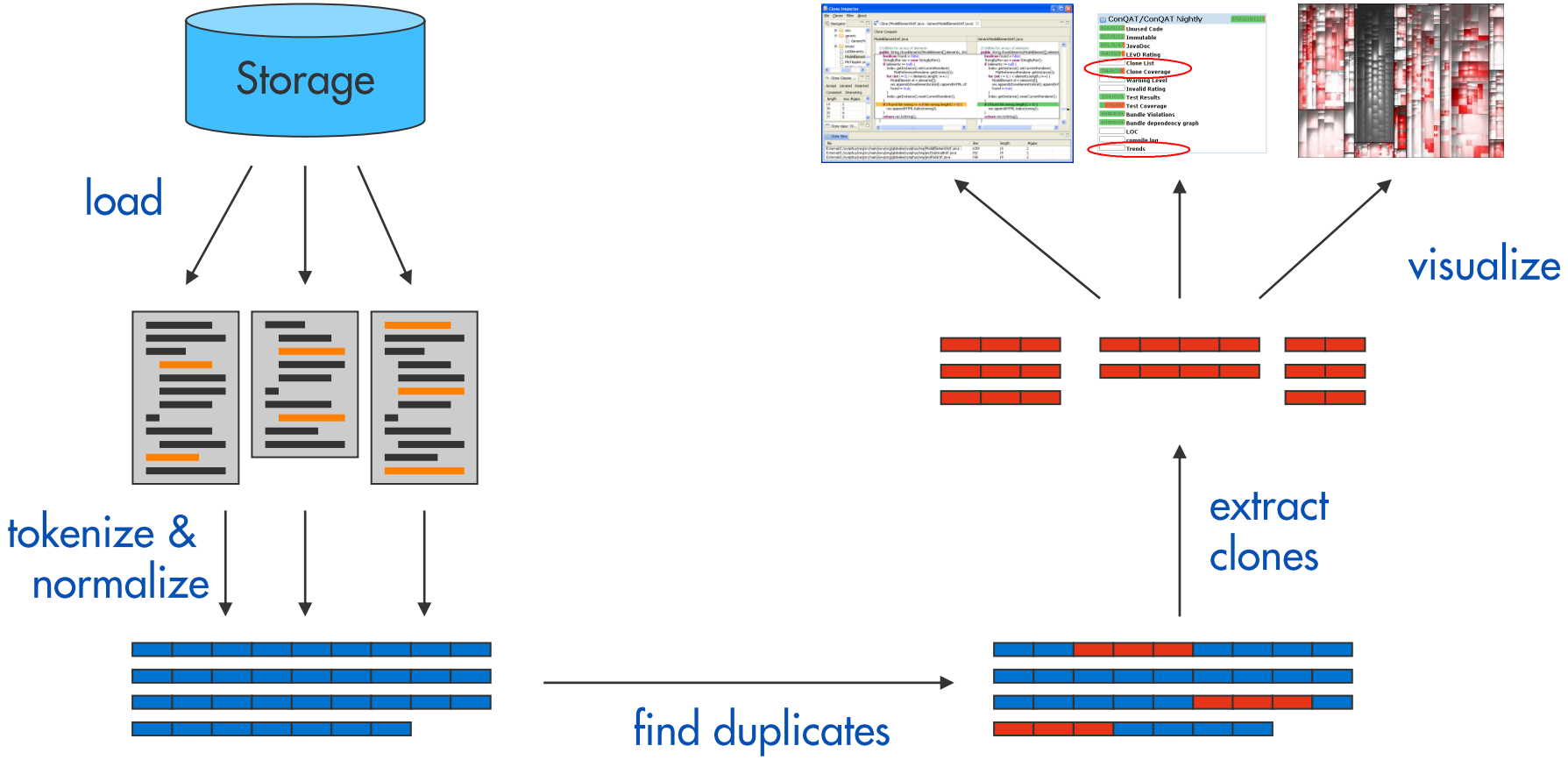


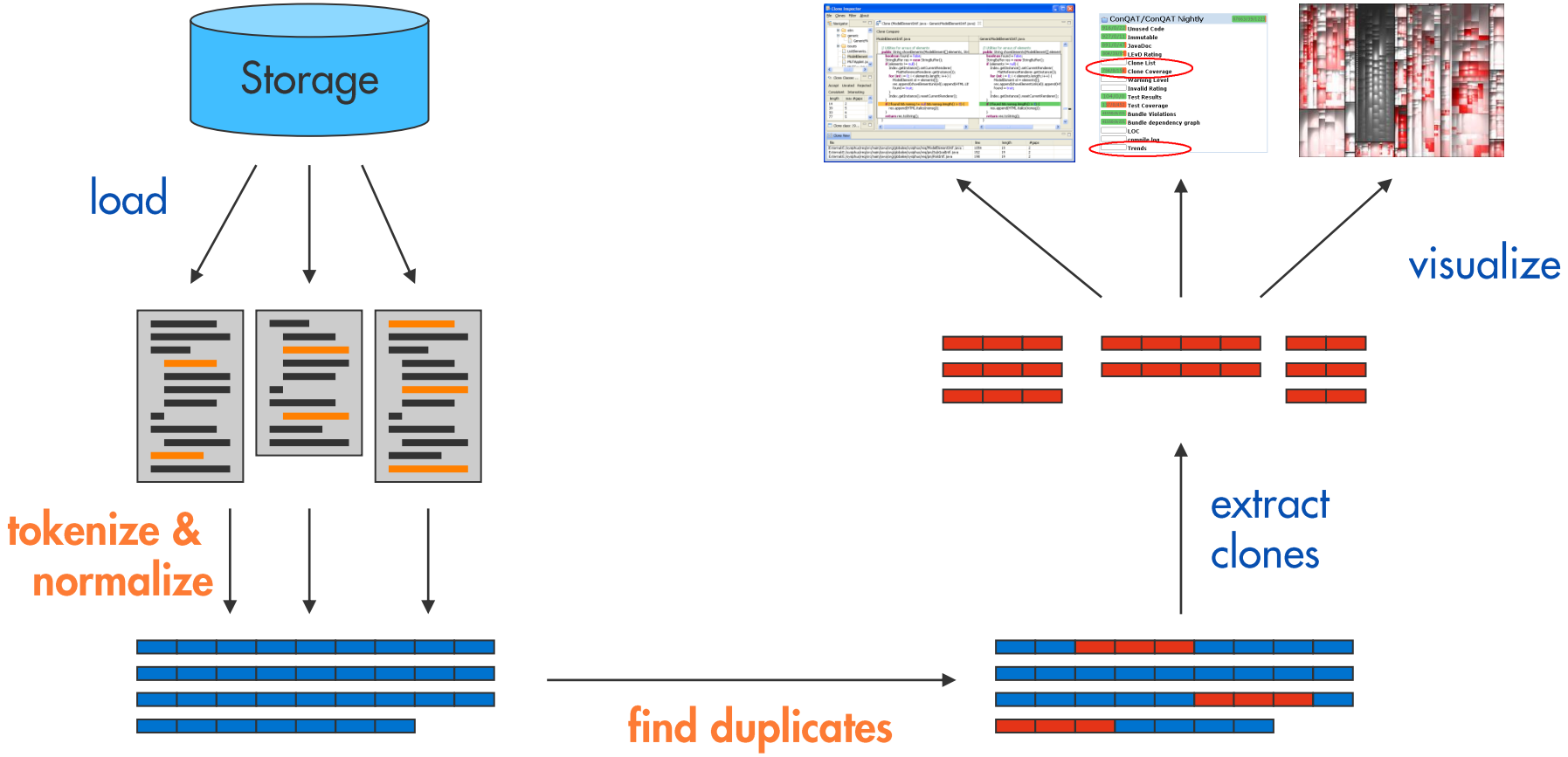



D0
T1
N3
W1
M3
T0
G2
B0
F1
C2
P3
P2
U0
W0
X0
E0
I1
S2
S0
A0
E2
L3
L1
U2
Z0
V2
K1
N1
E3
K3
B3
G1
V0
B1
S1
H1
O3
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35

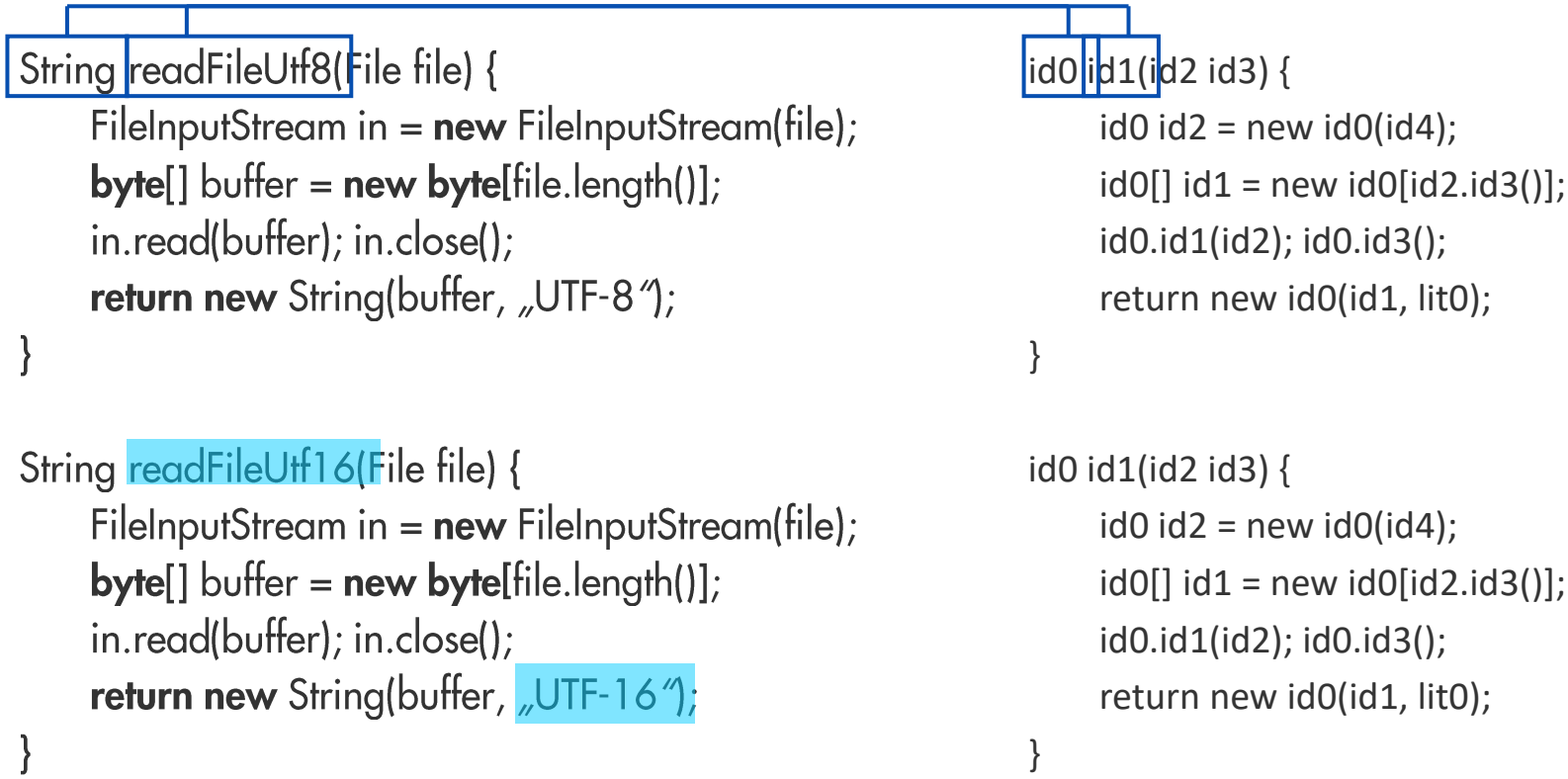

2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020

Wie funktioniert Clone Detection?


Storage
load
tokenize &
normalize
find duplicates
extract
clones
visualize
ConQAT/ConQAT Nightly
Unused Code
Immutable
JavaDoc
LEvD Rating
Clone List
Clone Coverage
Warning Level
Invalid Rating
Test Results
Test Coverage
Bundle Violations
Bundle dependency graph
LOC
Trends


Storage
load
tokenize &
normalize
find duplicates
extract
clones
visualize
ConQAT/ConQAT Nightly
Unused Code
Immutable
JavaDoc
LEvD Rating
Clone List
Clone Coverage
Warning Level
Invalid Rating
Test Results
Test Coverage
Bundle Violations
Bundle dependency graph
LOC
Compile log
Trends

Normalisierung



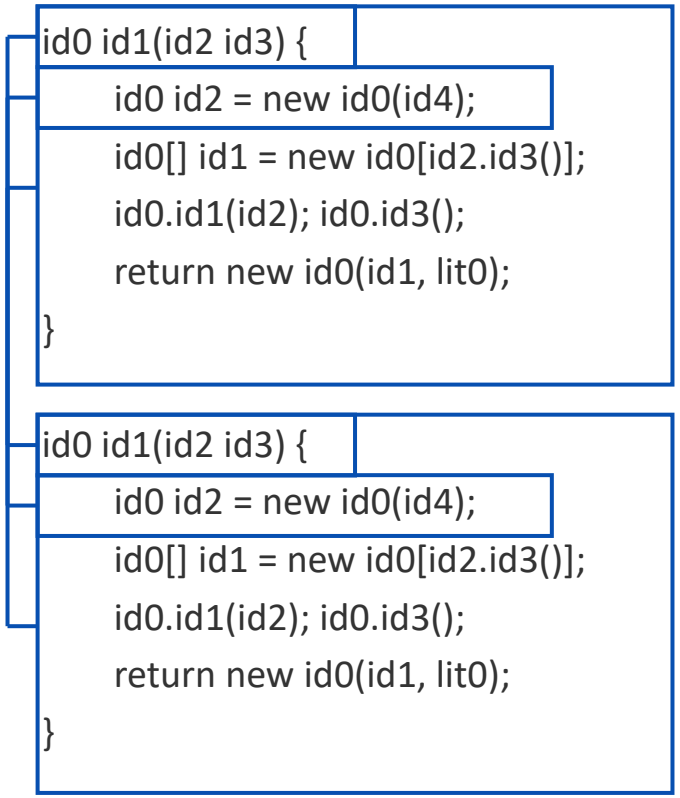
```
String readFileUtf8(File file) {
    FileInputStream in = new FileInputStream(file);
    byte[] buffer = new byte[file.length()];
    in.read(buffer); in.close();
    return new String(buffer, „UTF-8“);
}

String readFileUtf16(File file) {
    FileInputStream in = new FileInputStream(file);
    byte[] buffer = new byte[file.length()];
    in.read(buffer); in.close();
    return new String(buffer, „UTF-16“);
}
```

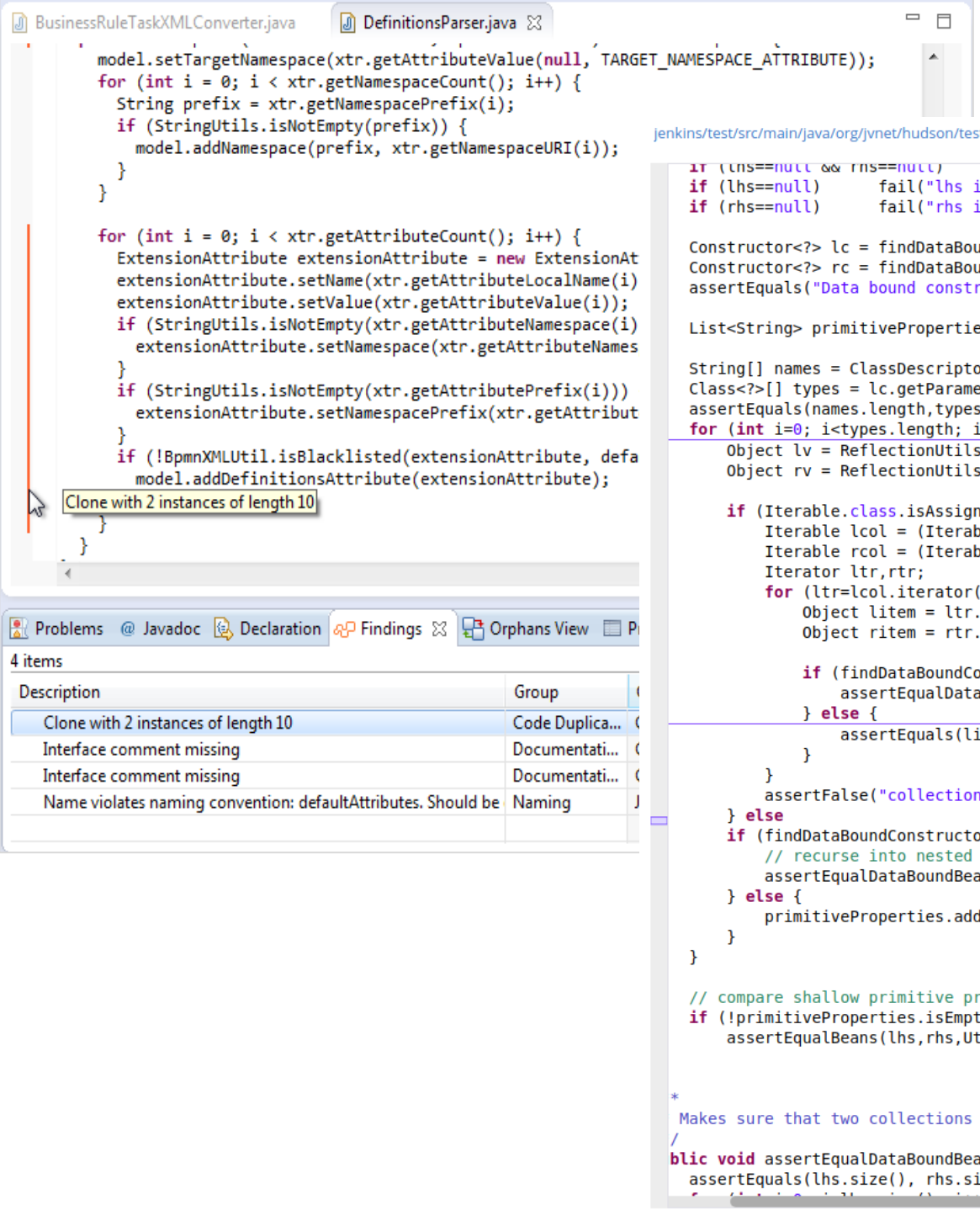
```
id0 id1(id2 id3) {
    id0 id2 = new id0(id4);
    id0[] id1 = new id0[id2.id3()];
    id0.id1(id2); id0.id3();
    return new id0(id1, lit0);
}

id0 id1(id2 id3) {
    id0 id2 = new id0(id4);
    id0[] id1 = new id0[id2.id3()];
    id0.id1(id2); id0.id3();
    return new id0(id1, lit0);
}
```

