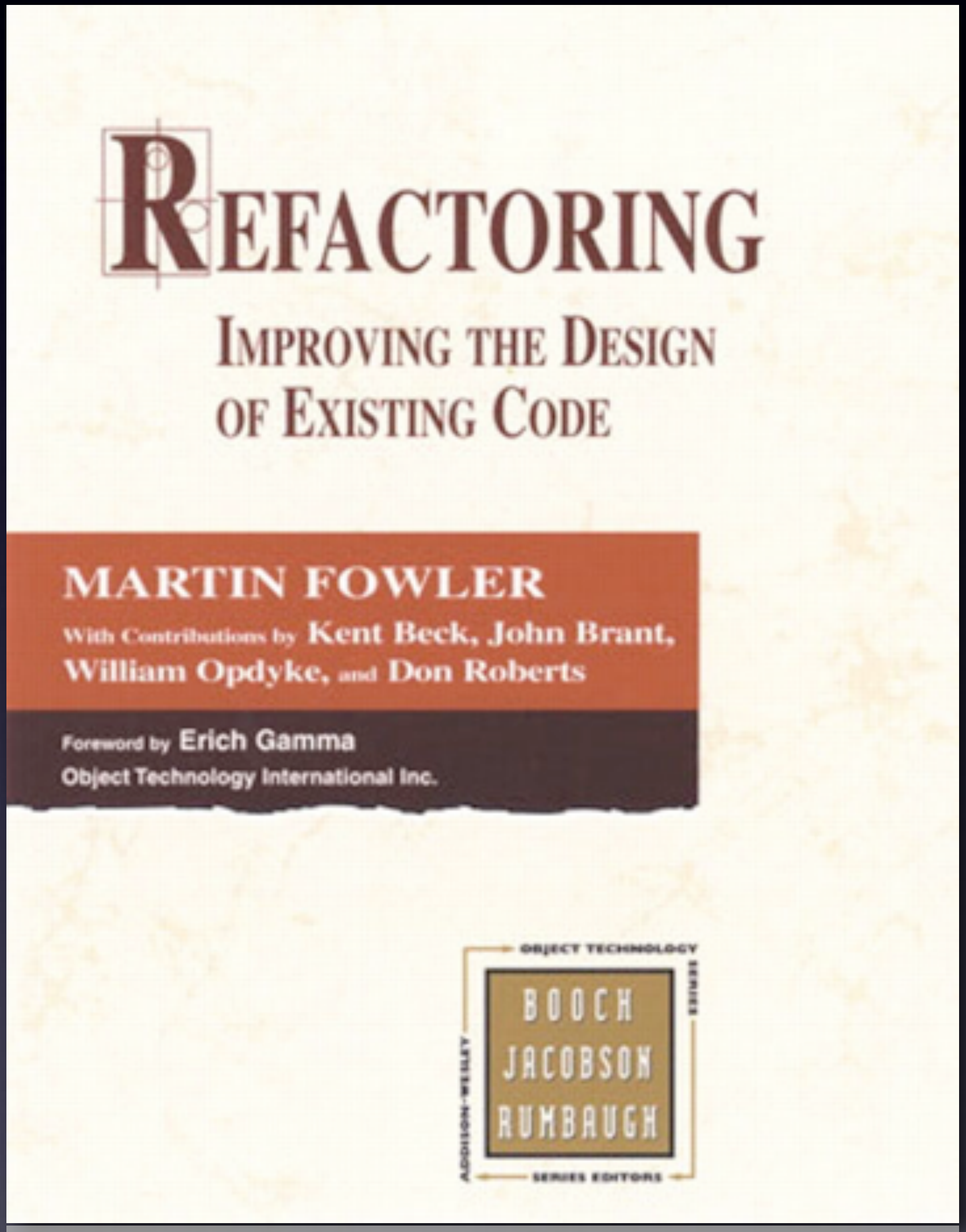
What happened to my clone?

Jens Krinke
Centre for Research on Evolution, Search and Testing (CREST)
University College London

COW 9, 23/11/2010

Duplicated Code

Number one in the stink parade is duplicated code. If you see the same code structure in more than one place, you can be sure that your program will be better if you find a way to unify them.


