

# Copy & Paste & Bug

Dr. Elmar Jürgens

# Über Mich

## Forschung

- Clone Detection
- Architekturanalyse

## Beratung

- Mitgründer
- Qualitäts-Bewertung & Qualitäts-Controlling

## Entwicklung

- Continuous Quality Assessment Toolkit ConQAT
- >300 kLOC, Apache Lizenz, >20.000 Downloads



```
// 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();
}
```

```
// 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();
}
```

A man with a beard and a striped jacket, smiling and holding a small object. He is wearing a lanyard with a badge around his neck. The background is a plain wall.

# #1 Code Smell (1999)



1975

```
// 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();
}
```

```
// 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();
}
```

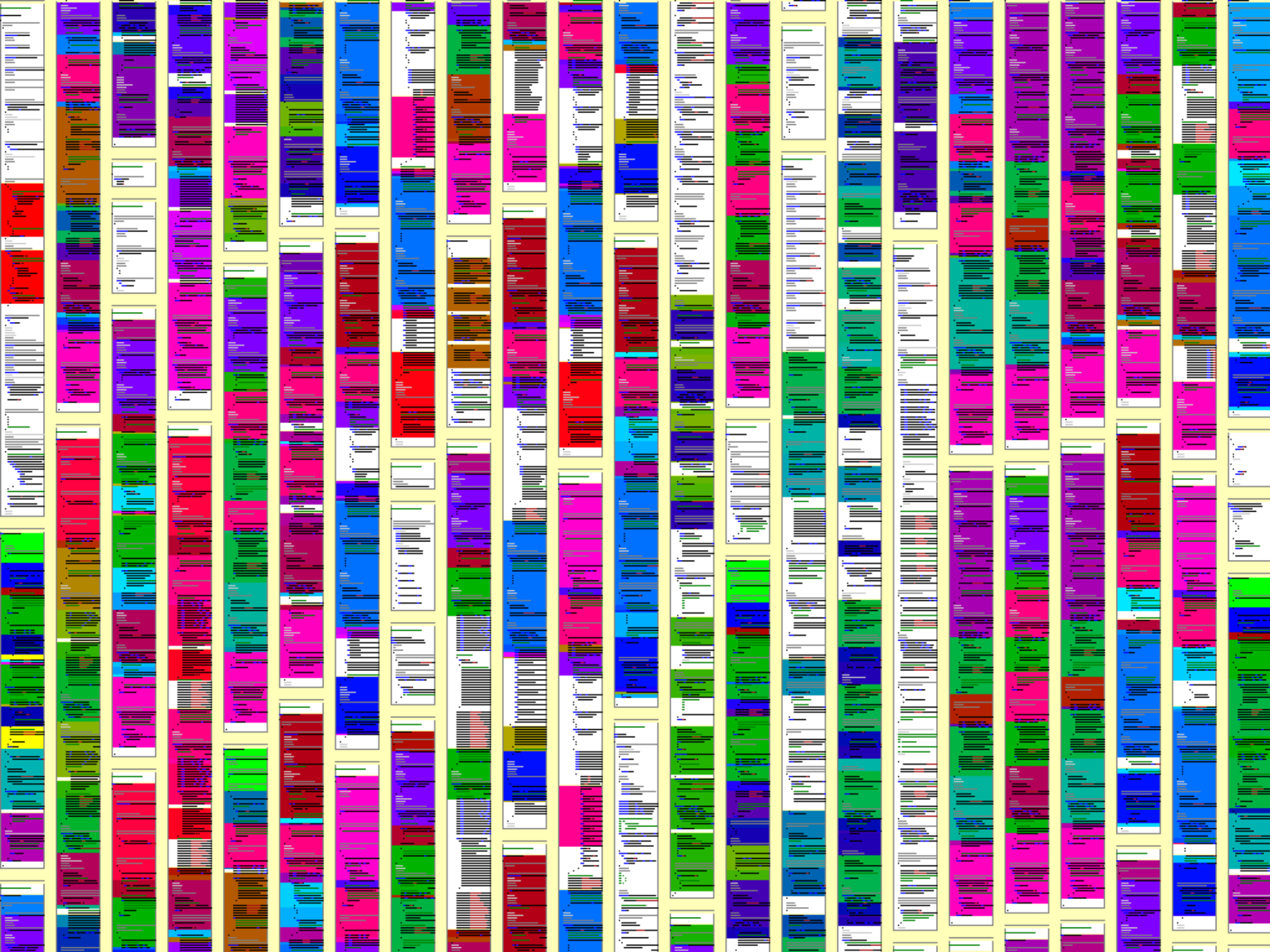
```
// Utilities for arrays of elements
public String showElements(ModelElement[] elements, String nomsg) {
    boolean found = false;
    StringBuffer res = new StringBuffer();
    if (elements != null) {
        Index.getInstance().setCurrentRenderer(
            FlatReferenceRenderer.getInstance());
        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();
}
```