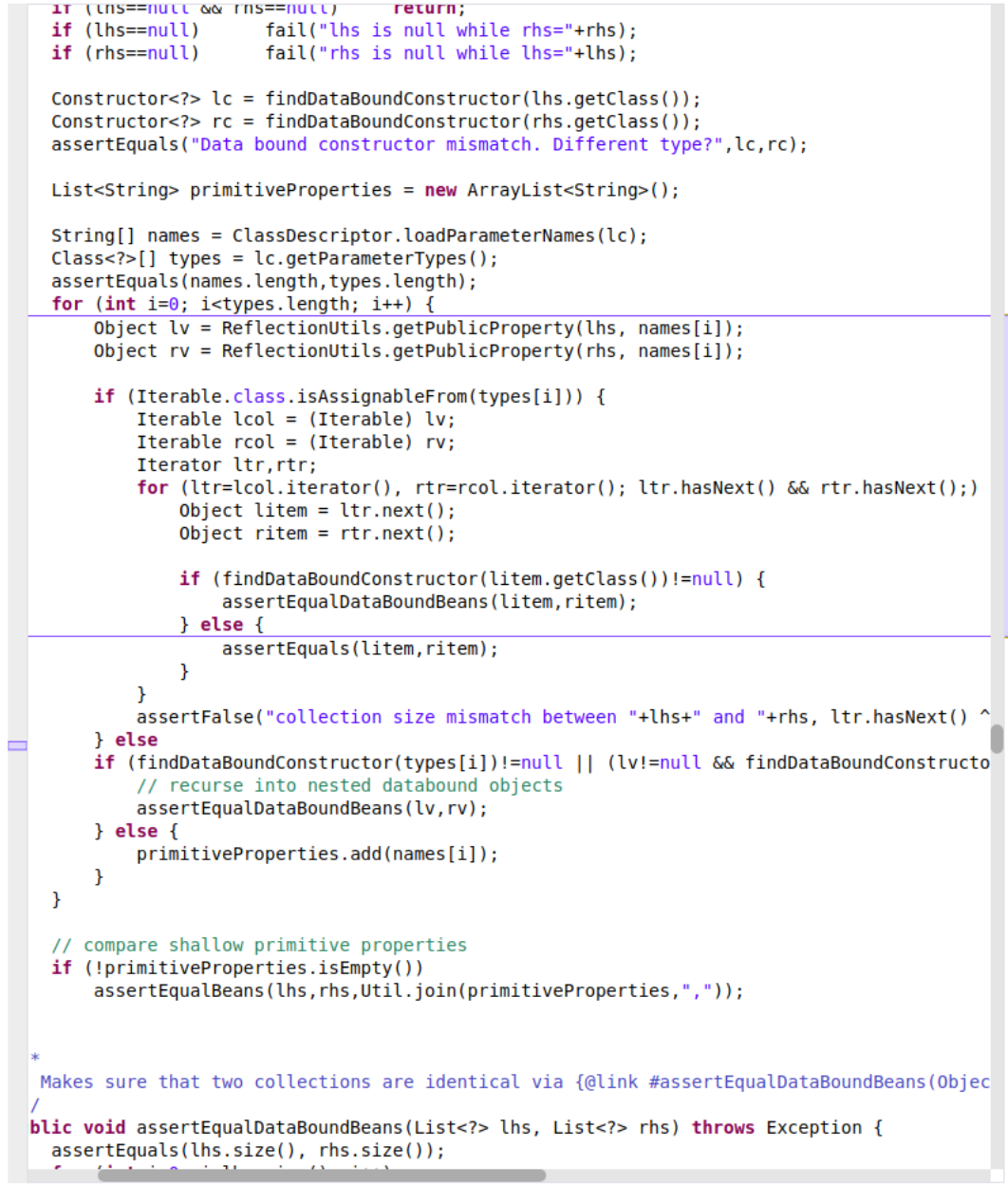
Normalisierung

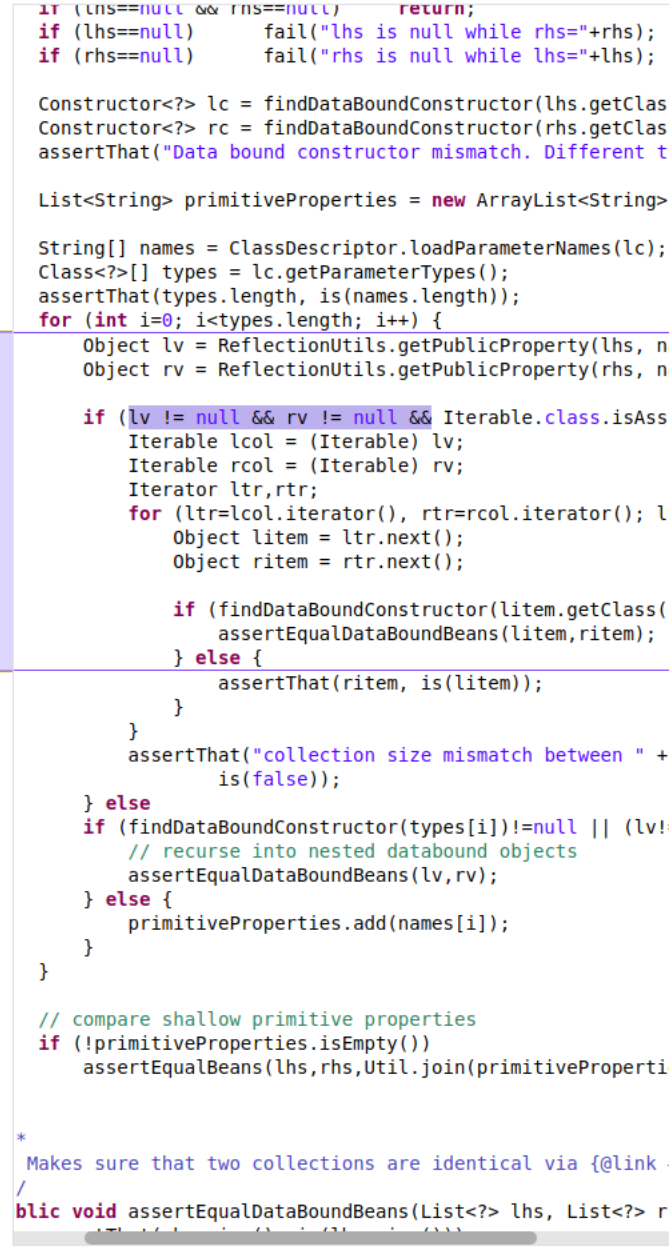
```
String readFileUtf8(File file) {
    FileInputStream in = new FileInputStream(file);
    byte[] buffer = new byte[file.length()];
    in.read(buffer); in.close();
    return new String(buffer, „UTF-8“);
}
```

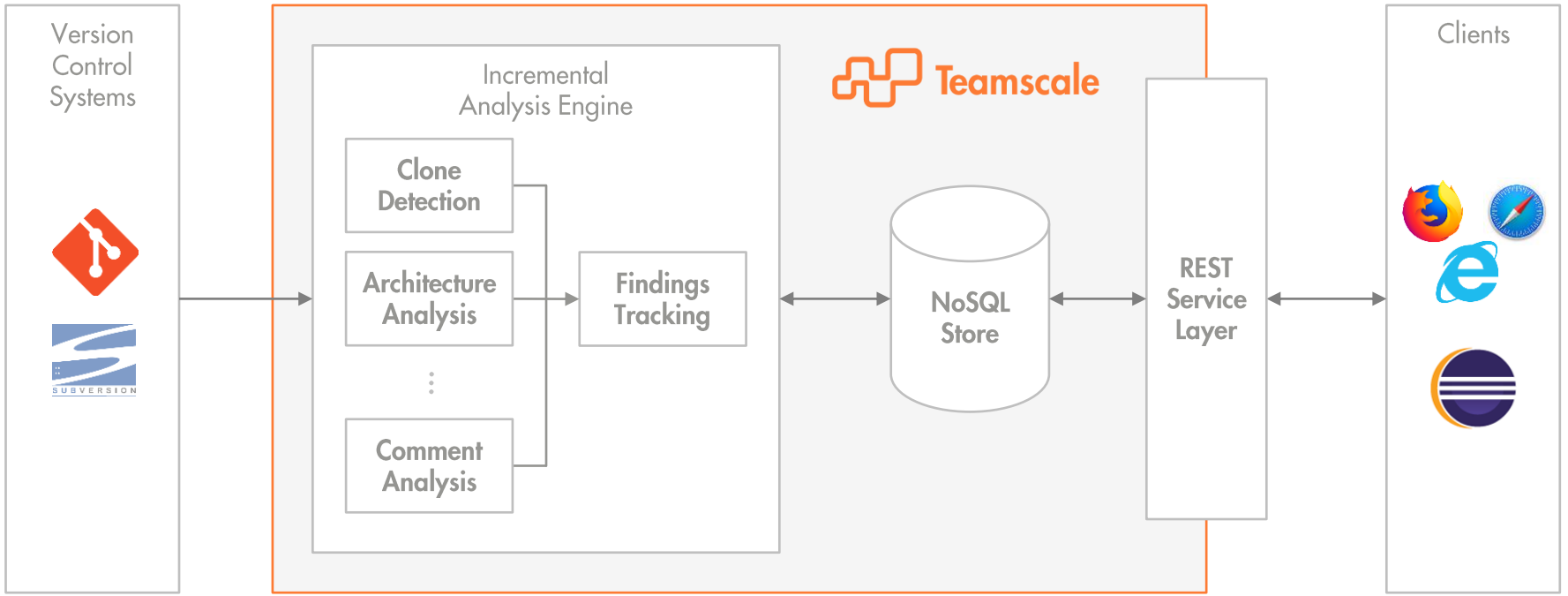
```
String readFileUtf16(File file) {
    FileInputStream in = new FileInputStream(file);
    byte[] buffer = new byte[file.length()];
    in.read(buffer); in.close();
    return new String(buffer, „UTF-16“);
}
```




BusinessRuleTaskXMLConverter.java
DefinitionsParser.java
model.setTargetNamespace(xtr.getAttributeValue(null, TARGET_NAMESPACE_ATTRIBUTE));
for (int i = 0; i < xtr.getNamespaceCount(); i++) {
String prefix = xtr.getNamespacePrefix(i);
if (StringUtils.isNotEmpty(prefix)) {
model.addNamespace(prefix, xtr.getNamespaceURI(i));
}
}
for (int i = 0; i < xtr.getAttributeCount(); i++) {
ExtensionAttribute extensionAttribute = new ExtensionAt
extensionAttribute.setName(xtr.getAttributeLocalName(i)
extensionAttribute.setValue(xtr.getAttributeValue(i));
if (StringUtils.isNotEmpty(xtr.getAttributeNamespace(i)
extensionAttribute.setNamespace(xtr.getAttributeNames
}
if (StringUtils.isNotEmpty(xtr.getAttributePrefix(i)))
extensionAttribute.setNamespacePrefix(xtr.getAttribut
}
if (!BpmnXMLUtil.isBlacklisted(extensionAttribute, defa
model.addDefinitionsAttribute(extensionAttribute);
Clone with 2 instances of length 10
}
}
Problems
Javadoc
Declaration
Findings
Orphans View
4 items
Description
Group
Clone with 2 instances of length 10
Code Duplica...
Interface comment missing
Documentati...
Interface comment missing
Documentati...
Name violates naming convention: defaultAttributes. Should be
Naming


jenkins/test/src/main/java/org/jvnet/hudson/test/HudsonTestCase.java
(revision 12d96a56...)
if (lhs==null && rhs==null) return;
if (lhs==null) fail("lhs is null while rhs="+rhs);
if (rhs==null) fail("rhs is null while lhs="+lhs);
Constructor<?> lc = findDataBoundConstructor(lhs.getClass());
Constructor<?> rc = findDataBoundConstructor(rhs.getClass());
assertEquals("Data bound constructor mismatch. Different type?",lc,rc);
List<String> primitiveProperties = new ArrayList<String>();
String[] names = ClassDescriptor.loadParameterNames(lc);
Class<?>[] types = lc.getParameterTypes();
assertEquals(names.length,types.length);
for (int i=0; i<types.length; i++) {
Object lv = ReflectionUtils.getPublicProperty(lhs, names[i]);
Object rv = ReflectionUtils.getPublicProperty(rhs, names[i]);
if (Iterable.class.isAssignableFrom(types[i])) {
Iterable lcol = (Iterable) lv;
Iterable rcol = (Iterable) rv;
Iterator ltr,rtr;
for (ltr=lcol.iterator(), rtr=rcol.iterator(); ltr.hasNext() && rtr.hasNext();)
Object litem = ltr.next();
Object ritem = rtr.next();
if (findDataBoundConstructor(litem.getClass())!=null) {
assertEqualDataBoundBeans(litem,ritem);
} else {
assertEquals(litem,ritem);
}
}
assertFalse("collection size mismatch between "+lhs+" and "+rhs, ltr.hasNext() ^
} else
if (findDataBoundConstructor(types[i])!=null || (lv!=null && findDataBoundConstructo
// recurse into nested databound objects
assertEqualDataBoundBeans(lv,rv);
} else {
primitiveProperties.add(names[i]);
}
}
// compare shallow primitive properties
if (!primitiveProperties.isEmpty())
assertEqualBeans(lhs,rhs,Util.join(primitiveProperties,","));
*
Makes sure that two collections are identical via {@link #assertEqualDataBoundBeans(Objec
/
blic void assertEqualDataBoundBeans(List<?> lhs, List<?> rhs) throws Exception {
assertEquals(lhs.size(), rhs.size());


jenkins/test/src/main/java/org/jvnet/hudson/test/JenkinsRule.java
if (lhs==null && rhs==null) return;
if (lhs==null) fail("lhs is null while rhs="+rhs);
if (rhs==null) fail("rhs is null while lhs="+lhs);
Constructor<?> lc = findDataBoundConstructor(lhs.getClas
Constructor<?> rc = findDataBoundConstructor(rhs.getClas
assertThat("Data bound constructor mismatch. Different t
List<String> primitiveProperties = new ArrayList<String>
String[] names = ClassDescriptor.loadParameterNames(lc);
Class<?>[] types = lc.getParameterTypes();
assertThat(types.length, is(names.length));
for (int i=0; i<types.length; i++) {
Object lv = ReflectionUtils.getPublicProperty(lhs, n
Object rv = ReflectionUtils.getPublicProperty(rhs, n
if (lv != null && rv != null && Iterable.class.isAss
Iterable lcol = (Iterable) lv;
Iterable rcol = (Iterable) rv;
Iterator ltr,rtr;
for (ltr=lcol.iterator(), rtr=rcol.iterator(); l
Object litem = ltr.next();
Object ritem = rtr.next();
if (findDataBoundConstructor(litem.getClass(
assertEqualDataBoundBeans(litem,ritem);
} else {
assertThat(ritem, is(litem));
}
}
assertThat("collection size mismatch between " +
is(false));
} else
if (findDataBoundConstructor(types[i])!=null || (lv!
// recurse into nested databound objects
assertEqualDataBoundBeans(lv,rv);
} else {
primitiveProperties.add(names[i]);
}
}
// compare shallow primitive properties
if (!primitiveProperties.isEmpty())
assertEqualBeans(lhs,rhs,Util.join(primitiveProperti
*
Makes sure that two collections are identical via {@link
/
blic void assertEqualDataBoundBeans(List<?> lhs, List<?> r


Version Control Systems
SUBVERSION
Incremental Analysis Engine
Clone Detection
Architecture Analysis
Comment Analysis
Findings Tracking
Teamscale
NoSQL Store
REST Service Layer
Clients

Sind Clones ein Problem für die Weiterentwicklung?

```java
// Utilities for arrays of elements
public String showElements(ModelElement[] elements, String nomsg) {
   boolean found = false;
   StringBuffer res = new StringBuffer();
   if (elements != null) {
      Index.getInstance().setCurrentRenderer(
            FlatReferenceRenderer.getInstance());
      for (int i = 0; i < elements.length; i++) {
         ModelElement el = elements[i];
         res.append(showElementLink(el)).append(HTML.LINE_BREAK);
         found = true;
      }
      Index.getInstance().resetCurrentRenderer();
   }
   if (!found && nomsg != null && nomsg.length() > 0) {
      res.append(HTML.italics(nomsg));
   }
   return res.toString();
}
```

```java
// Utilities for arrays of elements
public String showElements(ModelElement[] elements, String nomsg) {
   boolean found = false;
   StringBuffer res = new StringBuffer();
   if (elements != null) {
      Index.getInstance().setCurrentRenderer(
            FlatReferenceRenderer.getInstance());
      for (int i = 0; i < elements.length; i++) {
         ModelElement el = elements[i];
         res.append(showElementLink(el)).append(HTML.LINE_BREAK);
         found = true;
      }
      Index.getInstance().resetCurrentRenderer();
   }
   if (!found && nomsg.length() > 0) {
      res.append(HTML.italics(nomsg));
   }
   return res.toString();
}
```

Dashboard
Activity
Findings
Metrics
Tests
Issues
Tasks
Architecture
Delta
Projects
System
Admin

jenkins/test/src/main/java/org/jvnet/hudson/test/HudsonTestCase.java (revision 12d96a56...)

```
if (lhs==null && rhs==null)     return;
if (lhs==null)      fail("lhs is null while rhs="+rhs);
if (rhs==null)      fail("rhs is null while lhs="+lhs);

Constructor<?> lc = findDataBoundConstructor(lhs.getClass());
Constructor<?> rc = findDataBoundConstructor(rhs.getClass());
assertEquals("Data bound constructor mismatch. Different type?",lc,rc);

List<String> primitiveProperties = new ArrayList<String>();

String[] names = ClassDescriptor.loadParameterNames(lc);
Class<?>[] types = lc.getParameterTypes();
assertEquals(names.length,types.length);
for (int i=0; i<types.length; i++) {
    Object lv = ReflectionUtils.getPublicProperty(lhs, names[i]);
    Object rv = ReflectionUtils.getPublicProperty(rhs, names[i]);

    if (Iterable.class.isAssignableFrom(types[i])) {
        Iterable lcol = (Iterable) lv;
        Iterable rcol = (Iterable) rv;
        Iterator ltr,rtr;
        for (ltr=lcol.iterator(), rtr=rcol.iterator(); ltr.hasNext() && rtr.hasNext();)
            Object litem = ltr.next();
            Object ritem = rtr.next();

            if (findDataBoundConstructor(litem.getClass())!=null) {
                assertEqualDataBoundBeans(litem,ritem);
            } else {
                assertEquals(litem,ritem);
            }
        }
        assertFalse("collection size mismatch between "+lhs+" and "+rhs, ltr.hasNext() ^
    } else
    if (findDataBoundConstructor(types[i])!=null || (lv!=null && findDataBoundConstructo
        // recurse into nested databound objects
        assertEqualDataBoundBeans(lv,rv);
    } else {
        primitiveProperties.add(names[i]);
    }
}

// compare shallow primitive properties
if (!primitiveProperties.isEmpty())
    assertEqualBeans(lhs,rhs,Util.join(primitiveProperties,","));


*
 Makes sure that two collections are identical via {@link #assertEqualDataBoundBeans(Objec
/
blic void assertEqualDataBoundBeans(List<?> lhs, List<?> rhs) throws Exception {
  assertEquals(lhs.size(), rhs.size());
```

jenkins/test/src/main/java/org/jvnet/hudson/test/JenkinsRule.java (revis)

```
if (lhs==null && rhs==null)     return;
if (lhs==null)      fail("lhs is null while rhs="+rhs);
if (rhs==null)      fail("rhs is null while lhs="+lhs);

Constructor<?> lc = findDataBoundConstructor(lhs.getClass());
Constructor<?> rc = findDataBoundConstructor(rhs.getClass());
assertThat("Data bound constructor mismatch. Different type?", (Constructor)rc, is((Cons

List<String> primitiveProperties = new ArrayList<String>();

String[] names = ClassDescriptor.loadParameterNames(lc);
Class<?>[] types = lc.getParameterTypes();
assertThat(types.length, is(names.length));
for (int i=0; i<types.length; i++) {
    Object lv = ReflectionUtils.getPublicProperty(lhs, names[i]);
    Object rv = ReflectionUtils.getPublicProperty(rhs, names[i]);

    if (lv != null && rv != null && Iterable.class.isAssignableFrom(types[i])) {
        Iterable lcol = (Iterable) lv;
        Iterable rcol = (Iterable) rv;
        Iterator ltr,rtr;
        for (ltr=lcol.iterator(), rtr=rcol.iterator(); ltr.hasNext() && rtr.hasNext();)
            Object litem = ltr.next();
            Object ritem = rtr.next();

            if (findDataBoundConstructor(litem.getClass())!=null) {
                assertEqualDataBoundBeans(litem,ritem);
            } else {
                assertThat(ritem, is(litem));
            }
        }
        assertThat("collection size mismatch between " + lhs + " and " + rhs, ltr.hasNex
                is(false));
    } else
    if (findDataBoundConstructor(types[i])!=null || (lv!=null && findDataBoundConstructo
        // recurse into nested databound objects
        assertEqualDataBoundBeans(lv,rv);
    } else {
        primitiveProperties.add(names[i]);
    }
}

// compare shallow primitive properties
if (!primitiveProperties.isEmpty())
    assertEqualBeans(lhs,rhs,Util.join(primitiveProperties,","));


*
 Makes sure that two collections are identical via {@link #assertEqualDataBoundBeans(Objec
/
blic void assertEqualDataBoundBeans(List<?> lhs, List<?> rhs) throws Exception {
```