REFACTORING
IMPROVING THE DESIGN OF EXISTING CODE
MARTIN FOWLER
With Contributions by Kent Beck, John Brant, William Opdyke, and Don Roberts
Foreword by Erich Gamma
Object Technology International Inc.
OBJECT TECHNOLOGY SERIES
BOOCH
JACOBSON
RUMBAUGH
SERIES EDITORS
ADDISON-WESLEY

Overview

Research Questions:

1. Is Clone Code more stable?
2. Are Clones changed consistently?
3. Can Originals and Copies be identified?

RQ #1:
Is Clone Code more stable?

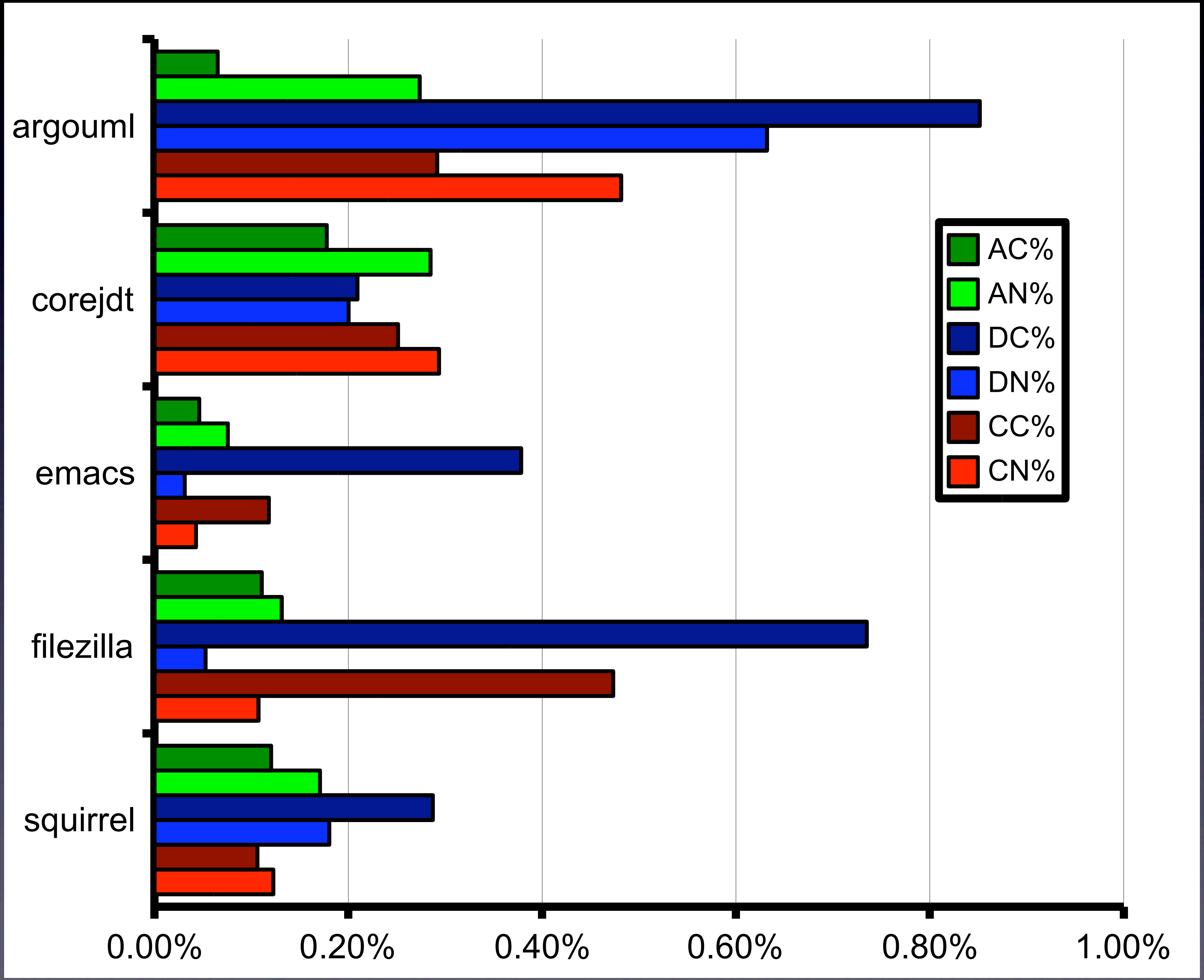
- If cloned code is changed often, it requires more attention and is more expensive
- If cloned code is more stable,
  its maintenance costs will be lower
- No data on stability exist