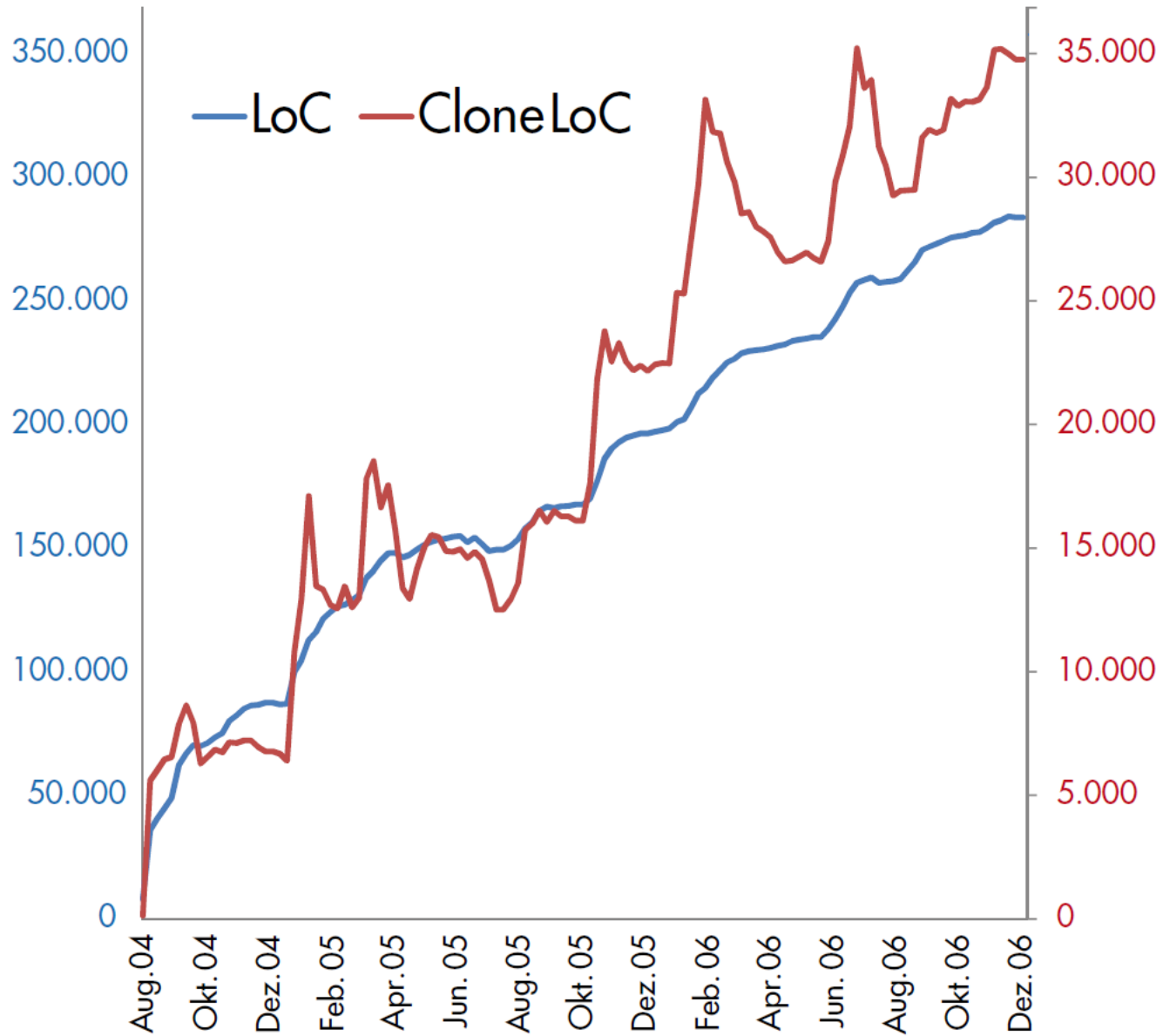
# Studie

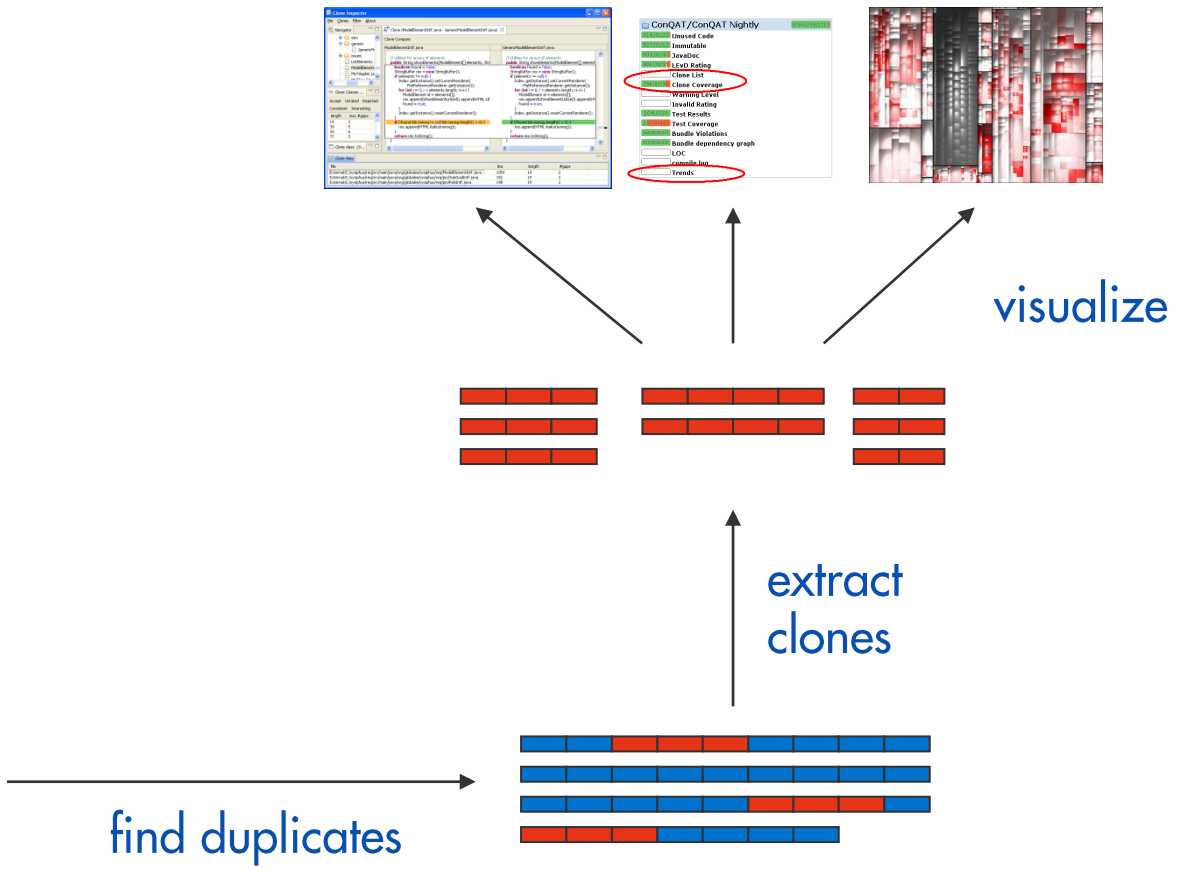
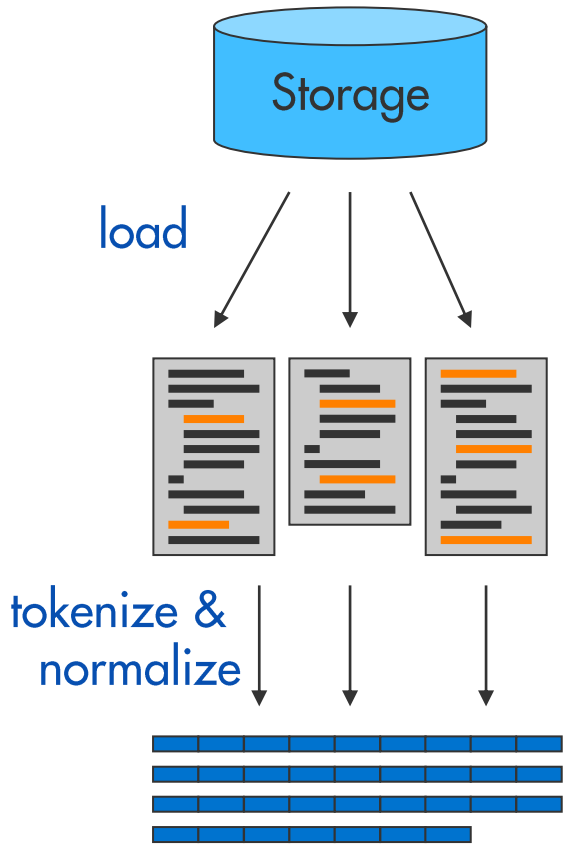