Collapse

```
    /* Process the input string received prior to the
    newline. */
    do
    {
        /* Pass the string to FreeRTOS+CLI. */
        xMoreDataToFollow = FreeRTOS_CLIProcessCommand( cInputString, cO

        /* Send the output generated by the command's
        implementation. */
        sendto( xSocket, cOutputString,  strlen( cOutputString ), 0, ( S

    } while( xMoreDataToFollow != pdFALSE ); /* Until the command does n

    /* All the strings generated by the command processing
    have been sent.  Clear the input string ready to receive
    the next command. */
    cInputIndex = 0;
    memset( cInputString, 0x00, cmdMAX_INPUT_SIZE );

    /* Transmit a spacer, just to make the command console
    easier to read. */
    sendto( xSocket, "\r\n",  strlen( "\r\n" ), 0, ( SOCKADDR * ) &xClie
}
else
{
    if( cInChar == '\r' )
    {
        /* Ignore the character.  Newlines are used to
        detect the end of the input string. */
    }
    else if( cInChar == '\b' )
    {
        /* Backspace was pressed.  Erase the last character
        in the string - if any. */
        if( cInputIndex > 0 )
        {
            cInputIndex--;
            cInputString[ cInputIndex ] = '\0';
        }
    }
    else
    {
```

```
    /* Process the input string received prior to the
    newline. */
    do
    {
        /* Pass the string to FreeRTOS+CLI. */
        xMoreDataToFollow = FreeRTOS_CLIProcessCommand( cInputString,

        /* Send the output generated by the command's
        implementation. */
        sendto( xSocket, cOutputString,  strlen( cOutputString ), 0,

    } while( xMoreDataToFollow != pdFALSE ); /* Until the command does

    /* All the strings generated by the command processing
    have been sent.  Clear the input string ready to receive
    the next command. */
    cInputIndex = 0;
    memset( cInputString, 0x00, cmdMAX_INPUT_SIZE );

    /* Transmit a spacer, just to make the command console
    easier to read. */
    sendto( xSocket, "\r\n",  strlen( "\r\n" ), 0, ( SOCKADDR * ) &xC
}
else
{
    if( cInChar == '\r' )
    {
        /* Ignore the character.  Newlines are used to
        detect the end of the input string. */
    }
    else if( ( cInChar == '\b' ) || ( cInChar == cmdASCII_DEL ) )
    {
        /* Backspace was pressed.  Erase the last character
        in the string - if any. */
        if( cInputIndex > 0 )
        {
            cInputIndex--;
            cInputString[ cInputIndex ] = '\0';
        }
    }
    else
    {
```

ABAP

```
    if i_kart-life_value < 0.
      write '50'                to it_down-newbs.
      i_kart-life_value = i_kart-life_value * -1.
      write i_kart-life_value       to it_down-wrbtr
                                currency i_kost_ges-currency.
    endif.
    it_down-xblnr              =  w_xblnr.
    it_down-zuonr              =  w_zuonr.
    it_down-sgtxt              =  w_sgtxt.
    it_down-kostl           = i_kost_ges-fi_life_cost_center.
    it_down-zzvber             = '10'.
    it_down-bukrs              = i_kost_ges-booking.
*
    if it_down-bukrs ne '0001'.
      write 'SA'                to it_down-blart.
      clear it_down-zzvber.
    endif.
*
    replace all occurrences of '.' in it_down-wrbtr with ''.
    append it_down.
  endloop.
*
  describe table it_down lines l_it_down.
  it_down-blkz = 'E'.
  modify it_down index l_it_down transporting blkz.
endloop.
*
*--- Non-Life Quartal
loop at i_kost_ges where period_type = 'q' and
                         non_life_value ne 0.
*                    and    fehler ne 'X'.
  if i_kost_ges-fibu_type = 'i'.
    d_fibu_type = text-k01.
```

```
    if i_kart-non_life_value < 0.
      write '50'                to it_down-newbs.
      i_kart-non_life_value = i_kart-non_life_value * -1.
      write i_kart-non_life_value    to it_down-wrbtr
                                currency i_kost_ges-currency.
    endif.
*
    it_down-xblnr              = w_xblnr.
    it_down-zuonr              = w_zuonr.
    it_down-sgtxt              = w_sgtxt.
    it_down-kostl          = i_kost_ges-fi_non_life_cost_center.
    it_down-zzvber             = '10'.
    it_down-bukrs              = i_kost_ges-booking.
*
    if it_down-bukrs ne '0001'.
      write 'SA'                to it_down-blart.
      clear it_down-zzvber.
    endif.
*
    append it_down.
*
  endloop.
  describe table it_down lines l_it_down.
  it_down-blkz = 'E'.
  modify it_down index l_it_down transporting blkz.
endloop.
*
*--- Monatlich Health
loop at i_kost_ges where period_type = 'm' and
                         health_value ne 0.
*                    fehler    ne 'X'.
  if i_kost_ges-fibu_type = 'i'.
    d_fibu_type = text-k01.
```

PL/SQL

larger/source/.../PriceElementUtil.plsql (revision default:1...)

```
223        AND   NVL(t.valid_from, Database_SYS.Get_First_(
224        AND   NVL(t.valid_to, Database_SYS.Get_Last_Cale
225
226  BEGIN
227     OPEN get_price_interval;
228     FETCH get_price_interval INTO valid_from_, valid_ur
229     CLOSE get_price_interval;
230     IF valid_from_ > Site_API.Get_Site_Date(contract_)
231        valid_date_ := valid_from_;
232     ELSE
233        valid_date_ := LEAST(NVL(valid_until_, TRUNC(Sit
234     END IF;
235
236     -- only calculate unit base (False) but no metal su
237     temp1_ :=0;
238
239     OPEN get_pl_currency_code;
240     FETCH get_pl_currency_code INTO pl_currency_code_;
241     CLOSE get_pl_currency_code;
242
243     FOR rec_ IN get_all_conn_unit_base_false LOOP
```

larger/source/.../PriceElementUtil.plsql (revision default:1...)

```
346  BEGIN
347     OPEN get_price_interval;
348     FETCH get_price_interval INTO valid_from_, valid_ur
349     CLOSE get_price_interval;
350     IF valid_from_ > Site_API.Get_Site_Date(contract_)
351        valid_date_ := valid_from_;
352     ELSE
353        -- Change for Ticket #1234 (START)
354        --valid_date_ := LEAST(NVL(valid_until_, TRUNC(S
355        valid_date_ := LEAST(NVL(valid_until_, Site_API.
356        -- Change for Ticket #1234 (FINISH)
357     END IF;
358
359     -- calculate only metal surchage without unit base
360     temp3_ :=0;
361
362     OPEN get_pl_currency_code;
363     FETCH get_pl_currency_code INTO pl_currency_code_;
364     CLOSE get_pl_currency_code;
365
366     FOR rec_ IN get_all_metal_surchage LOOP
```

Gosu

```
54        orCriteria.compare(TerrismTargetTier_Ext#County, Equals, polLocation.County)
55        orCriteria.compare(TerrismTargetTier_Ext#County, Equals, null)
56      })
57      tierQuery.or(\orCriteria -> {
58        orCriteria.compare(TerrismTargetTier_Ext#City, Equals, polLocation.City)
59        orCriteria.compare(TerrismTargetTier_Ext#City, Equals, null)
60      })
61      tierQuery.or(\orCriteria -> {
62        orCriteria.compare(TerrismTargetTier_Ext#ZipCode, Equals, polLocation.PostalCode)
63        orCriteria.compare(TerrismTargetTier_Ext#ZipCode, Equals, null)
64      })
65
66      var results = tierQuery.select()
67      if(results.HasElements) {
68        var result = results.firstWhere( \ elt -> elt.ZipCode == polLocation.PostalCode and
69            elt.County == polLocation.County and elt.City == polLocation.City)
70        var result2 = results.firstWhere( \ elt -> elt.County == polLocation.County and elt.City == p
71        var result3 = results.firstWhere( \ elt -> elt.County == polLocation.County)
72
73        if(result != null) {
```

```
54        orCriteria.compareIgnoreCase(TerrismTargetTier_Ext#County, Equals, polLocation.County)
55        orCriteria.compare(TerrismTargetTier_Ext#County, Equals, null)
56      })
57      tierQuery.or(\orCriteria -> {
58        orCriteria.compare(TerrismTargetTier_Ext#City, Equals, polLocation.City)
59        orCriteria.compare(TerrismTargetTier_Ext#City, Equals, null)
60      })
61      tierQuery.or(\orCriteria -> {
62        orCriteria.compare(TerrismTargetTier_Ext#ZipCode, Equals, polLocation.PostalCode)
63        orCriteria.compare(TerrismTargetTier_Ext#ZipCode, Equals, null)
64      })
65
66      var results = tierQuery.select()
67      if(results.HasElements) {
68        var result = results.firstWhere( \ elt -> elt.ZipCode == polLocation.PostalCode and
69            elt.County == polLocation.County and elt.City == polLocation.City)
70        var result2 = results.firstWhere( \ elt -> elt.County == polLocation.County and elt.City == p
71        //[redacted] - Defect 224[redacted] - Fix for defaulting target tier value irrespective of lower and upper
72        var result3 = results.firstWhere( \ elt -> elt.County.equalsIgnoreCase(polLocation.County))
73
74        if(result != null) {
```

Study

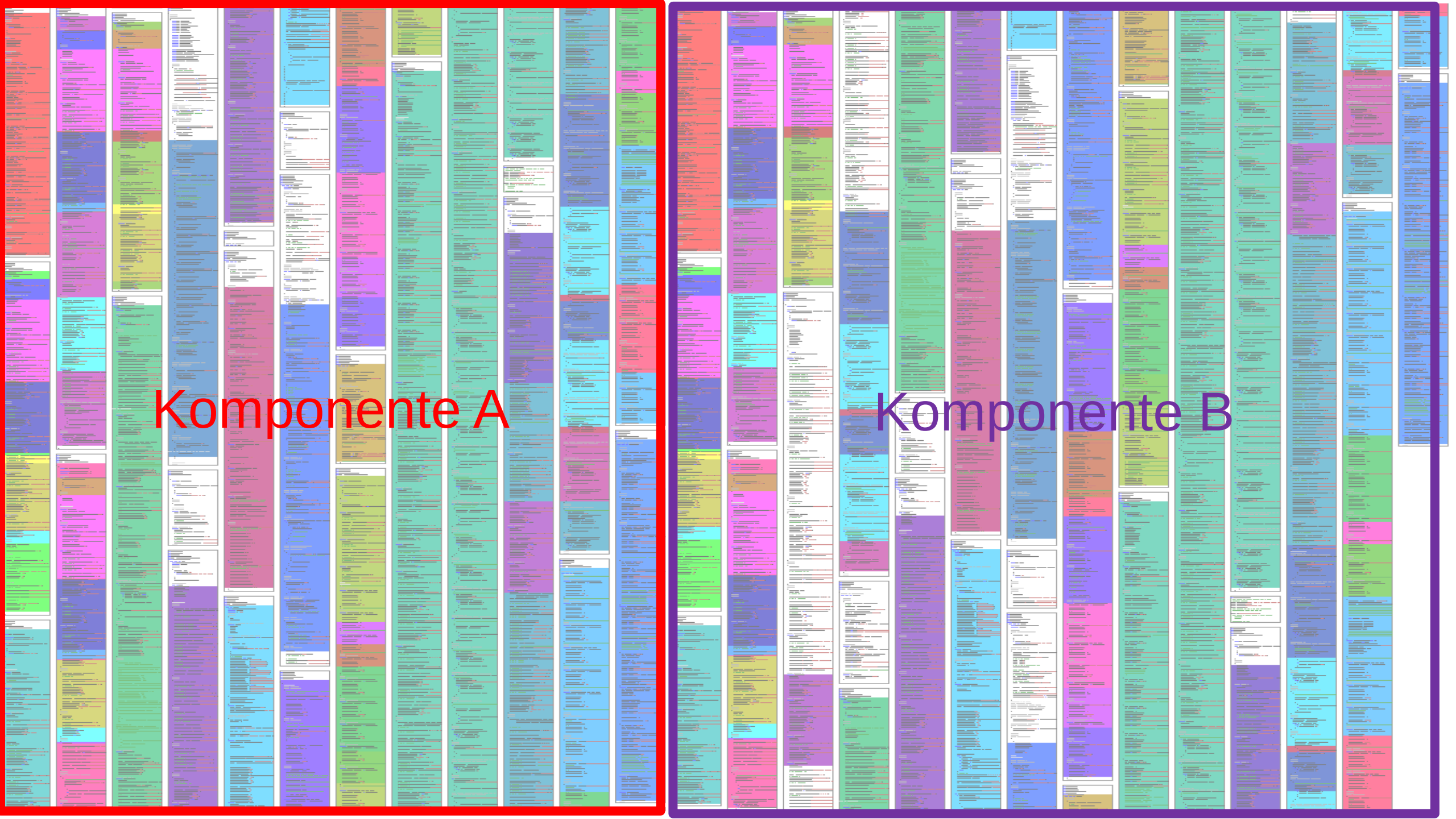
Munich Re LV 1871

- Over 100 Bugs in produktive Software

- 52% of all unintentional inconsistencies are buggy

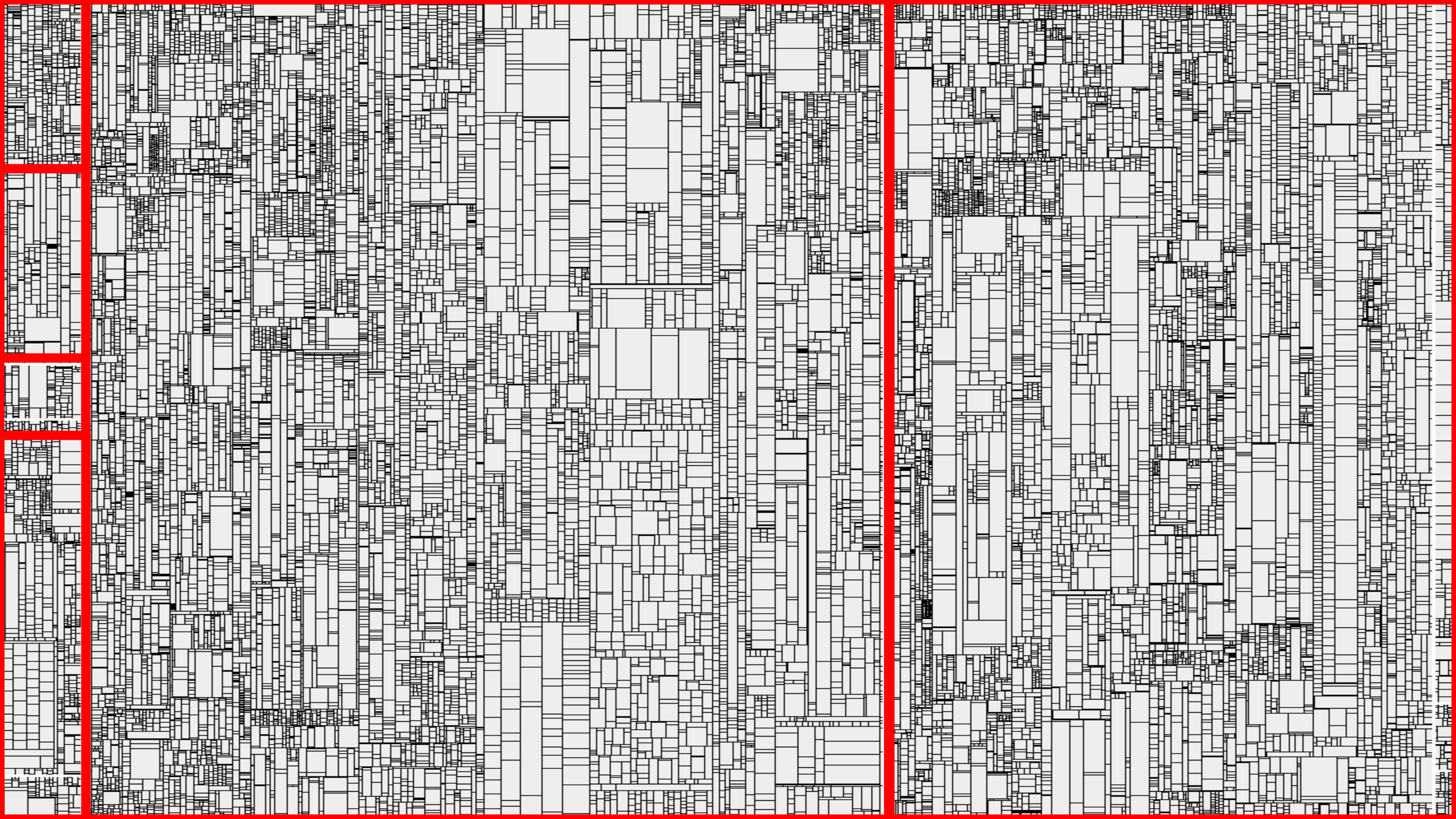
Juergens, Deissenboeck et al: *Do Code Clones Matter?* ICSE 2009

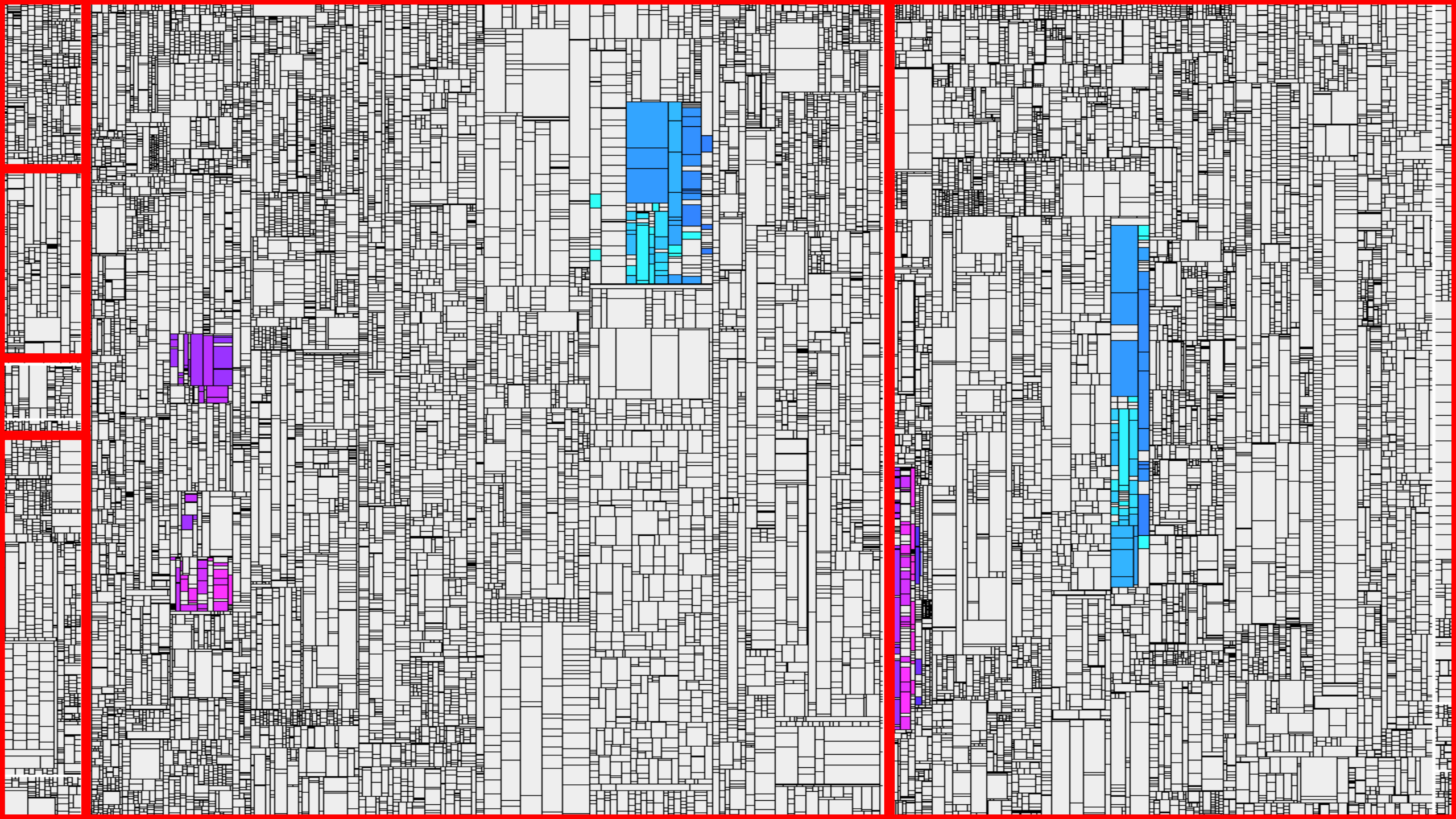
Was tun?

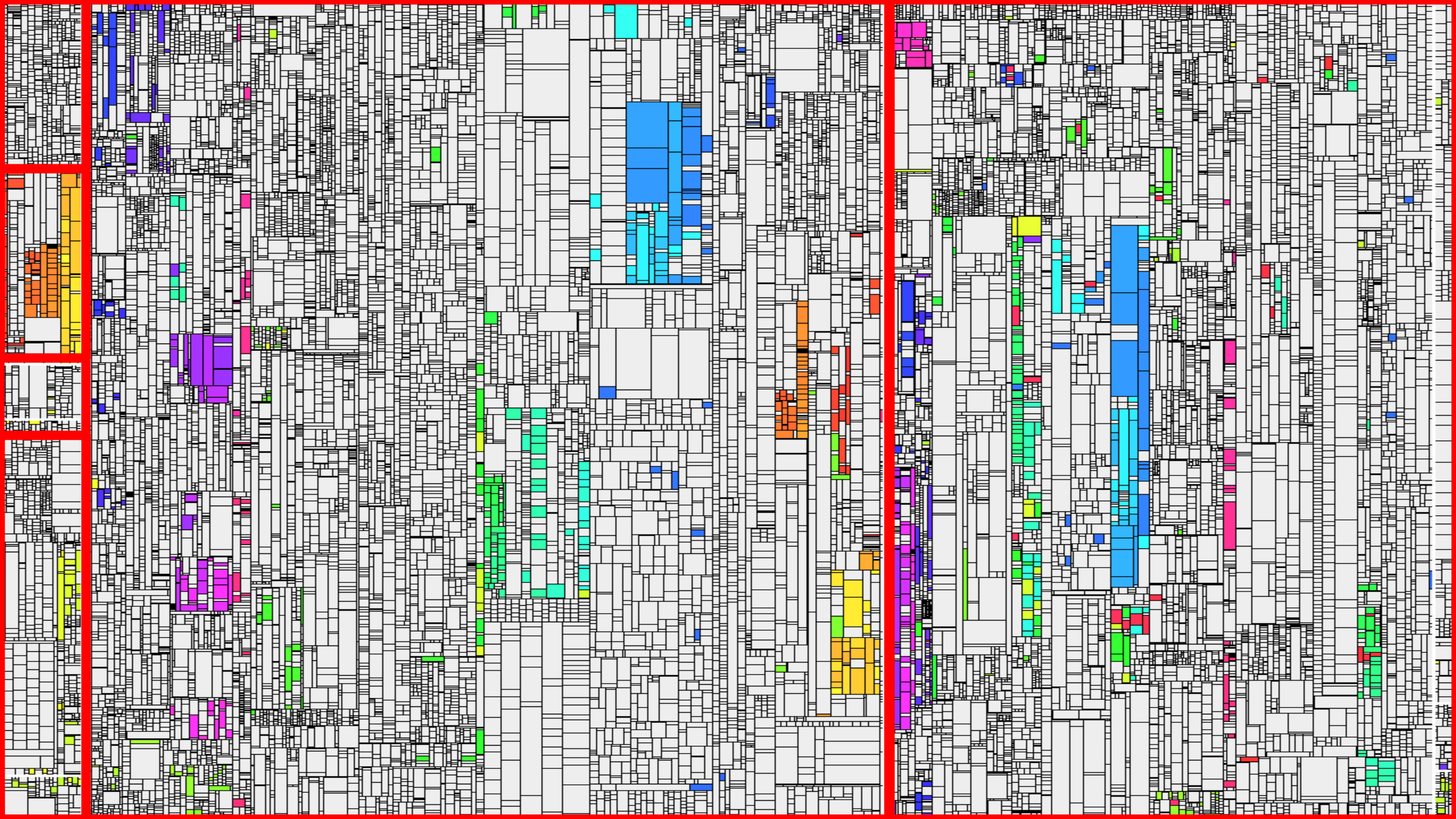

Komponente A
Komponente B

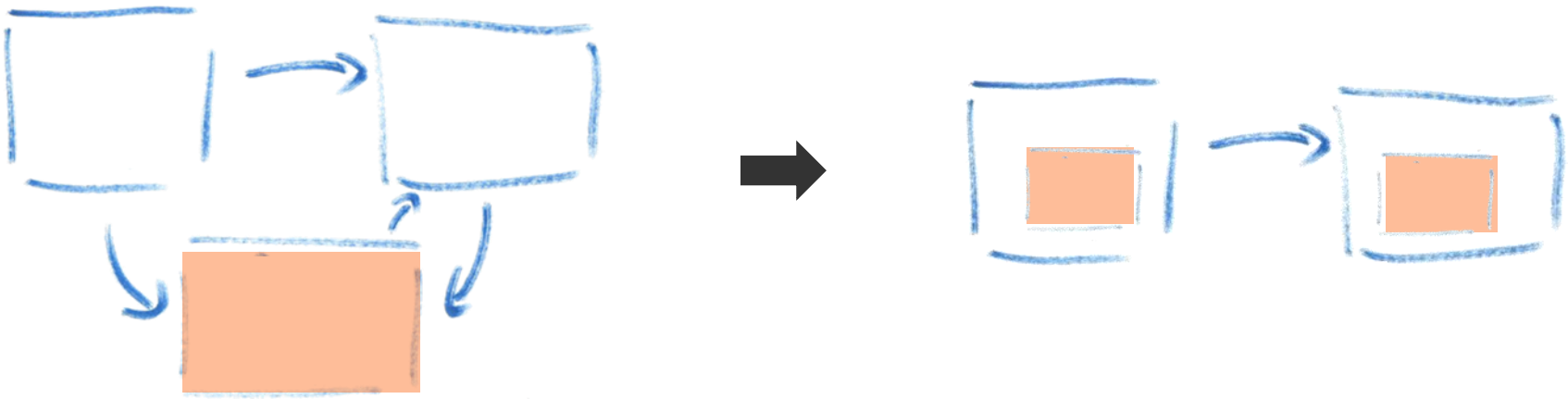
Ad-Hoc Reuse in a Build-System

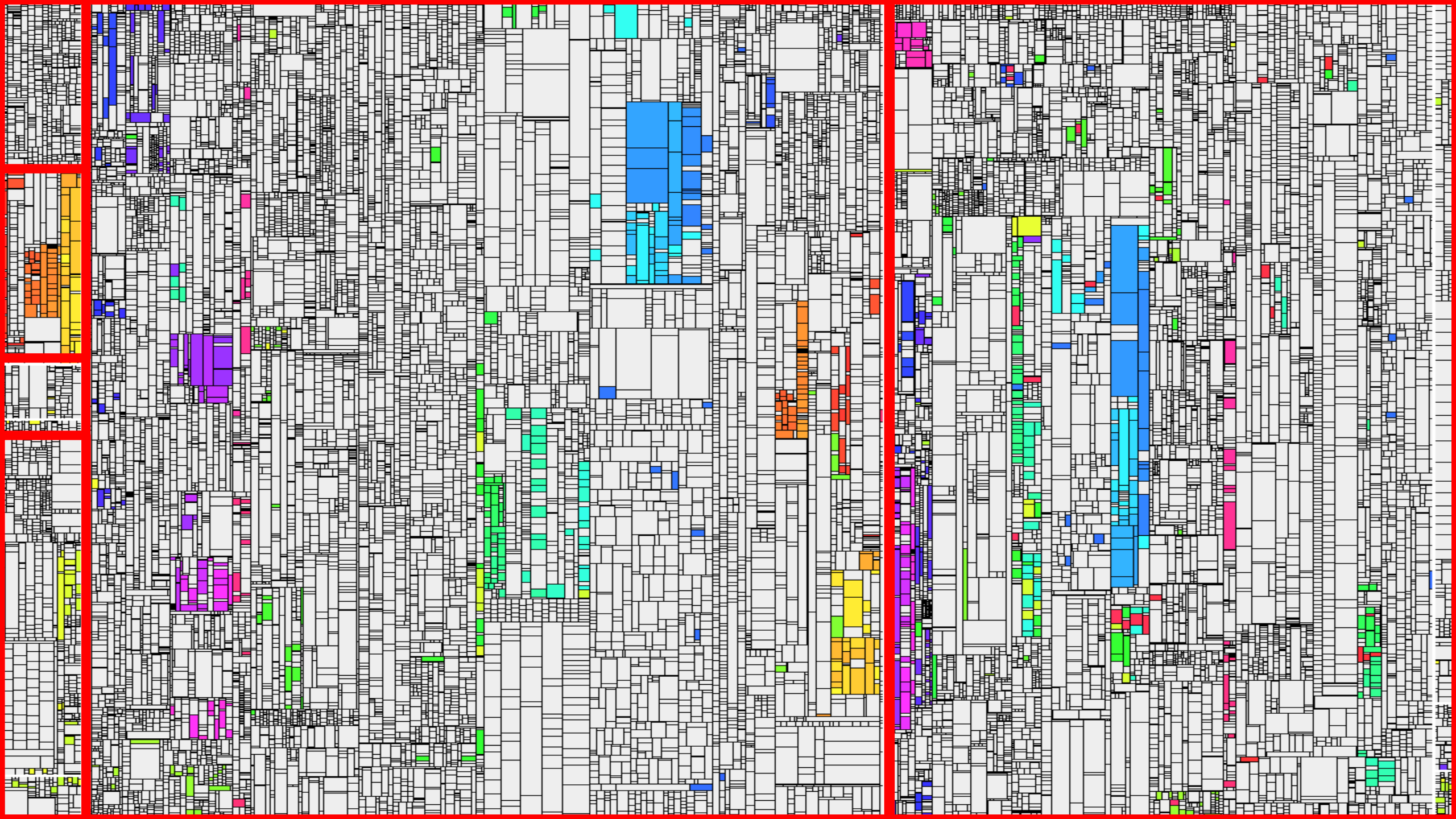
	Ad hoc Reuse
Build Definitions	81
Lines of Code	457.388
Clone Coverage	97,5%
Blow-Up	929%











Dashboard
Activity
Findings
Metrics
Tests
Issues
Tasks
Architecture
Delta
Projects
System
Admin

Collapse

jenkins/test/src/main/java/org/jvnet/hudson/test/HudsonTestCase.java (revision 12d96a56...)

```
 if (lhs==null && rhs==null)     return;
 if (lhs==null)      fail("lhs is null while rhs="+rhs);
 if (rhs==null)      fail("rhs is null while lhs="+lhs);

 Constructor<?> lc = findDataBoundConstructor(lhs.getClass());
 Constructor<?> rc = findDataBoundConstructor(rhs.getClass());
 assertEquals("Data bound constructor mismatch. Different type?",lc,rc);

 List<String> primitiveProperties = new ArrayList<String>();

 String[] names = ClassDescriptor.loadParameterNames(lc);
 Class<?>[] types = lc.getParameterTypes();
 assertEquals(names.length,types.length);
 for (int i=0; i<types.length; i++) {
     Object lv = ReflectionUtils.getPublicProperty(lhs, names[i]);
     Object rv = ReflectionUtils.getPublicProperty(rhs, names[i]);

     if (Iterable.class.isAssignableFrom(types[i])) {
         Iterable lcol = (Iterable) lv;
         Iterable rcol = (Iterable) rv;
         Iterator ltr,rtr;
         for (ltr=lcol.iterator(), rtr=rcol.iterator(); ltr.hasNext() && rtr.hasNext();)
             Object litem = ltr.next();
             Object ritem = rtr.next();

             if (findDataBoundConstructor(litem.getClass())!=null) {
                 assertEqualDataBoundBeans(litem,ritem);
             } else {
                 assertEquals(litem,ritem);
             }
         }
         assertFalse("collection size mismatch between "+lhs+" and "+rhs, ltr.hasNext() ^
     } else
     if (findDataBoundConstructor(types[i])!=null || (lv!=null && findDataBoundConstructo
         // recurse into nested databound objects
         assertEqualDataBoundBeans(lv,rv);
     } else {
         primitiveProperties.add(names[i]);
     }
 }

 // compare shallow primitive properties
 if (!primitiveProperties.isEmpty())
     assertEqualBeans(lhs,rhs,Util.join(primitiveProperties,","));


*
 Makes sure that two collections are identical via {@link #assertEqualDataBoundBeans(Objec
/
blic void assertEqualDataBoundBeans(List<?> lhs, List<?> rhs) throws Exception {
 assertEquals(lhs.size(), rhs.size());
```

jenkins/test/src/main/java/org/jvnet/hudson/test/JenkinsRule.java (revision 3909f5ac...)

```
 if (lhs==null && rhs==null)     return;
 if (lhs==null)      fail("lhs is null while rhs="+rhs);
 if (rhs==null)      fail("rhs is null while lhs="+lhs);

 Constructor<?> lc = findDataBoundConstructor(lhs.getClass());
 Constructor<?> rc = findDataBoundConstructor(rhs.getClass());
 assertThat("Data bound constructor mismatch. Different type?", (Constructor)rc, is((Cons

 List<String> primitiveProperties = new ArrayList<String>();

 String[] names = ClassDescriptor.loadParameterNames(lc);
 Class<?>[] types = lc.getParameterTypes();
 assertThat(types.length, is(names.length));
 for (int i=0; i<types.length; i++) {
     Object lv = ReflectionUtils.getPublicProperty(lhs, names[i]);
     Object rv = ReflectionUtils.getPublicProperty(rhs, names[i]);

     if (lv != null && rv != null && Iterable.class.isAssignableFrom(types[i])) {
         Iterable lcol = (Iterable) lv;
         Iterable rcol = (Iterable) rv;
         Iterator ltr,rtr;
         for (ltr=lcol.iterator(), rtr=rcol.iterator(); ltr.hasNext() && rtr.hasNext();)
             Object litem = ltr.next();
             Object ritem = rtr.next();

             if (findDataBoundConstructor(litem.getClass())!=null) {
                 assertEqualDataBoundBeans(litem,ritem);
             } else {
                 assertThat(ritem, is(litem));
             }
         }
         assertThat("collection size mismatch between " + lhs + " and " + rhs, ltr.hasNex
                 is(false));
     } else
     if (findDataBoundConstructor(types[i])!=null || (lv!=null && findDataBoundConstructo
         // recurse into nested databound objects
         assertEqualDataBoundBeans(lv,rv);
     } else {
         primitiveProperties.add(names[i]);
     }
 }

 // compare shallow primitive properties
 if (!primitiveProperties.isEmpty())
     assertEqualBeans(lhs,rhs,Util.join(primitiveProperties,","));


*
 Makes sure that two collections are identical via {@link #assertEqualDataBoundBeans(Objec
/
blic void assertEqualDataBoundBeans(List<?> lhs, List<?> rhs) throws Exception {
```

fixed: latest change is no longer lost when assigning entry to a keyword group while it is being edited

by jzieren in revision e0ca9a51b50c8b01f579f4eef79028bff6c34028 (git)