- Über 100 Fehler in produktiver Software



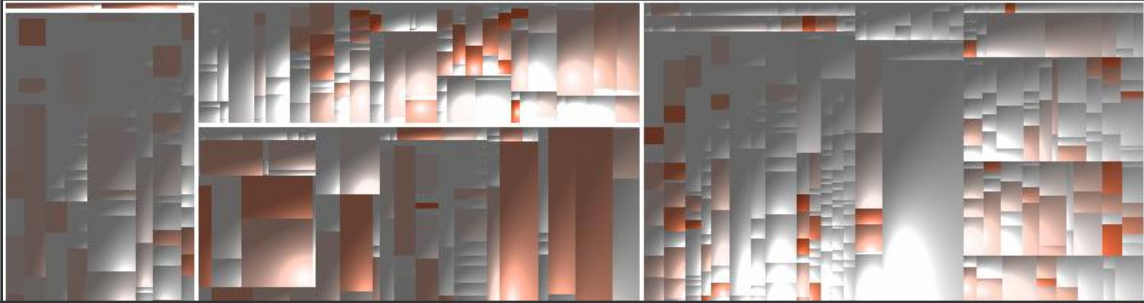
Juergens, Deissenboeck et al: *Do Code Clones Matter?* ICSE 2009





```
JabRef/java/net/sf/jabref/Util.java
if (fileParts.length > 1) {
    for (int i = 0; i < fileParts.length - 1; i++) {
        String dirToProcess = fileParts[i];
```

```
JabRef/java/net/sf/jabref/external/RegExpFileSearch.java
if (fileParts.length > 1) {
    for (int i = 0; i < fileParts.length - 1; i++) {
        String dirToProcess = fileParts[i];
        findBrackets(dirToProcess, entry,
            ("^.:$") { // Windows Drive Le
                (dirToProcess + "/");
            }
            ".") { // Stay in current dire
            }
            "..") {
                (directory
```



PROG_Y...	PROG_Y...	PROG_Y...

### Clone Coverage Trend



```
try {
    assertEquals(Double
        fail();
    } catch (AssertionFail
    )
}
```

Sibling 3: FloatAssertTest.java(13-30)  
 Focus on CloneClass: 12  
 Select this clone instance: DoublePrecisionAssertTest.java(13-30)











# Best Practice: Diskriminierung (von Code)

## Art der Wartung

- Manuell
- Generator
- Überhaupt nicht (Wegwerf-Prototyp)

## Aufgabe im Projekt

- Teil der Anwendung
- Test
- Hilfswerkzeug

Mit manuell gewartetem Applikationscode beginnen

JabRef/java/net/sf/jabref/Util.java

```
if (fileParts.length > 1) {
    for (int i = 0; i < fileParts.length - 1; i++) {
        String dirToProcess = fileParts[i];
        dirToProcess = expandBrackets(dirToProcess, entry, databas

        if (dirToProcess.matches("^.:$")) { // Windows Drive Lette
            directory = new File(dirToProcess + "/");
            continue;
        }
        if (dirToProcess.equals(".")) { // Stay in current directo
            continue;
        }
        if (dirToProcess.equals("..")) {
            directory = new File(directory.getParent());
            continue;
        }
        if (dirToProcess.equals("**")) { // Do for all direct subdi

            File[] subDirs = directory.listFiles();
            if (subDirs == null)
                return null; // No permission?

            String restOfFileString = join(fileParts, "/", i + 1,

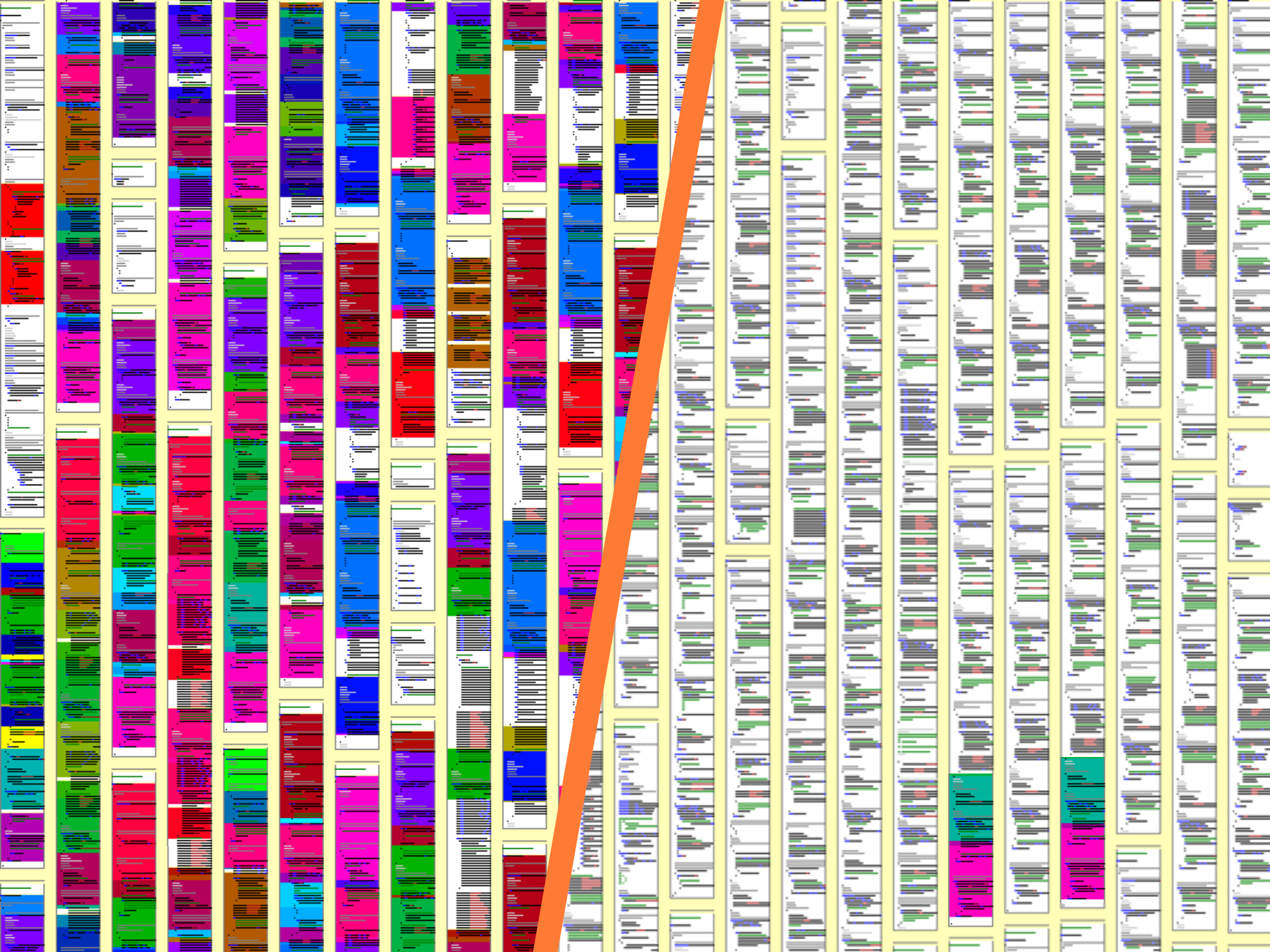
            for (int sub = 0; sub < subDirs.length; sub++) {
                if (subDirs[sub].isDirectory()) {
```