May 26 2005
15:58

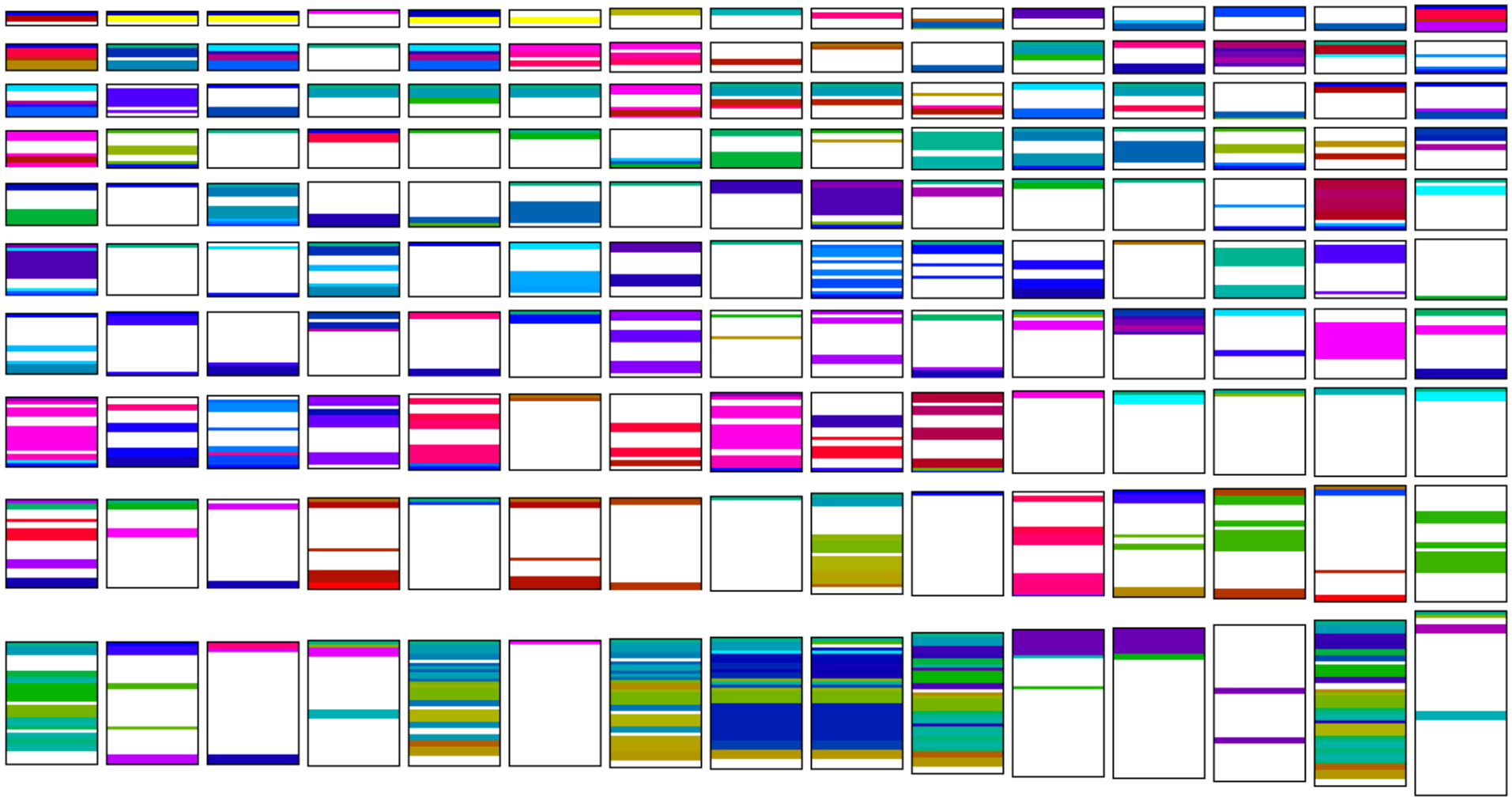
1 alerts:

Message	Context
Found potential inconsistent clone change in RightClickMenu.java	[Broken clone] [Old clone finding] [Code change]

2 removed findings:

Message	Location	Finding Group
Clone with 2 instances of length 10	src/java/net/sf/.../RightClickMenu.java:366-380	Code Duplication / Cloning
Clone with 2 instances of length 10	src/java/net/sf/.../RightClickMenu.java:340-354	Code Duplication / Cloning

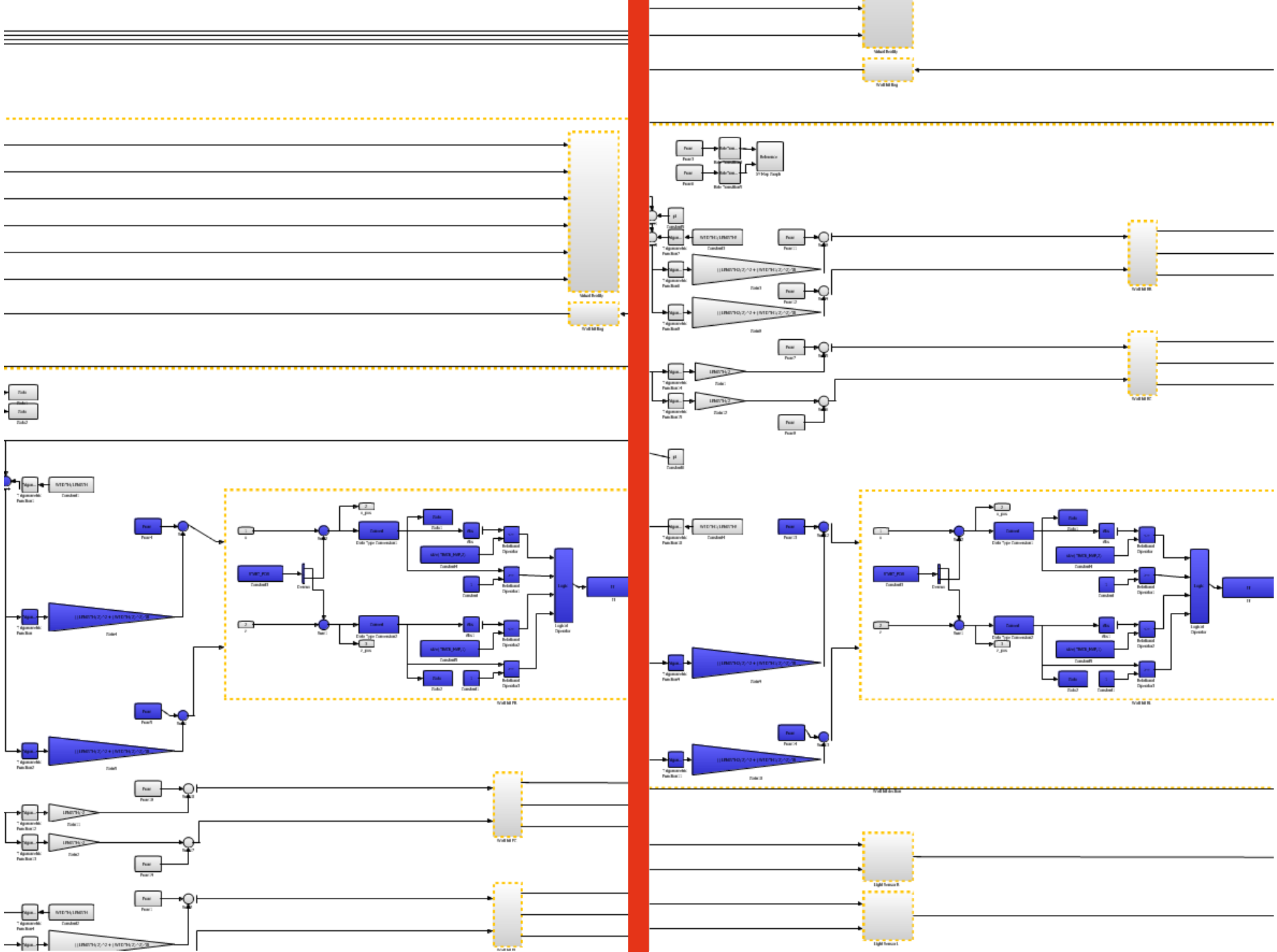
Software ist mehr als Code



176 use cases in total, 150 contain cloning

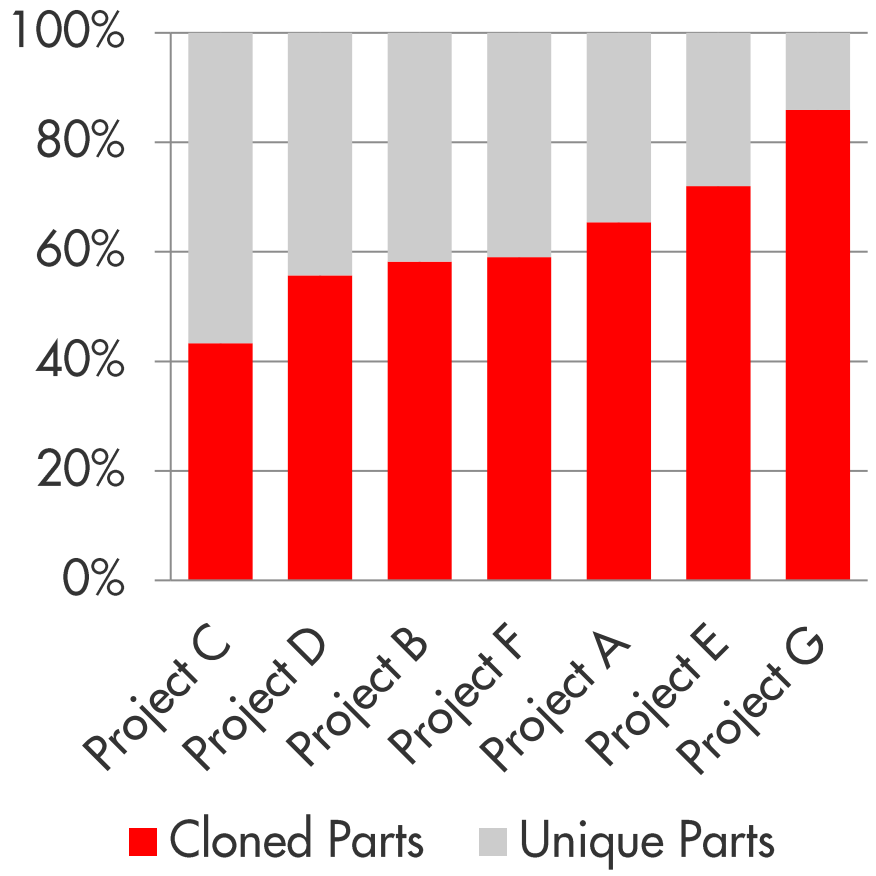
Spec	Pages	Words	Clone cov.	Clone groups	Clones	Blow-up relative	Blow-up words
A	517	41,482	35.0%	259	914	32.6%	10,191
B	1,013	130,968	8.9%	265	639	5.3%	6,639
C	133	18,447	18.5%	37	88	11.5%	1,907
D	241	37,969	8.1%	105	479	6.9%	2,463
E	185	37,056	0.9%	6	12	0.4%	161
F	42	7,662	51.1%	50	162	60.6%	2,890
H	160	19,632	71.6%	71	360	129.6%	11,083
X	158	19,679	12.4%	21	45	6.8%	1,253
Y	235	49,425	21.9%	181	553	18.2%	7,593
Z	[illegible]	13,807	19.6%	50	117	14.2%	1,718
AB	3,100	274,489	12.1%	635	1818	8.7%	21,993
AC	696	81,410	5.4%	65	148	3.2%	2,549
Avg			13.6%			13.5%	
Σ	8,667	1,242,765		2,631	7,669		100,178

Juergens et al: *Can Clone Detection Support QA of Requirement Specs?*, ICSE 2010

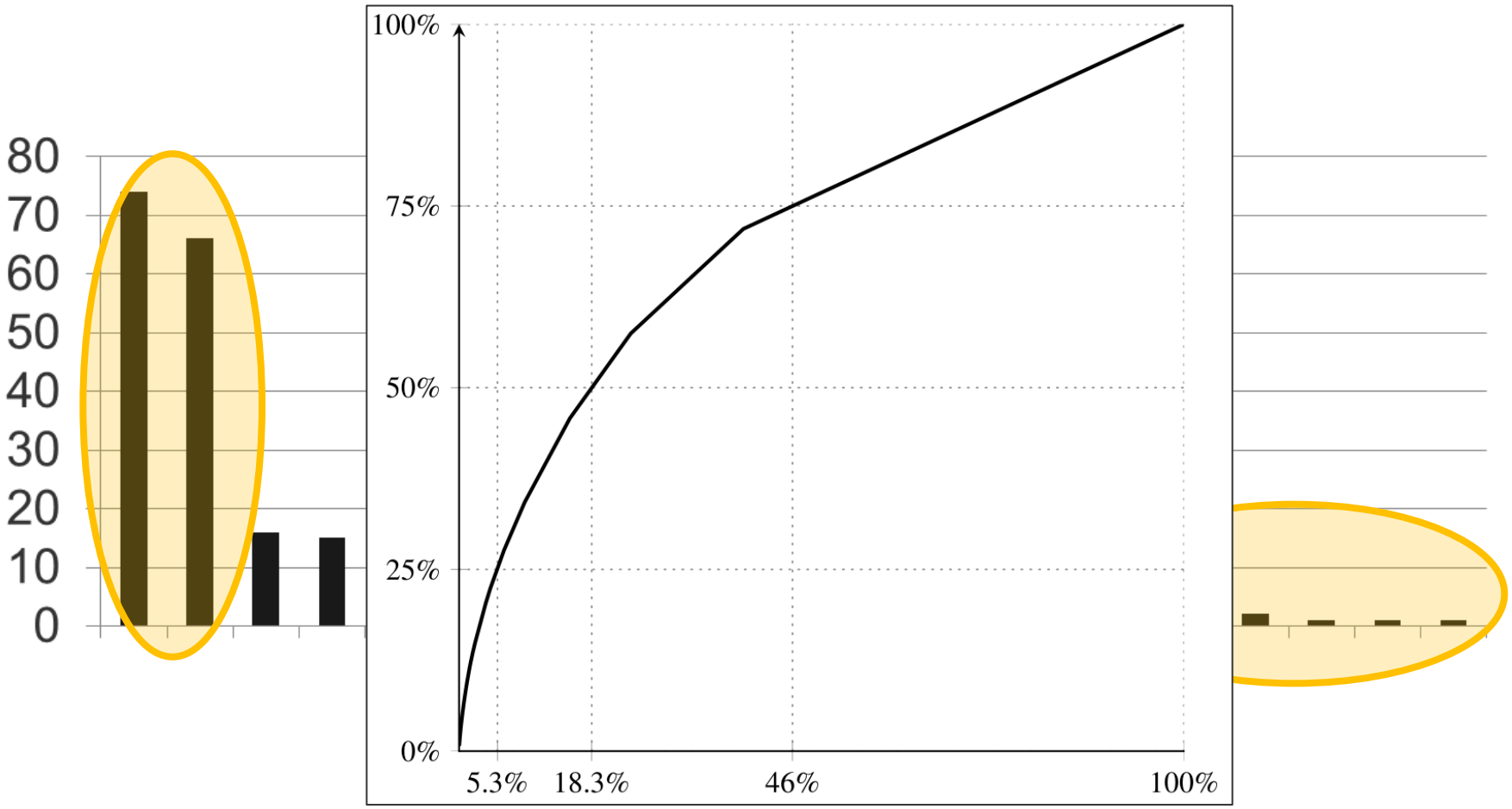


Munich RE

	System under Test	Test Suite
		# Tests
System A	330 kLoC	266
System B	580 kLoC	1,059
System C	150 kLoC	72
System D	430 kLoC	180
System E	760 kLoC	1,804
System F	1,400 kLoC	135
System G	160 kLoC	605