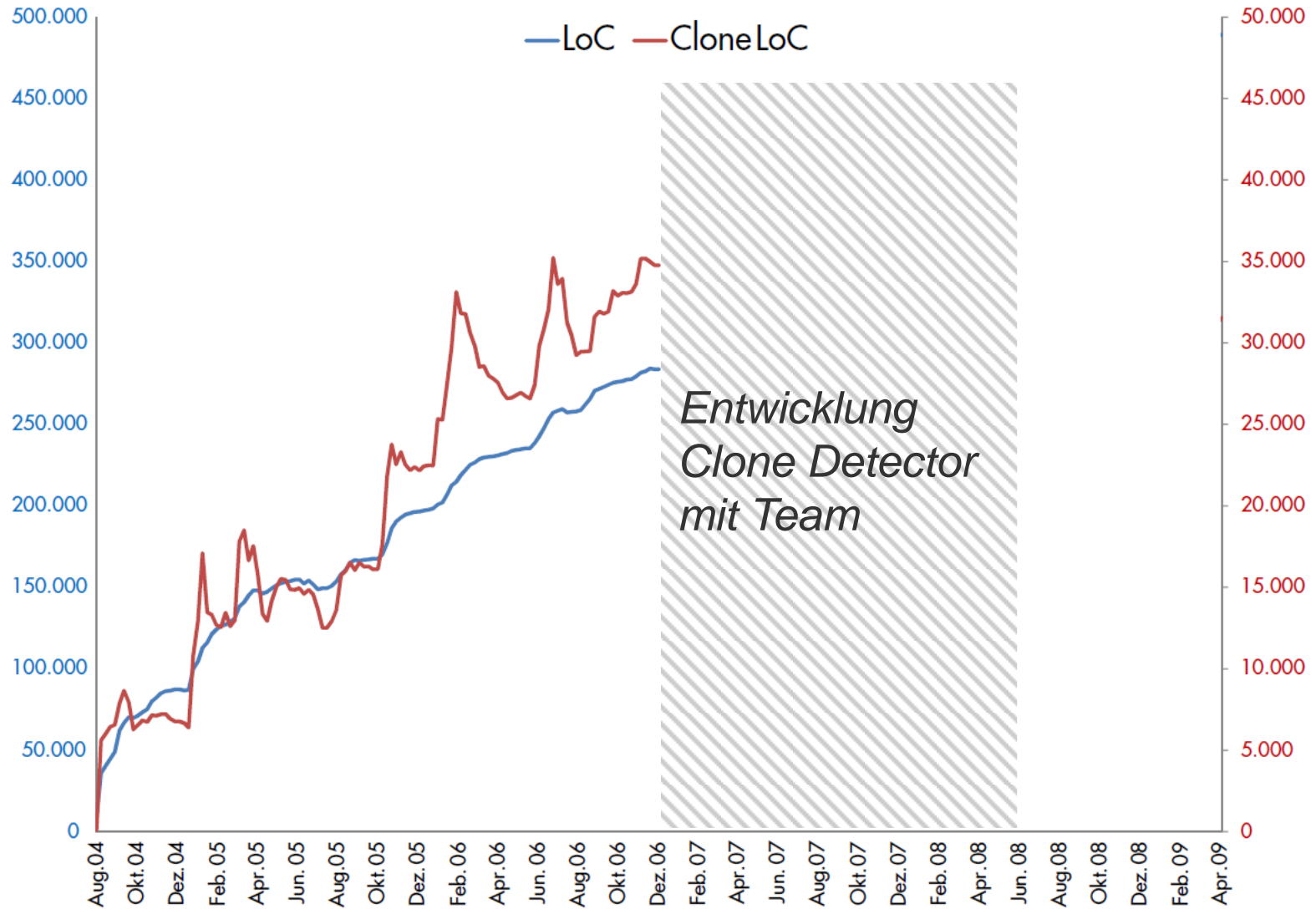
JabRef/java/net/sf/jabref/external/RegExpFileSearch.java

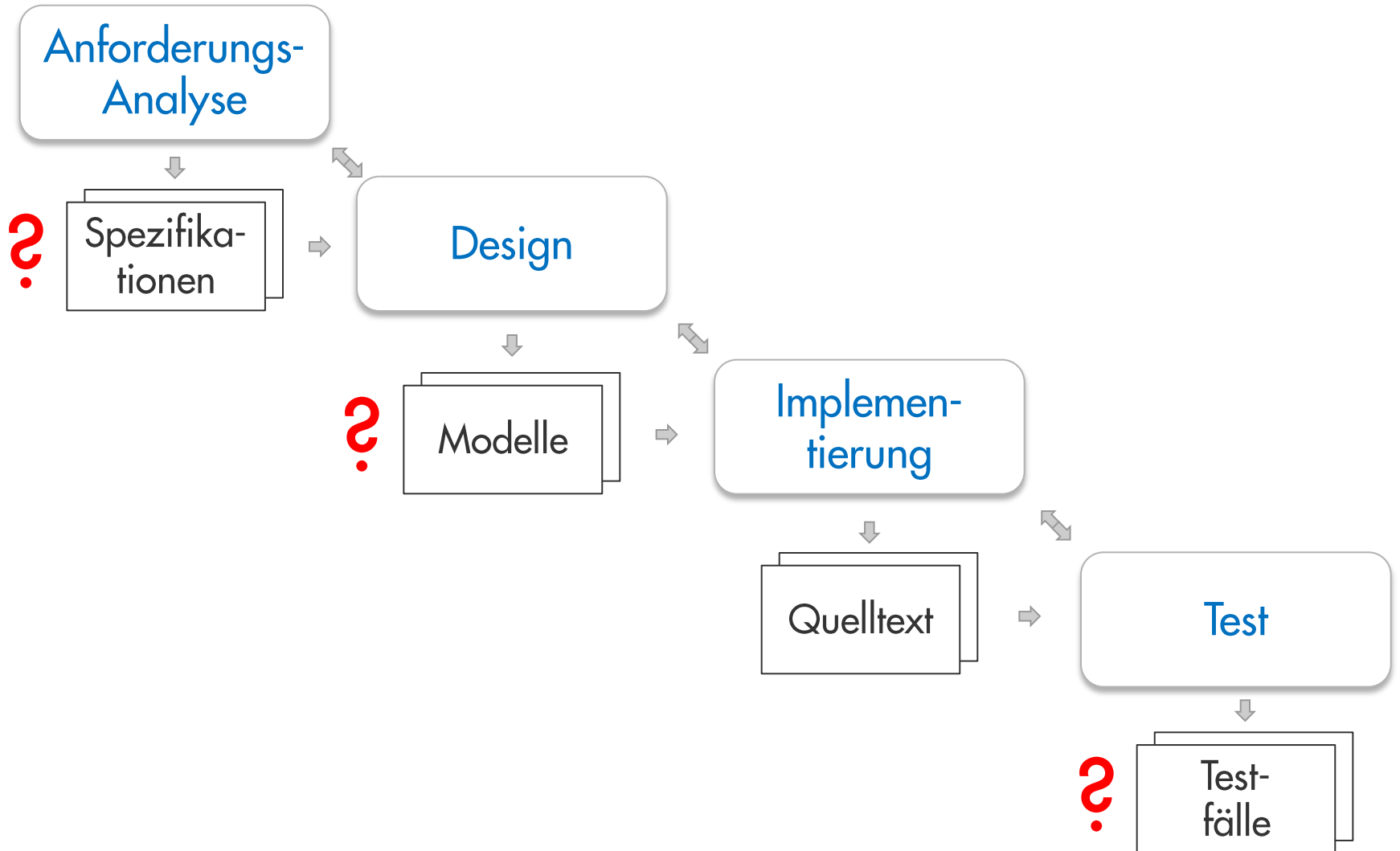
```
if (fileParts.length > 1) {
    for (int i = 0; i < fileParts.length - 1; i++) {
        String dirToProcess = fileParts[i];
        dirToProcess = Util.expandBrackets(dirToProcess, entry,

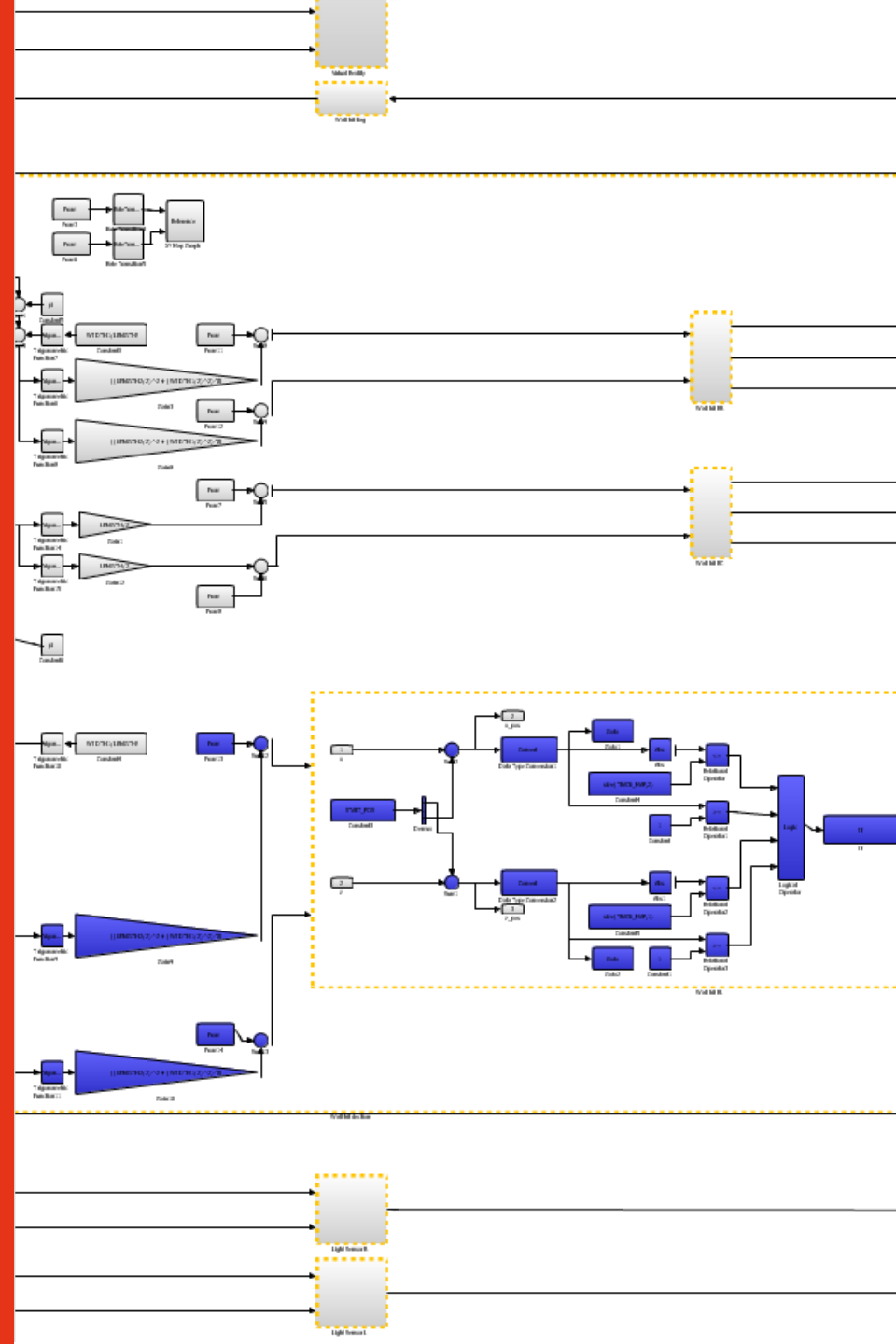
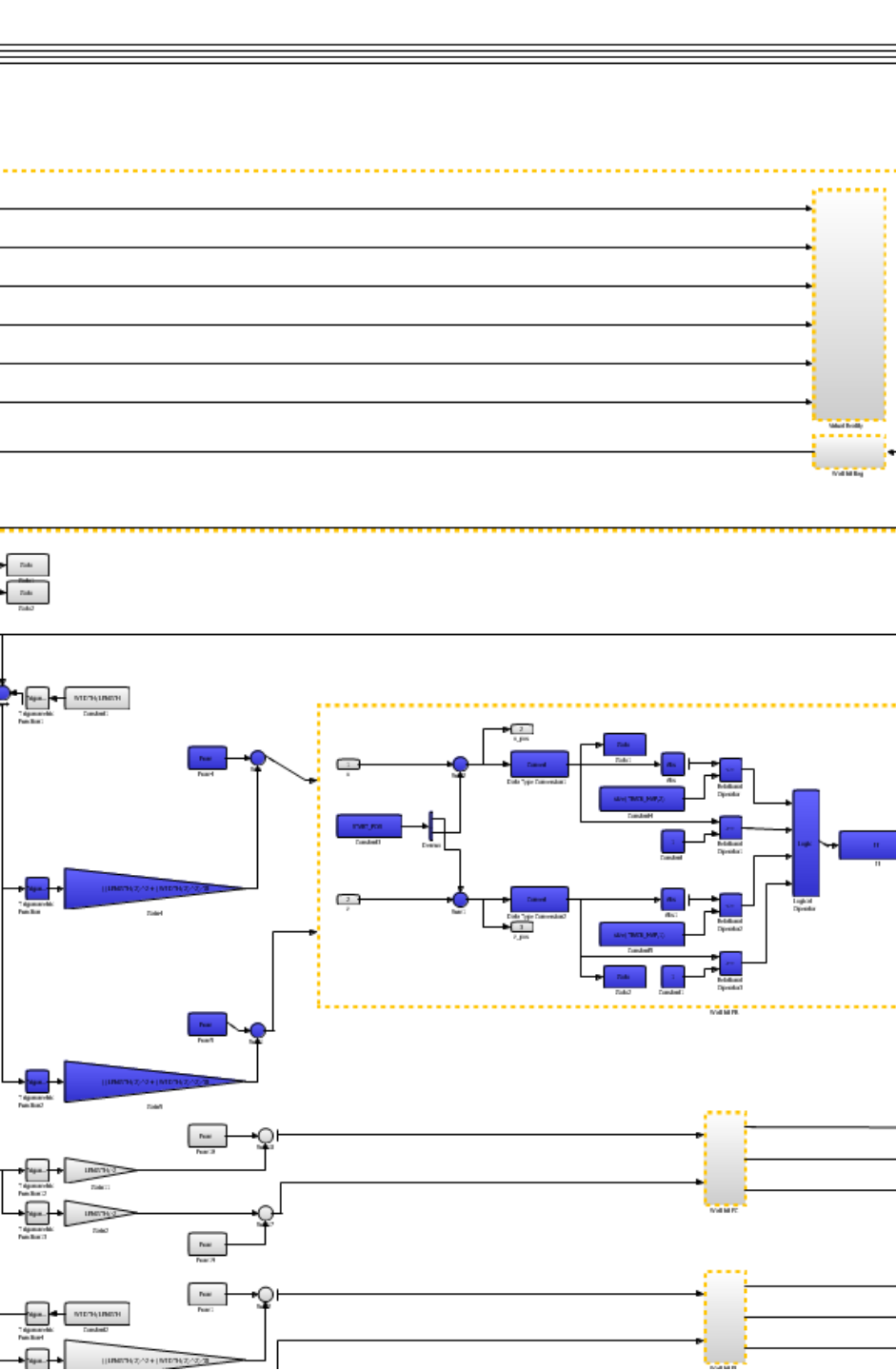
        if (dirToProcess.matches("^.:$")) { // Windows Drive Le
            directory = new File(dirToProcess + "/");
            continue;
        }
        if (dirToProcess.equals(".")) { // Stay in current dire
            continue;
        }
        if (dirToProcess.equals("..")) {
            directory = new File(directory.getParent());
            continue;
        }
        if (dirToProcess.equals("**")) { // Do for all direct su

            File[] subDirs = directory.listFiles();
            if (subDirs != null) {
                String restOfFileString = Util.join(fileParts,
                for (int sub = 0; sub < subDirs.length; sub++)
                    if (subDirs[sub].isDirectory()) {
                        res.addAll(findFile(entry, database, su
                        restOfFileString, extensionRegExp))
                    }
            }
        }
    }
}
```