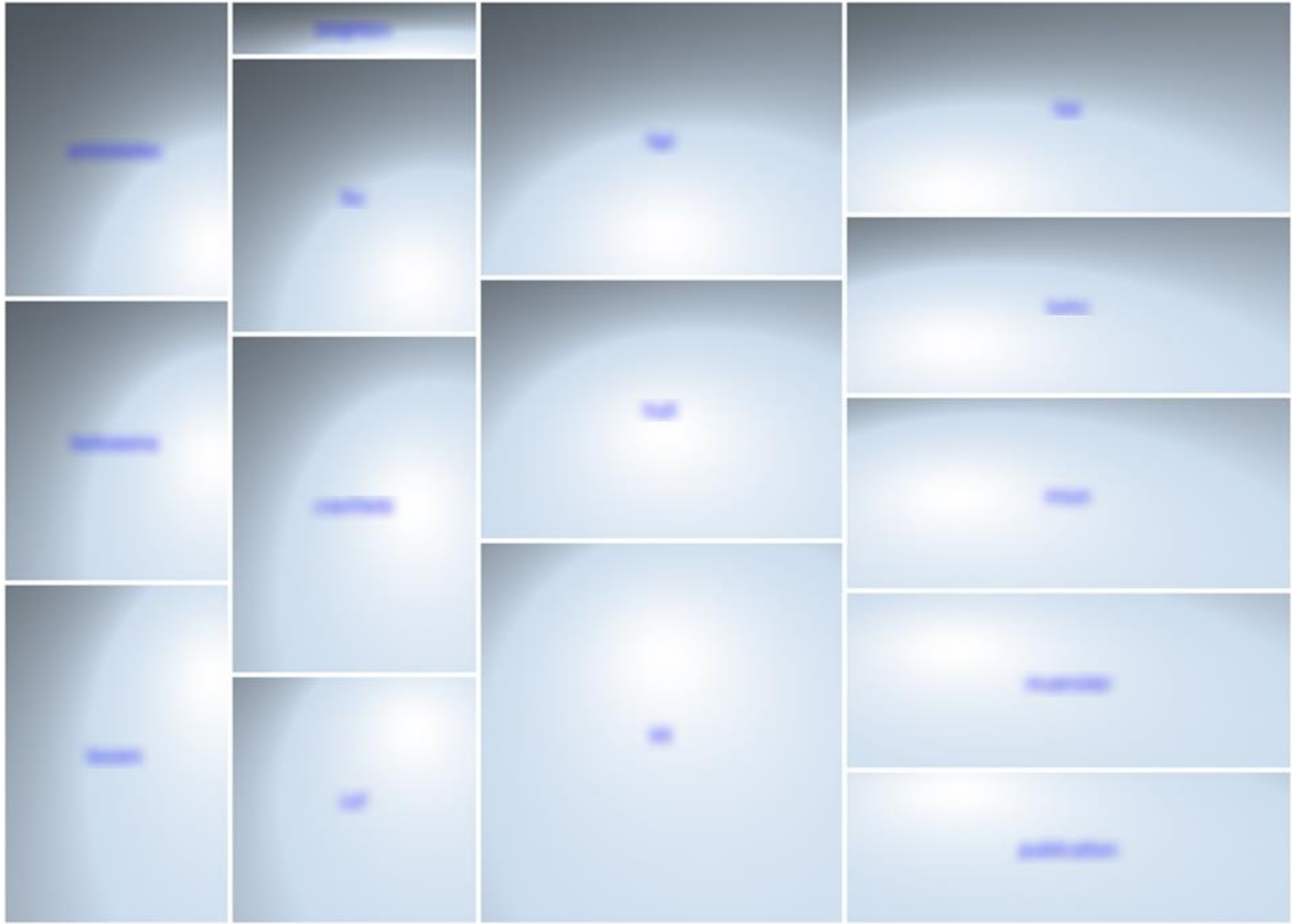


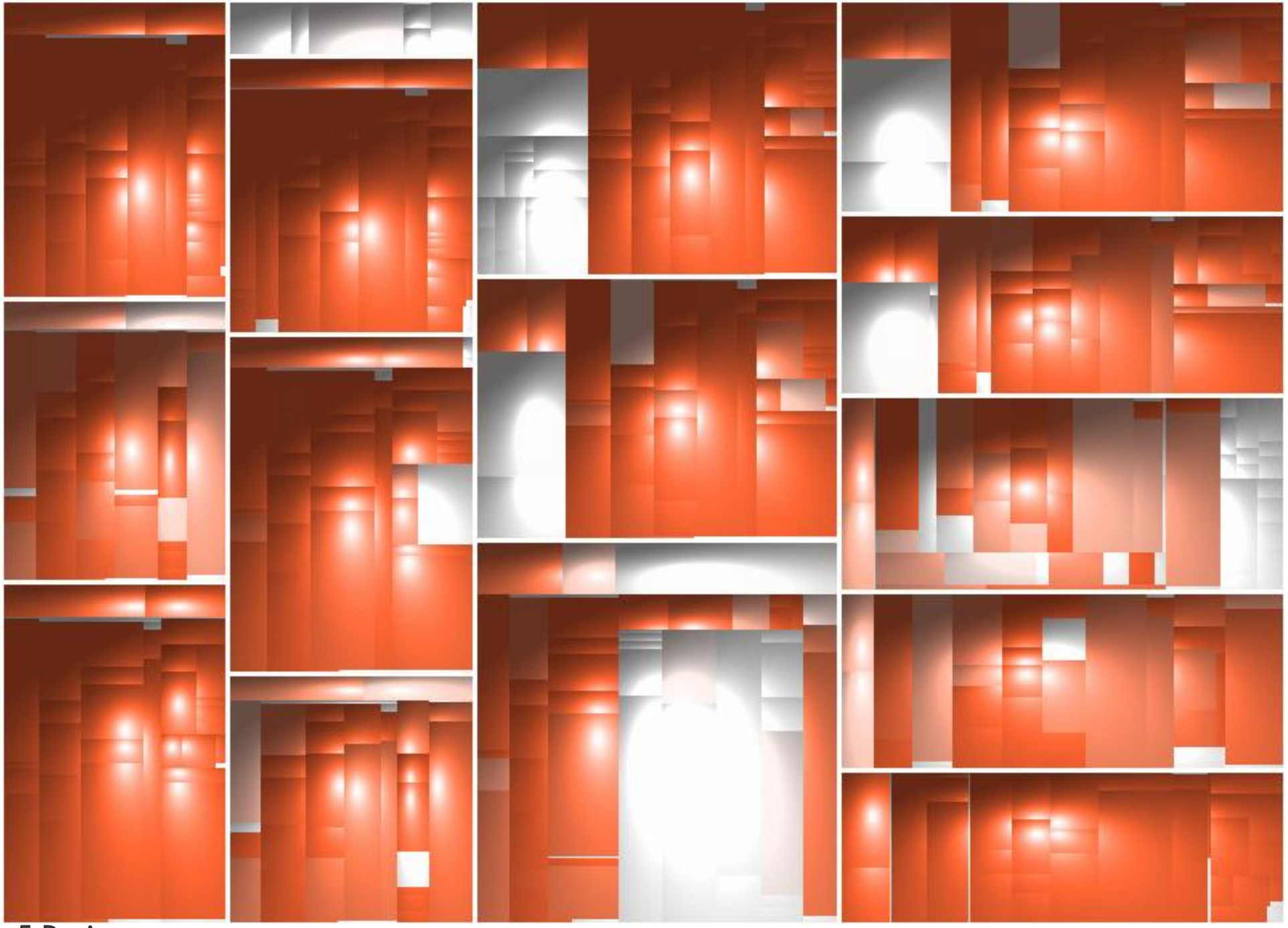
Hauptmann, Juergens et al: *Can Clone Detection Support Test Comprehension?*, ICPC 2012



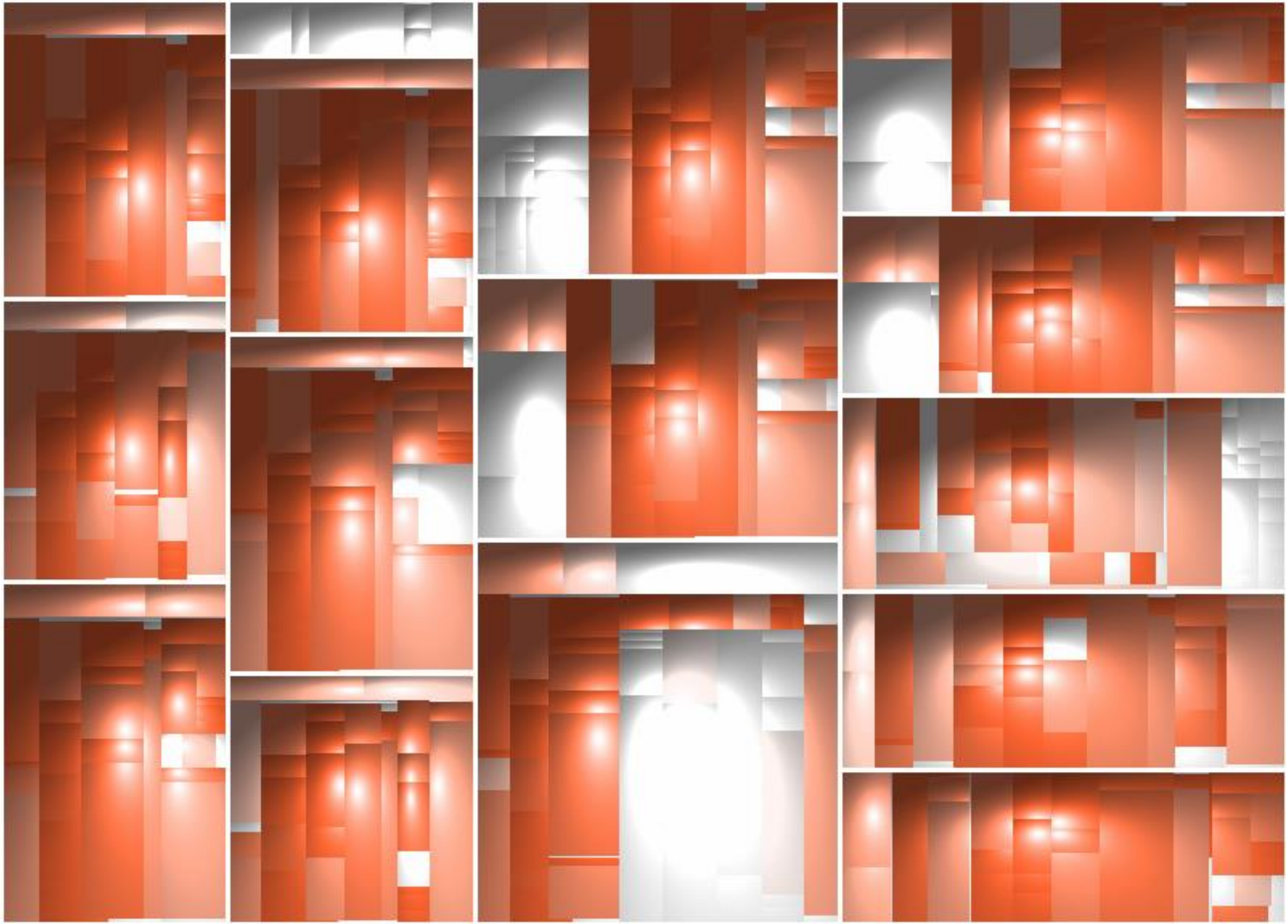
Hauptmann, Juergens et al: *Can Clone Detection Support Test Comprehension?*, ICPC 2012

Evolution von Produktlinien

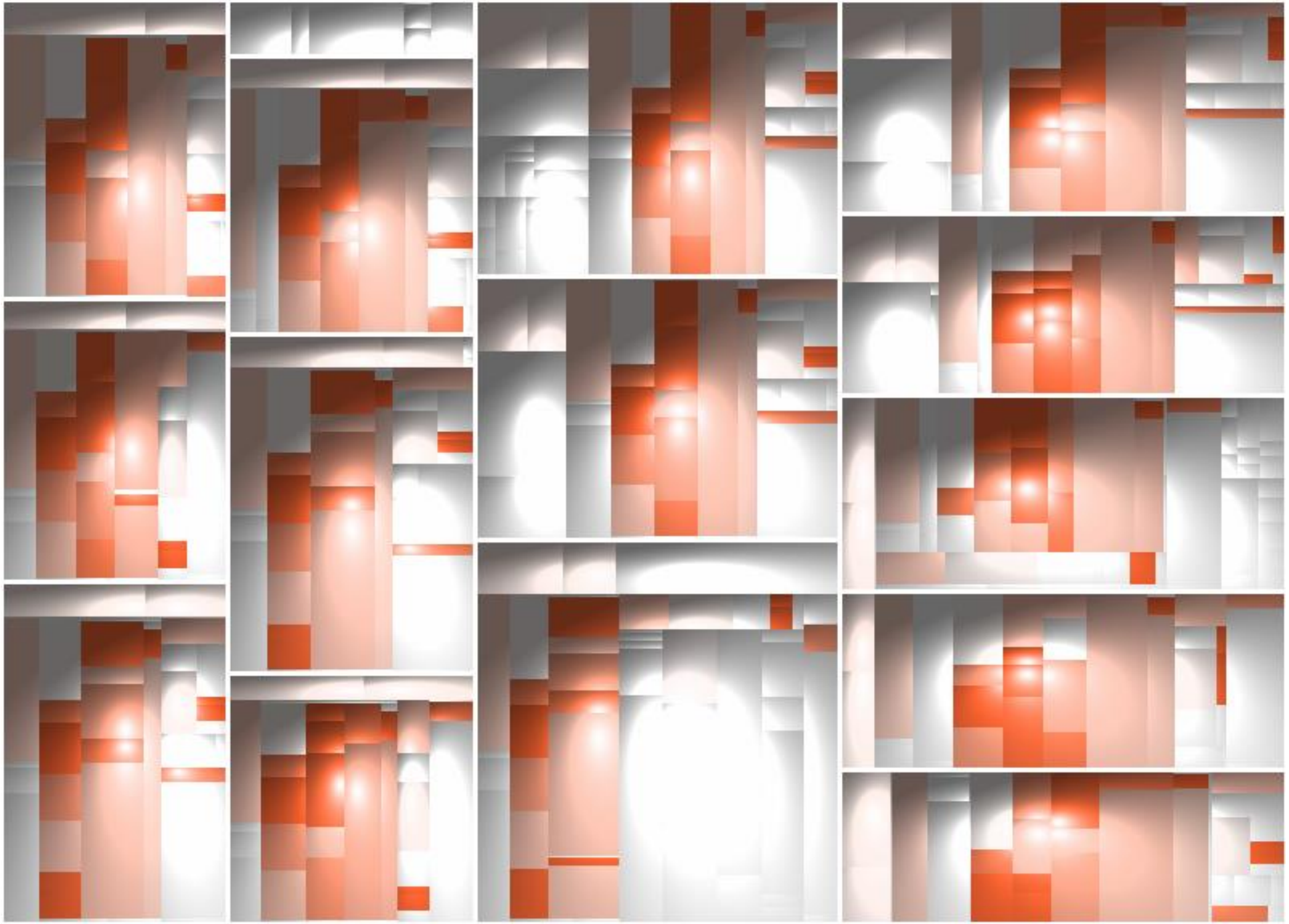




5 Projects



10 Projects



14 Projects

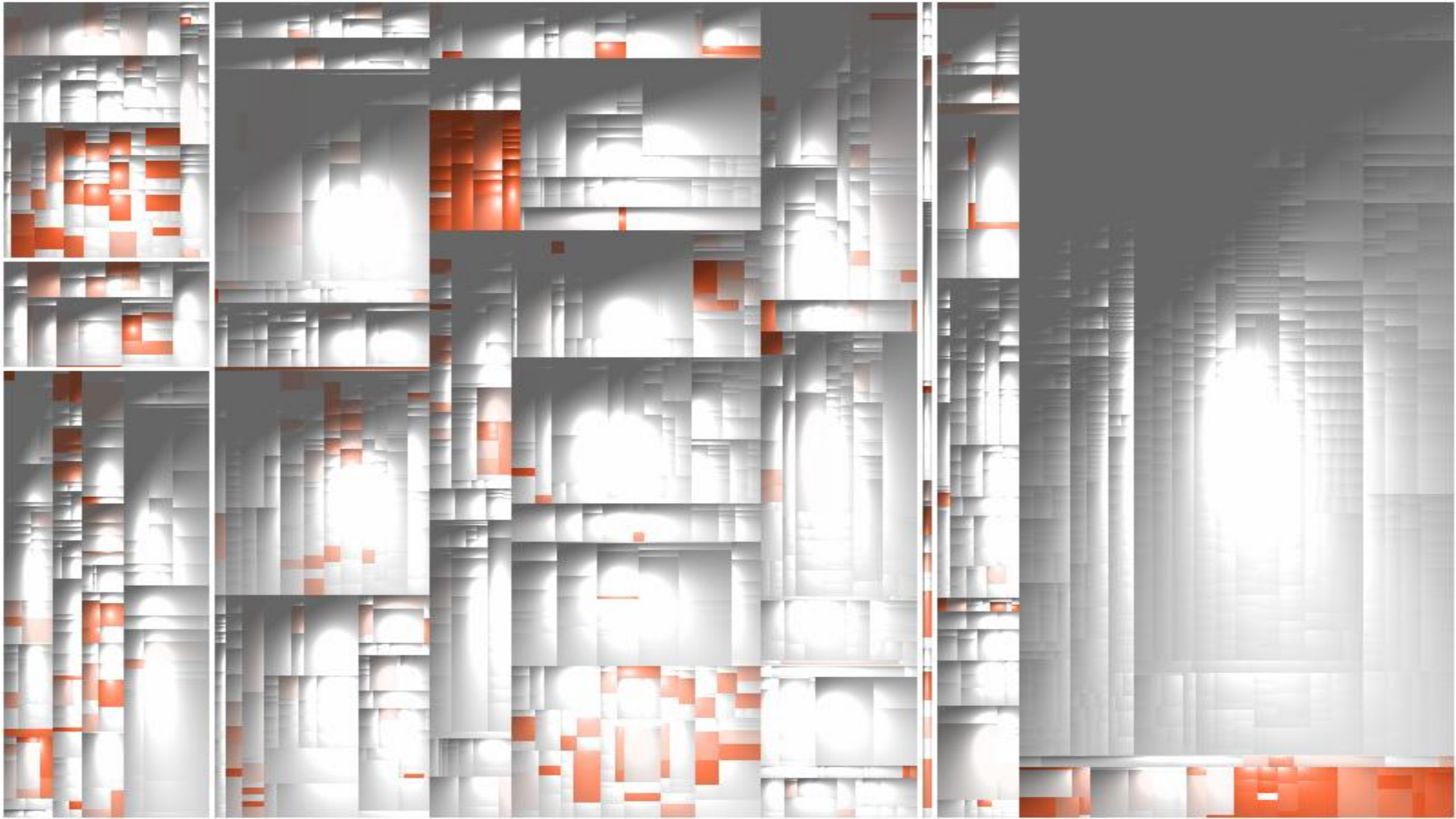
Product1

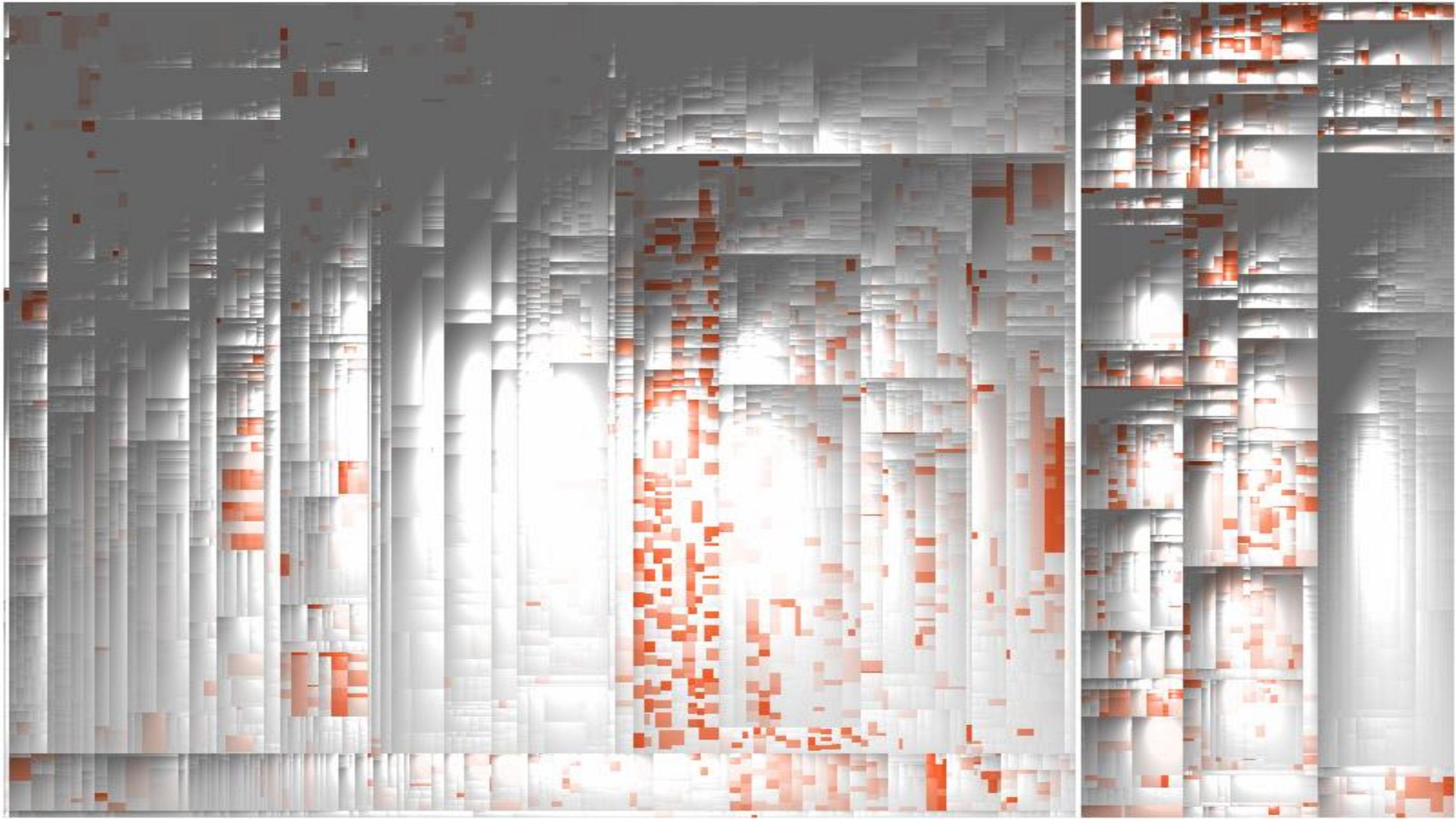
Product 2

Product 3

Product 4

Product 5





Was bringt Clone Management lanfgristig?

```java
// Utilities for arrays of elements
public String showElements(ModelElement[] elements, String nomsg) {
   boolean found = false;
   StringBuffer res = new StringBuffer();
   if (elements != null) {
      Index.getInstance().setCurrentRenderer(
            FlatReferenceRenderer.getInstance());
      for (int i = 0; i < elements.length; i++) {
         ModelElement el = elements[i];
         res.append(showElementLink(el)).append(HTML.LINE_BREAK);
         found = true;
      }
      Index.getInstance().resetCurrentRenderer();
   }
   if (!found && nomsg != null && nomsg.length() > 0) {
      res.append(HTML.italics(nomsg));
   }
   return res.toString();
}
```

```java
// Utilities for arrays of elements
public String showElements(ModelElement[] elements, String nomsg) {
   boolean found = false;
   StringBuffer res = new StringBuffer();
   if (elements != null) {
      Index.getInstance().setCurrentRenderer(
            FlatReferenceRenderer.getInstance());
      for (int i = 0; i < elements.length; i++) {
         ModelElement el = elements[i];
         res.append(showElementLink(el)).append(HTML.LINE_BREAK);
         found = true;
      }
      Index.getInstance().resetCurrentRenderer();
   }
   if (!found && nomsg.length() > 0) {
      res.append(HTML.italics(nomsg));
   }
   return res.toString();
}
```