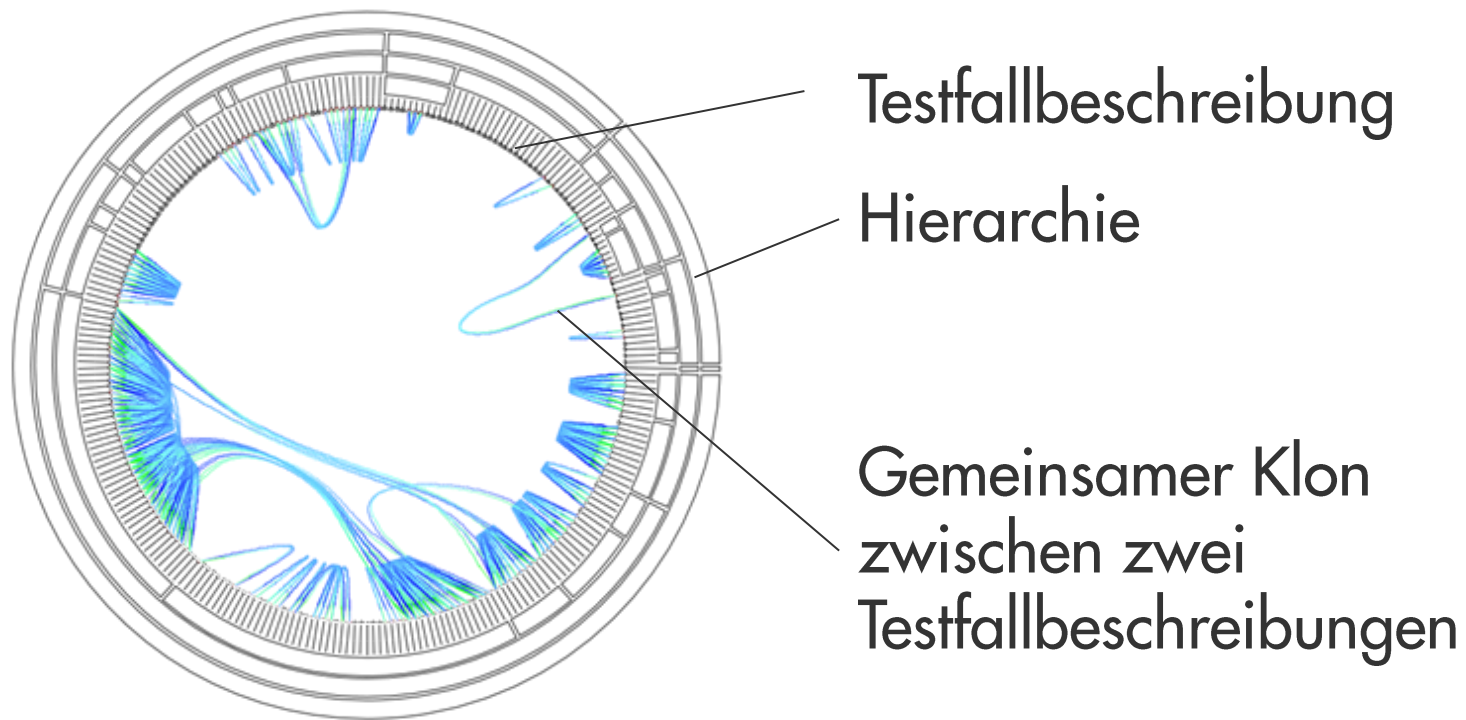


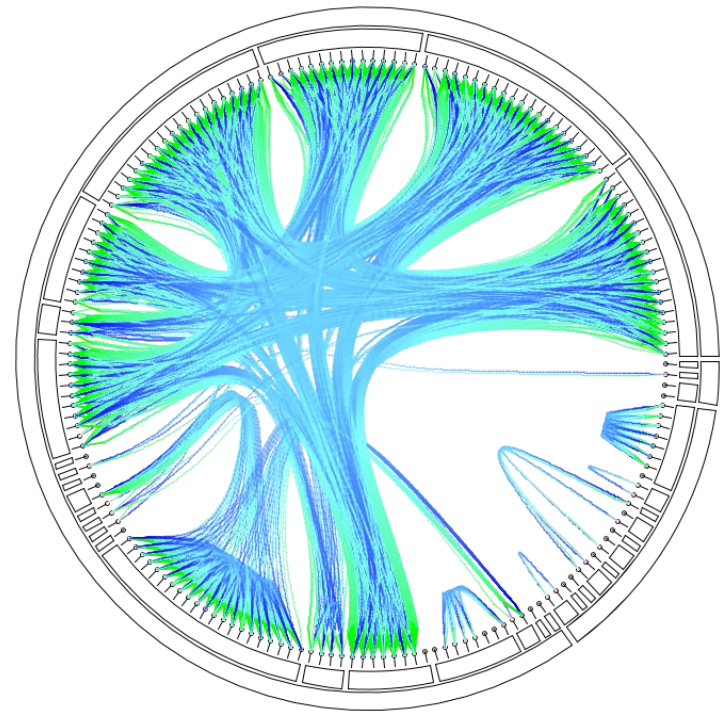
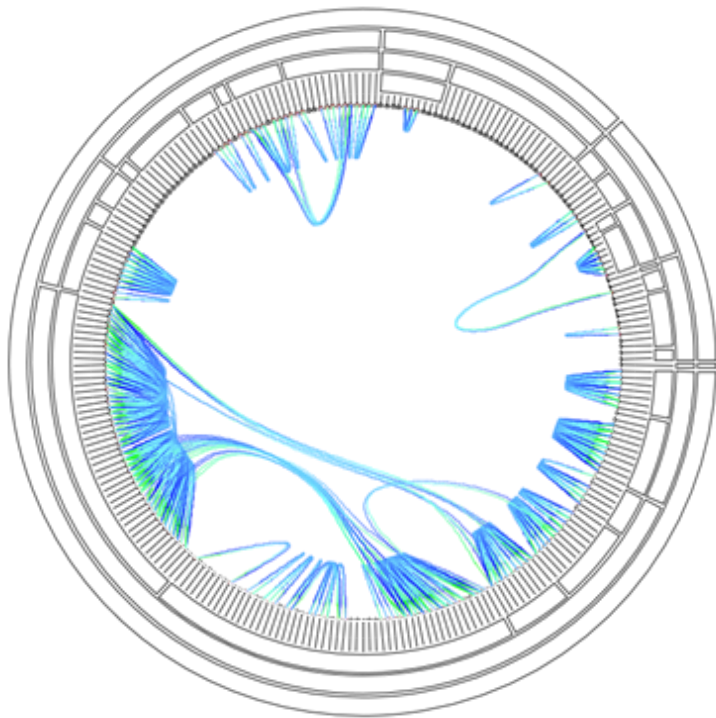