Studie

Munich Re LV 1871

- Über 100 Fehler in produktiver Software

17	44	46
Kritisch	Nutzersichtbar	Nicht nutzersichtbar

- 52% aller ungewollten Unterschiede fehlerhaft

Juergens, Deissenboeck et al. *Do Code Clones Matter?* ICSE 2009

$$\text{Anzahl } \frac{Fehler}{Jahr} \times \text{Fehlerfolgekosten } \frac{PT}{Fehler}$$

$$\text{Anzahl } \frac{Fehler}{Jahr} \times \text{Fehlerfolgekosten } \frac{PT}{Fehler}$$

#Fehler durch inkonsistente Klone @ Munich Re

Daten aus Studie

- 3 Systeme von Munich Re analysiert
- 79 Fehler gefunden (Impact auf Funktionalität, nicht nur Wartbarkeit o.ä.)
- System waren produktiv, einzelne Fehler schon durch Anwender als Tickets reportet
- 1 Produktionsfehler durch inkonsistente Klone / 17k SLOC

Bedeutung heute

- Betrachtetes Portfolio der Munich Re umfasst ca. 8,25 Millionen SLOC
- Konservative Annahme: Clone Management spart 1 Produktionsfehler pro 50k SLOC pro Jahr
- 8,25 Millionen SLOC / 50k = 165

$$\text{Anzahl } \frac{Fehler}{Jahr} \times \text{Fehlerfolgekosten } \frac{PT}{Fehler}$$

$$165\ \frac{Fehler}{Jahr} \times \text{Fehlerfolgekosten}\ \frac{PT}{Fehler}$$

$$165\ \frac{Fehler}{Jahr} \times \textbf{Fehlerfolgekosten}\ \frac{PT}{Fehler}$$

ØFehlerfolgekosten von Fehlern in Produktion

Mögliche Auswirkungen fehlerhafter Software

- Nutzer bekommen falsche Ergebnisse
- Anwendung stürzt ab
- Daten gehen verloren
- Frustration bei Nutzern (Kunden und Mitarbeiter)

? PT

Aufwand für Reparatur

- Nutzer schreibt Ticket für Fehler
- Debugging (Nachstellen, Diagnose, …)
- Fixing
- Test
- Ggf. Deployment

? PT

ØFehlerfolgekosten von Fehlern in Produktion

Mögliche Auswirkungen fehlerhafter Software

- Nutzer bekommen falsche Ergebnisse
- Anwendung stürzt ab
- Daten gehen verloren
- Frustration bei Nutzern (Kunden und Mitarbeiter)

0 PT: bewusste Unterschätzung

Aufwand für Reparatur

- Nutzer schreibt Ticket für Fehler
- Debugging (Nachstellen, Diagnose, …)
- Fixing
- Test
- Ggf. Deployment

3 PT

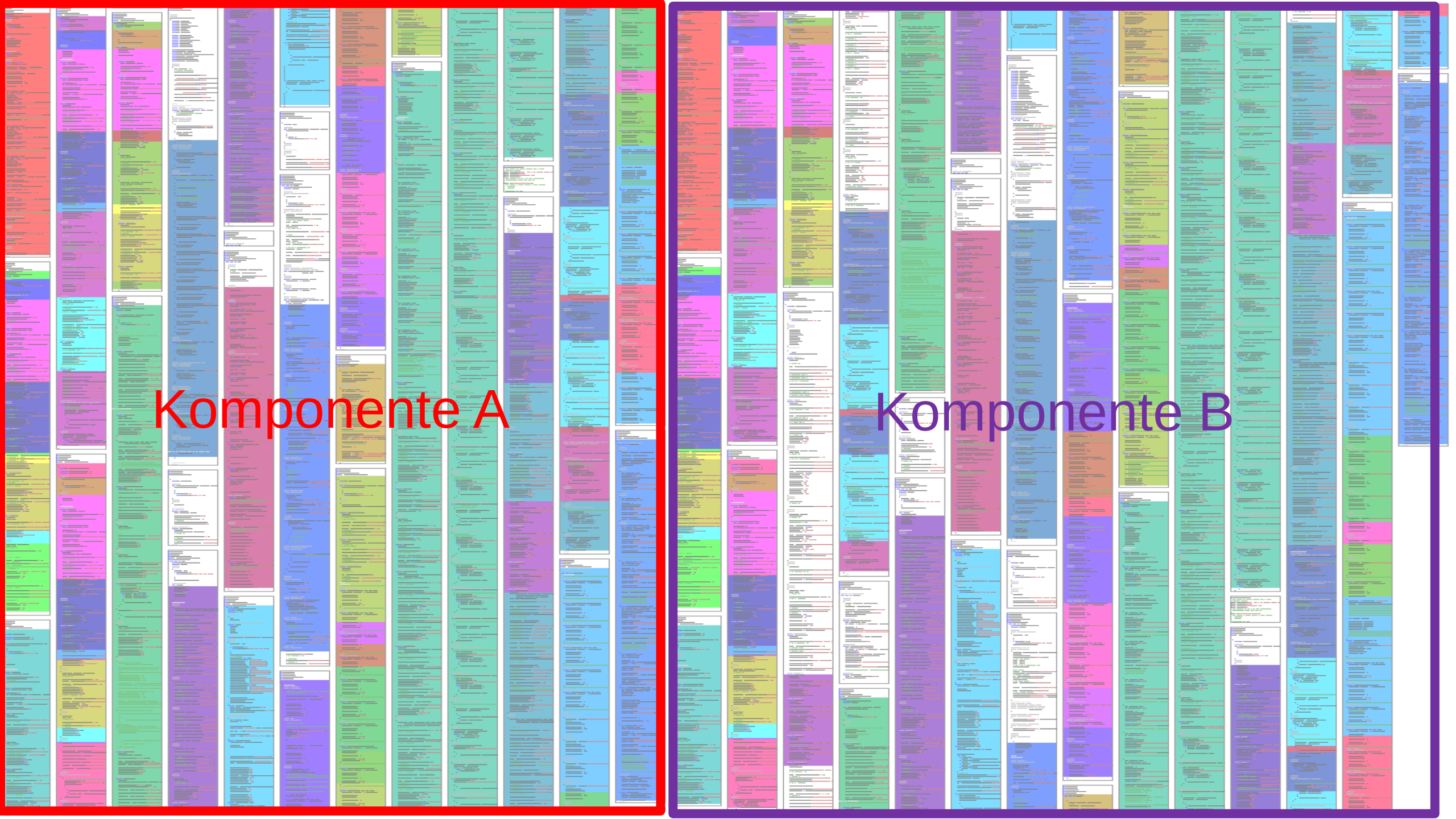
$$165\ \frac{Fehler}{Jahr} \times \textbf{Fehlerfolgekosten}\ \frac{PT}{Fehler}$$

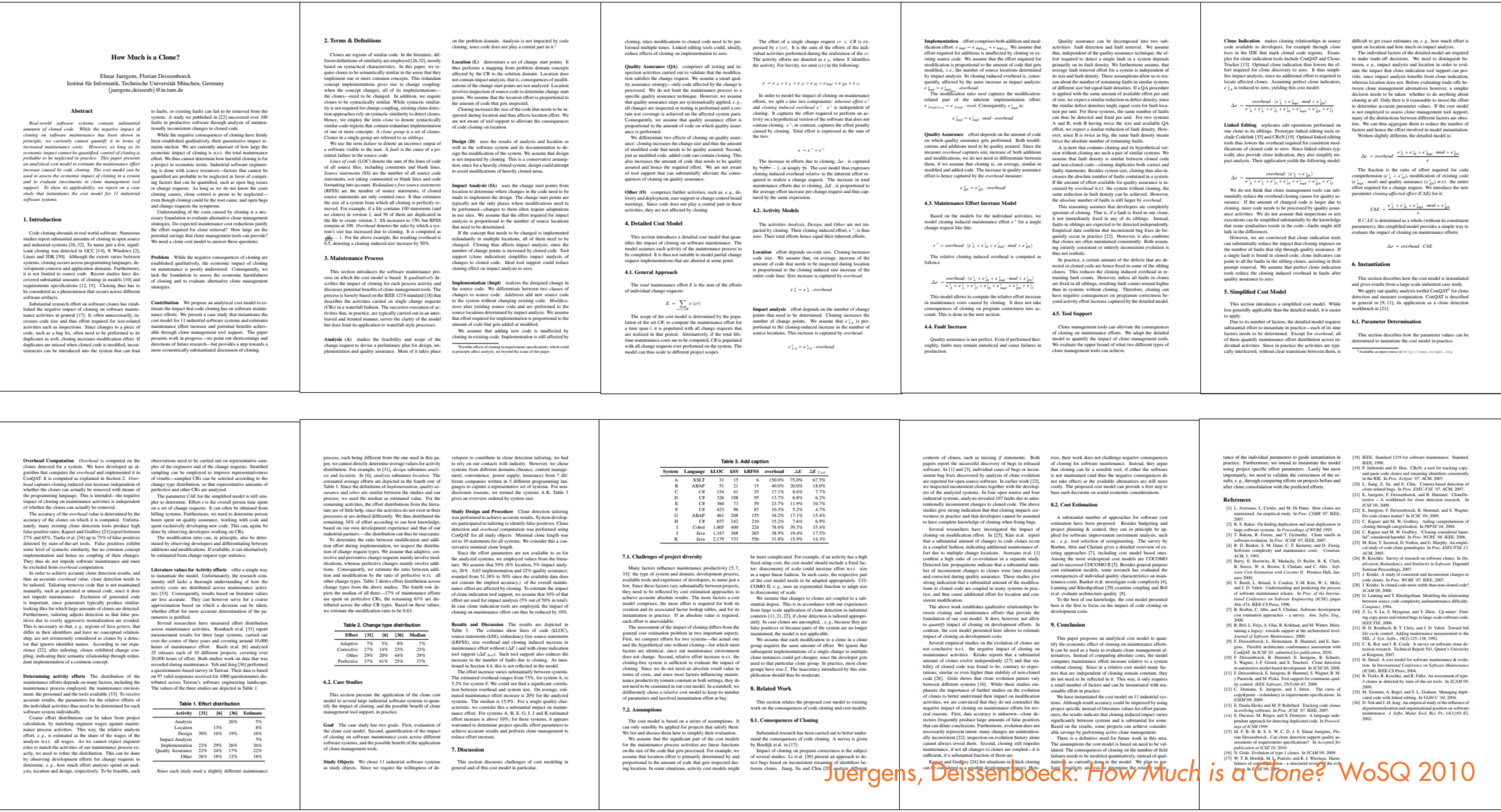
$$165\ \frac{Fehler}{Jahr} \times 3\ \frac{PT}{Fehler}$$

$$495 \frac{PT}{Jahr}$$

$$500 \frac{PT}{Jahr}$$

Munich Re spart durch Einsatz von Clone Management jährlich ca. 500 PT Aufwand für Fehlerbehebung


Komponente A
Komponente B



Juergens, Deissenboeck: *How Much is a Clone?* WoSQ 2010

$$\Delta\text{Aufwand} = \%\text{BlowUp} \times \%\text{CloneAffectedEffort}$$

$$\Delta\text{Aufwand} = \textbf{\%BlowUp} \times \%\text{CloneAffectedEffort}$$

Blow-Up: 0%

20 Lines

20 Lines

Blow-Up: 50%

20 Lines

20 Lines

20 Lines

$$\Delta\text{Aufwand} = \%\mathbf{12} \times \%\text{CloneAffectedEffort}$$

$$\Delta\text{Aufwand} = \%12 \times \textbf{\%CloneAffectedEffort}$$

%CloneAffectedEffort

Aktivitäten	Aufwändiger durch Cloning
Analysis	-
Location	Location
Design	-
Impact Analysis	Impact Analysis
Implementation	Implementation
Quality Assurance	Quality Assurance
Other	-

Detaillierte Herleitung und Berechnung im Paper.

Wert für Berechnung: **50%**.

$$\Delta \text{Aufwand} = \%12 \times \mathbf{\%50}$$

$$\Delta \text{Aufwand} = \%12 \times \%50 = \mathbf{6}\%$$

Die Munich Re setzt Clone Detection seit ca. 10 Jahren ein.

Wie sähe es ohne aus?

Continuous Software Quality Control in Practice

Daniela Steidl*, Florian Deissenboeck*, Martin Poehlmann*, Robert Heinke†, Bärbel Uhink-Mergenthaler†
* CQSE GmbH, Garching b. München, Germany
† Munich RE, München, Germany

***Abstract*—Many companies struggle with unexpectedly high maintenance costs for their software development which are often caused by insufficient code quality. Although companies often use static analyses tools, they do not derive consequences from the metric results and, hence, the code quality does not actually improve. We provide an experience report of the quality consulting company CQSE, and show how code quality can be improved in practice: we revise our former expectations on quality control from [1] and propose an enhanced continuous quality control process which requires the combination of metrics, manual action, and a close cooperation between quality engineers, developers, and managers. We show the applicability of our approach with a case study on 41 systems of Munich RE and demonstrate its impact.**

I. Introduction

Software systems evolve over time and are often maintained for decades. Without effective counter measures, the quality of software systems gradually decays [2], [3] and maintenance costs increase. To avoid quality decay, *continuous quality control* is necessary during development and later maintenance [1]: for us, quality control comprises all activities to monitor the system's current quality status and to ensure that the quality meets the quality goal (defined by the principal who outsourced the software development or the development team itself).

Research has proposed various metrics to assess software quality, including structural metrics[1] or code duplication, and has led to a massive development of analysis tools [4]. Much of current research focuses on better metrics and better tools [1], and mature tools such as ConQAT [5], Teamscale [6], or Sonar[2] have been available for several years.

In [1], we briefly illustrated how tools should be combined with manual reviews to improve software quality continuously, see Figure 1: We perceived quality control as a simple, continuous feedback loop in which metric results and manual reviews are used to assess software quality. A quality engineer – a representative of the quality control group – provides feedback to the developers based on the differences between the current and the desired quality. However, we underestimated the amount of required manual action to create an impact. Within five years of experience as software quality consultants in different domains (insurance companies, automotive manufacturers, or engineering companies), we frequently experienced that tool

This work was partially funded by the German Federal Ministry of Education and Research (BMBF), grant EvoCon, 01IS12034A. The responsibility for this article lies with the authors.

[1] e. g., file size, method length, or nesting depth
[2] http://www.sonarqube.org/

Fig. 1. The former understanding of a quality control process

support alone is not sufficient for successful quality control in practice. We have seen that most companies cannot create an impact on their code quality although they employ tools for quality measurements because the pressure to implement new features does not allow time for quality assurance: often, newly introduced tools get attention only for a short period of time, and are then forgotten. Based on our experience, quality control requires actions beyond tool support.

In this paper, we revise our view on quality control from [1] and propose an enhanced quality control process. The enhanced process combines automatic static analyses with a significantly larger amount of manual action than previously assumed to be necessary: Metrics constitute the basis but quality engineers must manually interpret metric results within their context and turn them into actionable refactoring tasks for the developers. We demonstrate the success and practicability of our process with a running case study with Munich RE which contains 32 .NET and 9 SAP systems.

II. Terms And Definitions

- A *quality criterion* comprises a metric and a threshold to evaluate the metric. A criterion can be, *e. g.*, to have a clone coverage below 10% or to have at most 30% code in long methods (*e. g.*, methods with more than 40 LoC).
- *(Quality) Findings* result from a violation of a metric threshold (*e. g.*, a long method) or from the result of a static code analysis (*e. g.*, a code clone).
- *Quality goals* describe the abstract goal of the process and provide a strategy how to deal with new and existing findings during further development: The highest goal is to have no findings at all, *i. e.*, all findings must be removed immediately. Another goal is to avoid new findings, *i. e.*, existing findings are tolerated but new findings must not be introduced. (III-B will provide more information).

III. The Enhanced Quality Control Process

Our quality control process is designed to be *transparent* (all stakeholders involved agree on the goal and consequences

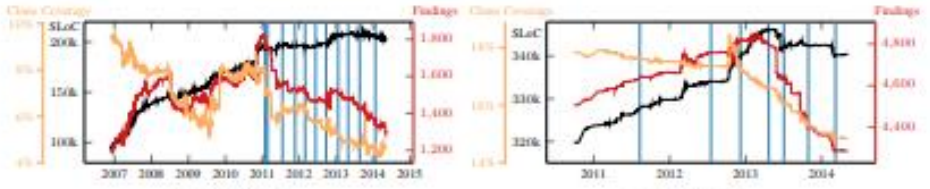


Fig. 3. System *A*

Fig. 5. System *C*

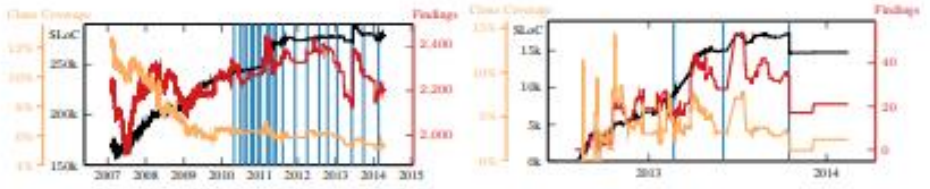


Fig. 4. System *B*

Fig. 6. System *D*

continuously in the available history (Figure 4). The number of findings, however, increases until mid 2012. In 2012, the project switched from QG2 to QG3. After this change, the number of findings decreases and the clone coverage settles around 6%, which is a success of the quality control. The major increase in the number of findings in 2013 is only due to an automated code refactoring introducing braces that led to threshold violations of few hundred methods. After this increase, the number of findings start decreasing again, showing the manual effort of the developers to remove findings.

For System C (Figure 5), the quality control process shows a significant impact after two years: Since the end of 2012, when the project also switched from QG2 to QG3, both the clone coverage and the overall number of findings decline. In the year before, the project transitioned between development teams and, hence, we only wrote two reports (July 2011 and July 2012).