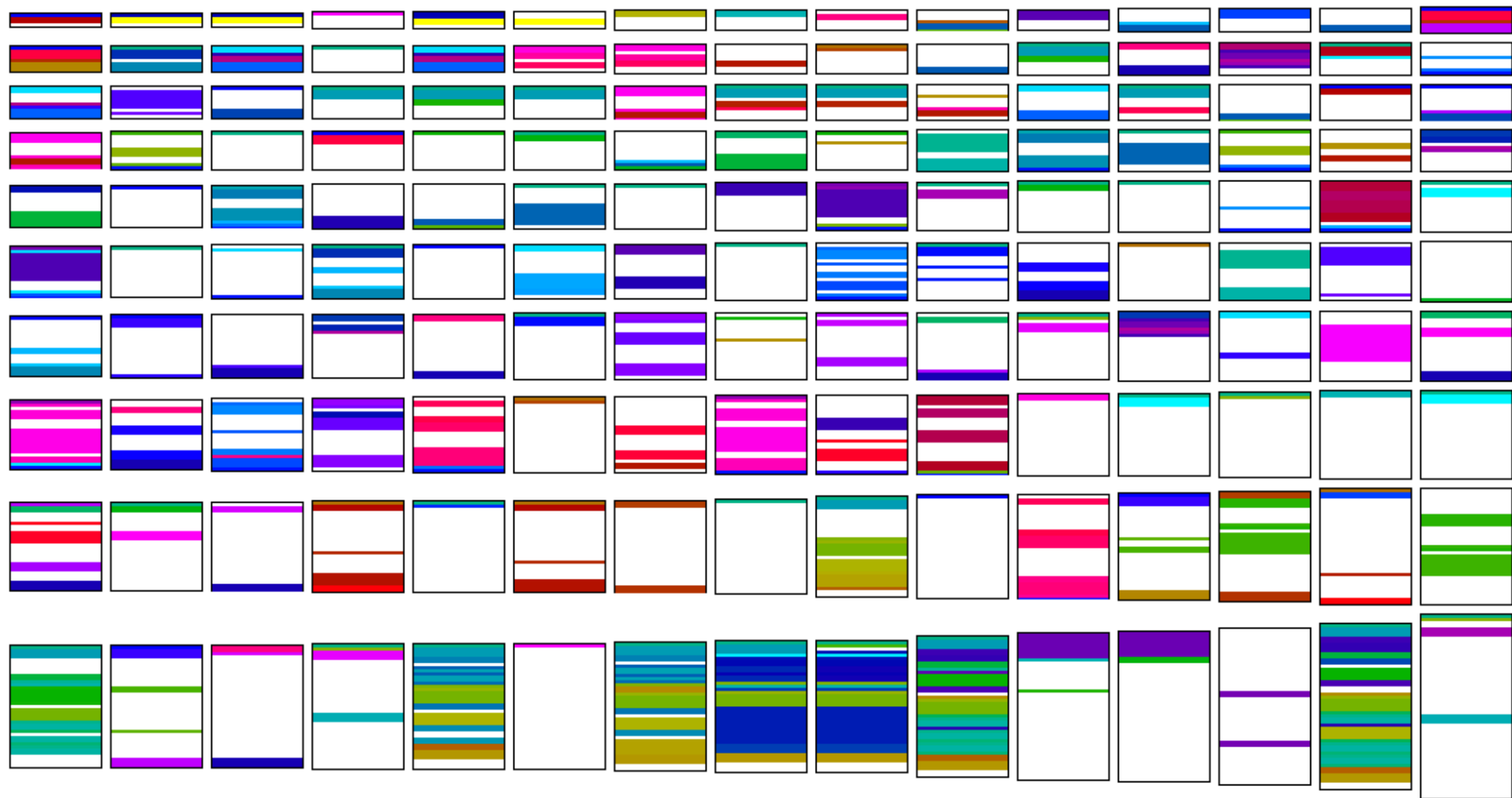












176 use cases in total, 150 contain cloning

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	-	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

For function class DynamicArray, the messages are defined as follows:

OPType	Parameter
Set	Tag, PosY, Data
Get	Tag, PosY
SetGet	Tag, PosY, Data
Increment	Tag, PosY, NSteps
Decrement	Tag, PosY, NSteps
GetInterface	
Status	Tag, PosY, Data
Interface	Refer to section
Error	ErrorCode, Error

Tag	Unsigned Word
PosY	Unsigned Byte

The Tag belongs to the data field denotes the Tag. With respect to in a DynamicArray indicates the included within the NMax counter

For the function class Map, the messages are defined as follows:

OPType	Parameter
Set	Tag, PosY, Data
Get	Tag, PosY
SetGet	Tag, PosY, Data
Increment	Tag, PosY, NSteps
Decrement	Tag, PosY, NSteps
GetInterface	
Status	Tag, PosY, {Data}
Interface	Refer to section 2.2.4.2.2 on page 80
Error	ErrorCode, ErrorInfo

Tag	Unsigned Word	=	0x0000	all lines
		<>	0x0000	one special line
PosY	Unsigned Byte	<>	0x00	one special column (only if Tag <> 0x0000)
		=	0x01	not allowed, no access to Tag

The Tag belongs to the data field and is returned at the start of every line. PosY = 0x01 denotes the Tag. With respect to consistency, accesses to a column are not reasonable. As with PosX, the value 0x0000 for Tag is reserved to indicate the whole Array. Because of the unordered nature of the function class Map, no last line with a Tag value of 0xFFFF needs to be stored. However, 0xFFFF is used to indicate the end of the transmission of the whole Array.



### Splitting with java.lang.String.split()

```
String[] addresses2 = addresses.split(Pattern.  
    quote(String.valueOf(separator)));
```

### Splitting with java.util.StringTokenizer

```
ArrayList<String> validEmails = new ArrayList<  
    String>();  
StringTokenizer st = new StringTokenizer(  
    addresses, Character.toString(separator));  
while (st.hasMoreTokens()) {  
    String tmp = st.nextToken();  
    validEmails.add(tmp);  
}
```

### Splitting with custom algorithm 1

```
List<String> result = new ArrayList<String>();  
int z = 0;  
for (int i=0; i<addresses.length(); i++) {  
    if (i==addresses.length()-1) {  
        result.add(addresses.substring(z, i+1));  
    }  
    if (addresses.charAt(i)==separator) {  
        result.add(addresses.substring(z, i));  
        z=i+1;  
    }  
}
```

### Splitting with custom algorithm 2

```
List<String> curAddrs = new ArrayList<String>();  
String buffer = "";  
for (int i=0; i<addresses.length(); i++) {  
    if (addresses.charAt(i) != separator) {  
        buffer += addresses.charAt(i);  
    } else {  
        curAddrs.add(buffer);  
        buffer = "";  
    }  
}  
curAddrs.add(buffer);
```

### Splitting with custom algorithm 3

```
List<String> emailListe= new ArrayList<String>();  
int trenneralt = 0;  
while (addresses.indexOf(separator, trenneralt) !=  
    -1) {  
    int trennerneu = addresses.indexOf(separator,  
        trenneralt);  
    emailListe.add(addresses.substring(trenneralt,  
        trennerneu));  
    trenneralt = trennerneu + 1;  
}
```

PASTE

PASTE

COPY

PASTE

PAS







**COPY**

**PASTE**

**PASTE**

**PASTE**

**PASTE**

# Fazit

Klone sind ein Problem für die Weiterentwicklung. In allen Softwareartefakten.

# Kontakt

Dr. Elmar Jürgens · [juergens@cqse.eu](mailto:juergens@cqse.eu) · +49 179 675 3863

CQSE GmbH  
Lichtenbergstraße 8  
85748 Garching bei München