System *D* (Figure 6) almost fulfills QG4 as after 1 year of development, it has only 21 findings in total and a clone coverage of 2.5%. Technically, under QG4, the system should have zero findings. However, in practice, exactly zero findings is not feasible as there are always some findings (*e. g.*, a long method to create UI objects or clones in test code) that are not a major threat to maintainability. Only a human can judge based on manual inspection of the findings whether a system still fulfills QG4, if it does not have exactly zero findings. In the case of System D, we consider 21 findings to be few and minor enough to fulfill QG4.

To summarize, our trends show that our process leads to actual measurable quality improvement. Those trends go beyond anecdotal evidence but are not sufficient to scientifically proof our method. However, Munich RE decided only recently to extend our quality control from the .NET area to all SAP development. As Munich RE develops mainly in the .NET and SAP area, most application development is now supported by quality control. The decision to extend the scope of quality control confirms that Munich Re is convinced by the benefit of quality control. Since the process has been established, maintainability issues like code cloning are now an integral part of discussions among developers and management.

V. Conclusion

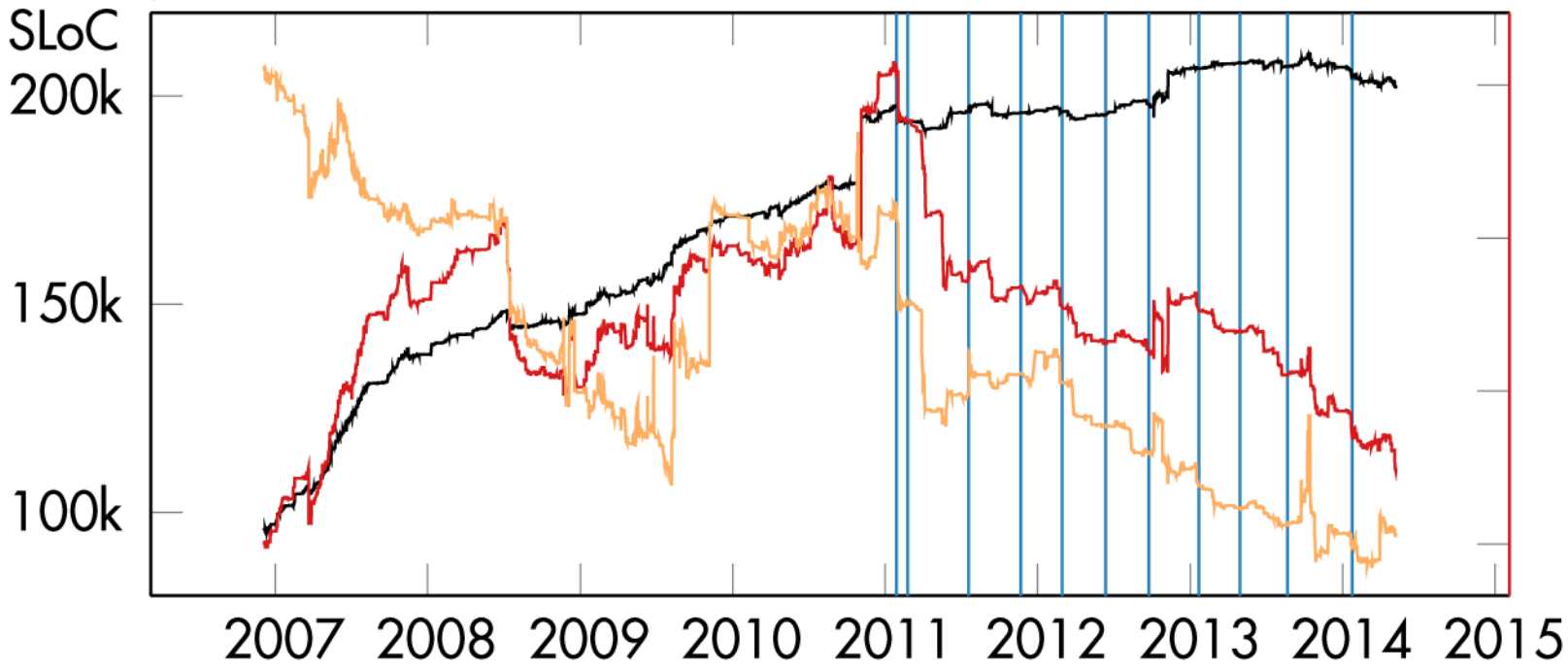
Quality analyses must not be solely based on automated measurements, but need to be combined with a significant amount of human evaluation and interaction. Based on our experience, we proposed a new quality control process for which we provided a running case study of 41 industry projects. With a qualitative impact analysis at Munich RE we showed measurable, long-term quality improvements. Our process has led to measurable quality improvement and an increased maintenance awareness up to management level at Munich Re.

References

[1] F. Deissenboeck, E. Juergens, B. Hummel, S. Wagner, B. M. y Parareda, and M. Pizka, "Tool support for continuous quality control," in *IEEE Software*, 2008.
[2] D. L. Parnas, "Software aging," in *ICSE '94*.
[3] S. G. Eick, T. L. Graves, A. F. Karr, J. S. Marron, and A. Mockus, "Does code decay? Assessing the evidence from change management data," *IEEE Trans. Software Eng.*, 2001.
[4] P. Johnson, "Requirement and design trade-offs in hackystat: An in-process software engineering measurement and analysis system," in *ESEM'07*.
[5] F. Deissenboeck, M. Pizka, and T. Seifert, "Tool support for continuous quality assessment," in *STEP'05*.
[6] L. Heinemann, B. Hummel, and D. Steidl, "Teamscale: Software quality control in real-time," in *ICSE'14*.
[7] R. C. Martin, *Clean Code: A Handbook of Agile Software Craftsmanship*, 2008.

Steidl, Heinke, Uhink: Continuous Software Quality Control in Practice? *ICSME* 2014

Einsparung durch Clone Detection



Menge an geklontem Code hat sich seit der Einführung von Clone Detection halbiert.
Ohne Clone Detection wäre der Clone Blow-Up daher vorr. doppelt so groß.

Ersparnis Aufwand = **6%**

Munich Re spart durch Einsatz von Clone Detection jährlich 6% Aufwand durch vermiedene Redundanz ein.

Findings

All	**6793**
Architecture	**1**
Architecture Conformance	1
Code Anomalies	**1344**
Bad practice	971
Correctness	2
Exception Handling	62
General checks (built-in)	120
Null pointer dereference	13
Performance	36
Unused code	93
Unused variable or parameter	47
Code Duplication	**988**
Cloning	101
Redundant Literals	887
Documentation	**3378**
Comment completeness	3236
Task tags	142
Formatting	**6**
Code formatting	6
Naming	**110**
Java naming conventions	110
Structure	**966**
File Size	38
Method Length	278
Nesting Depth	650

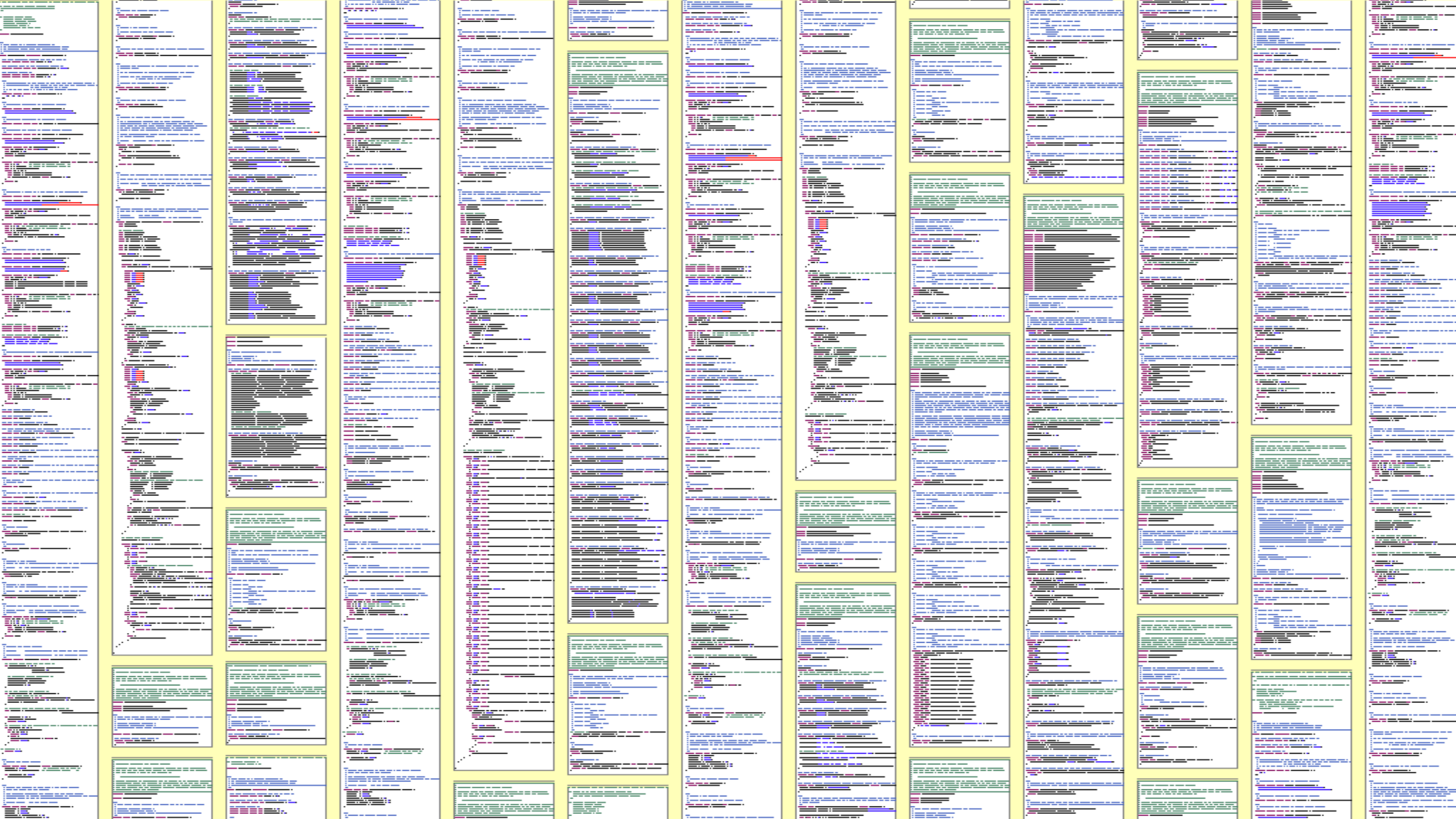
$$500 \frac{PT}{Jahr}$$

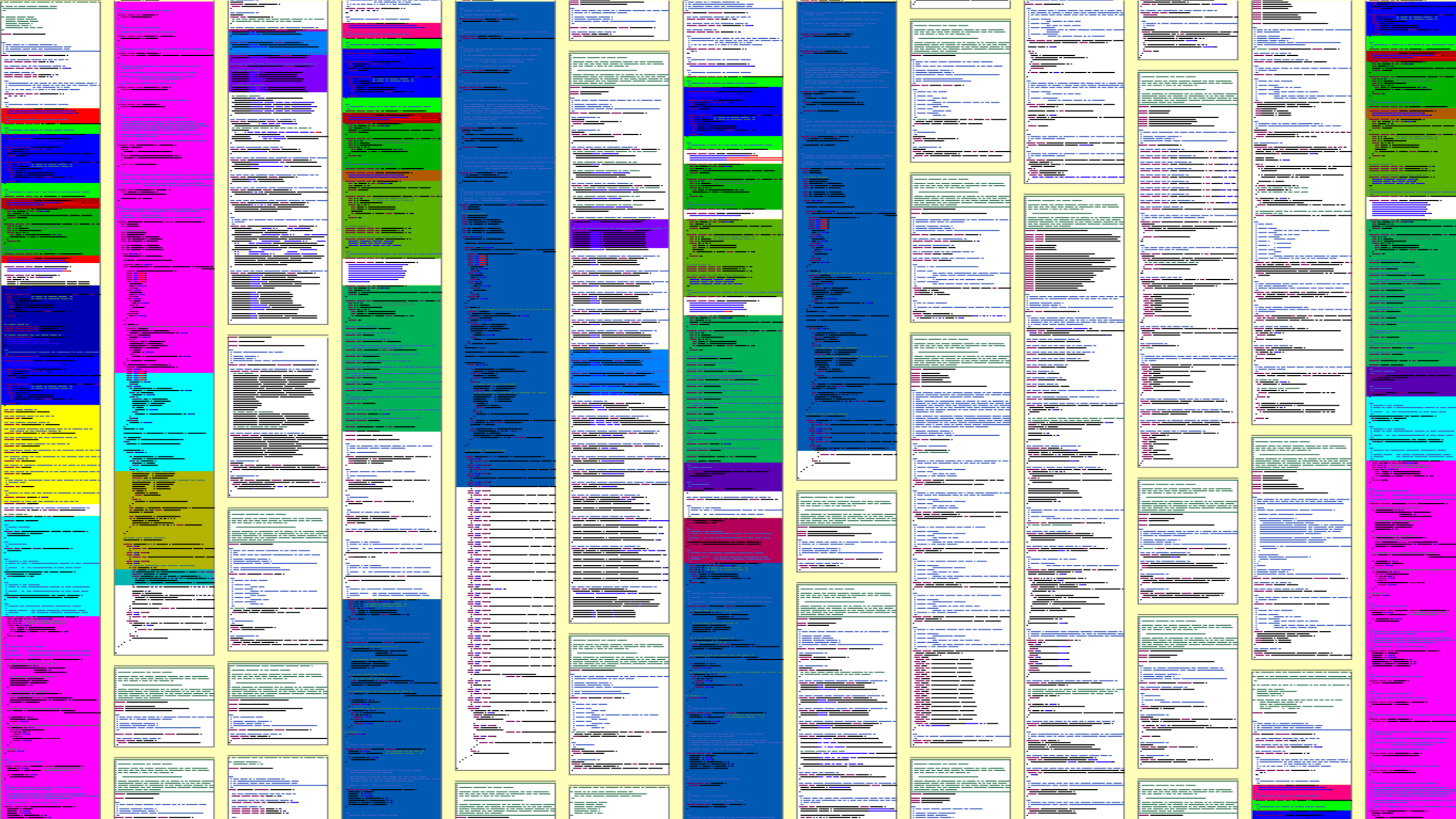
Munich Re spart durch Einsatz von Clone Management jährlich ca. 500 PT Aufwand für Fehlerbehebung

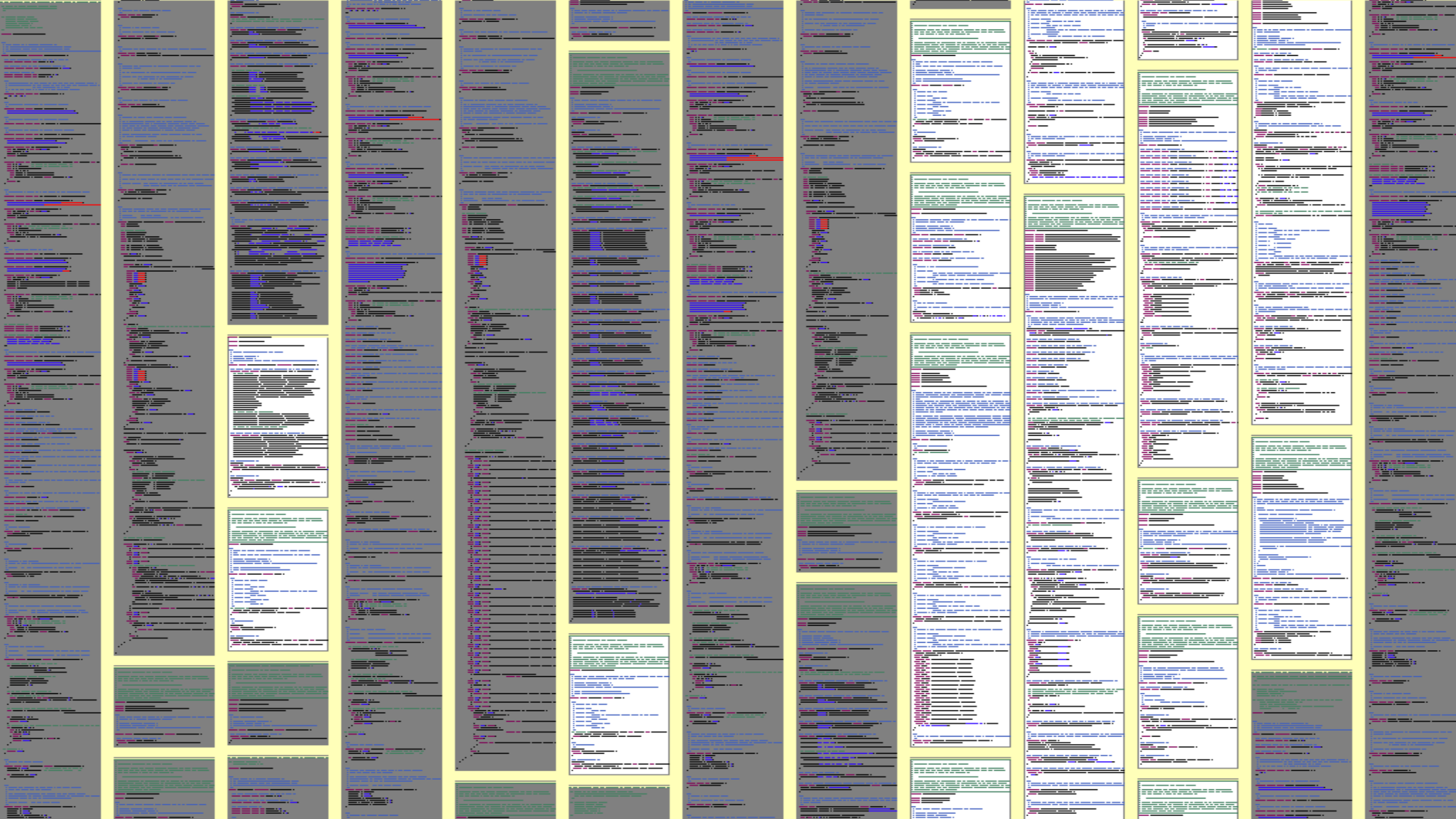
$$\text{Ersparnis Aufwand} = \mathbf{6}\%$$

Munich Re spart durch Einsatz von Clone Detection jährlich 6% Aufwand durch vermiedene Redundanz ein.

Wie ausprobieren?







CQSE Workshop

Clone Management

Copy, Paste und dann …?

09. November
10:30-12:00 Uhr
cqse.eu/clone-22-11-saat

Wie zukunftssicher ist Ihr Softwaresystem?

Unser Vorgehen und Erfahrungen aus 10 Jahren Softwareaudits

10. November
10:30-12:00 Uhr
cqse.eu/audit-22-11-saat

Kontakt – Freue mich auf Diskussionen ☺

Dr. Elmar Jürgens · juergens@cqse.eu · +49 179 675 3863

CQSE GmbH
Lichtenbergstraße 8
85748 Garching bei München
www.cqse.eu

CQSE