Empirical Study

- 5 open source systems
- 200 weeks of evolution:
  200 snapshots
- Clones:
  200 sets (using simian)
- Changes (addition, deletions, changes):
  200 diffs to the next week
- Changes are mapped to clones

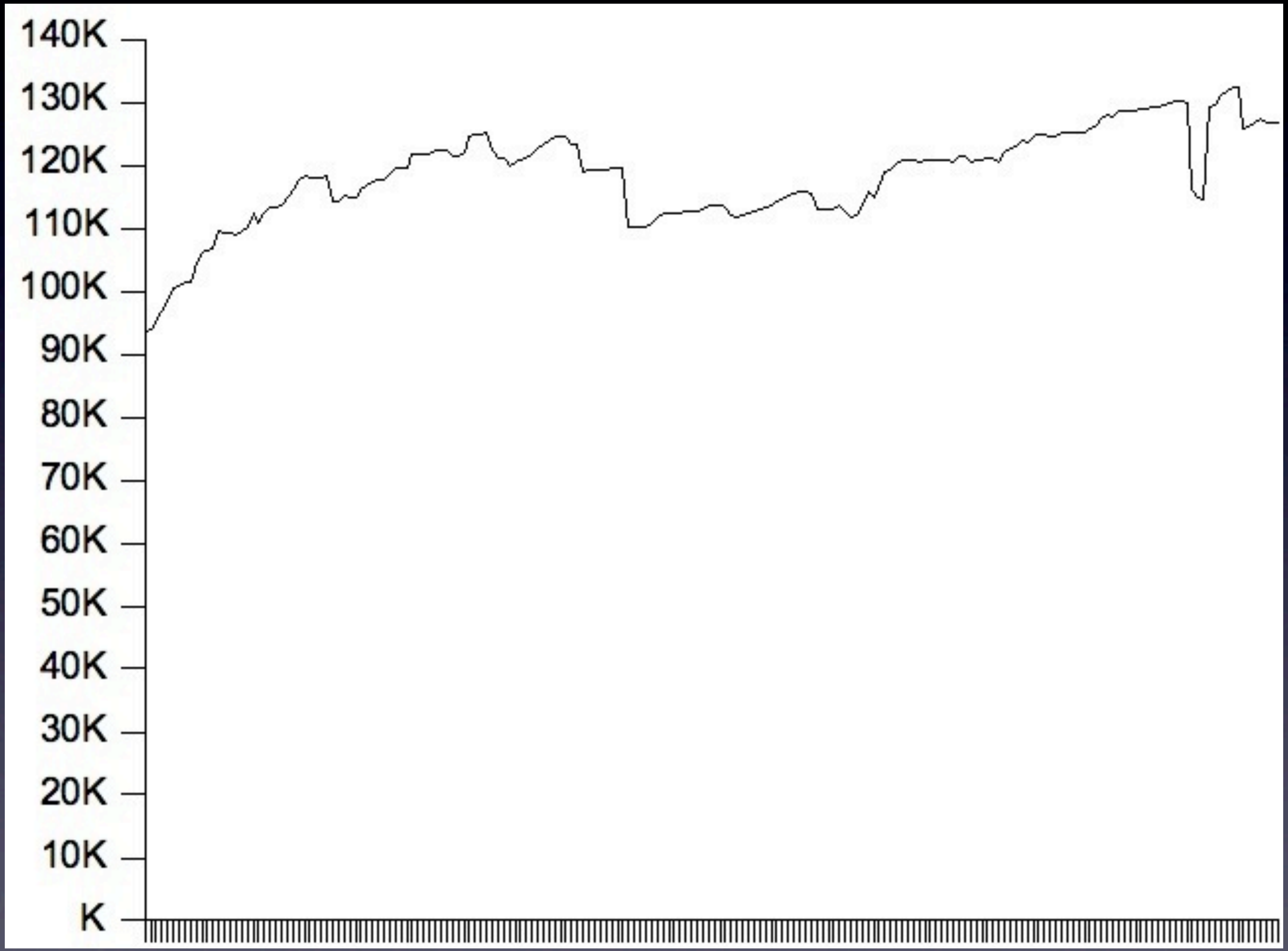
ArgoUML	118.316	12%
jdt.core	192.624	15%
Emacs	227.919	10%
FileZilla	90.302	16%
SQuirreL	69.981	8%

Results

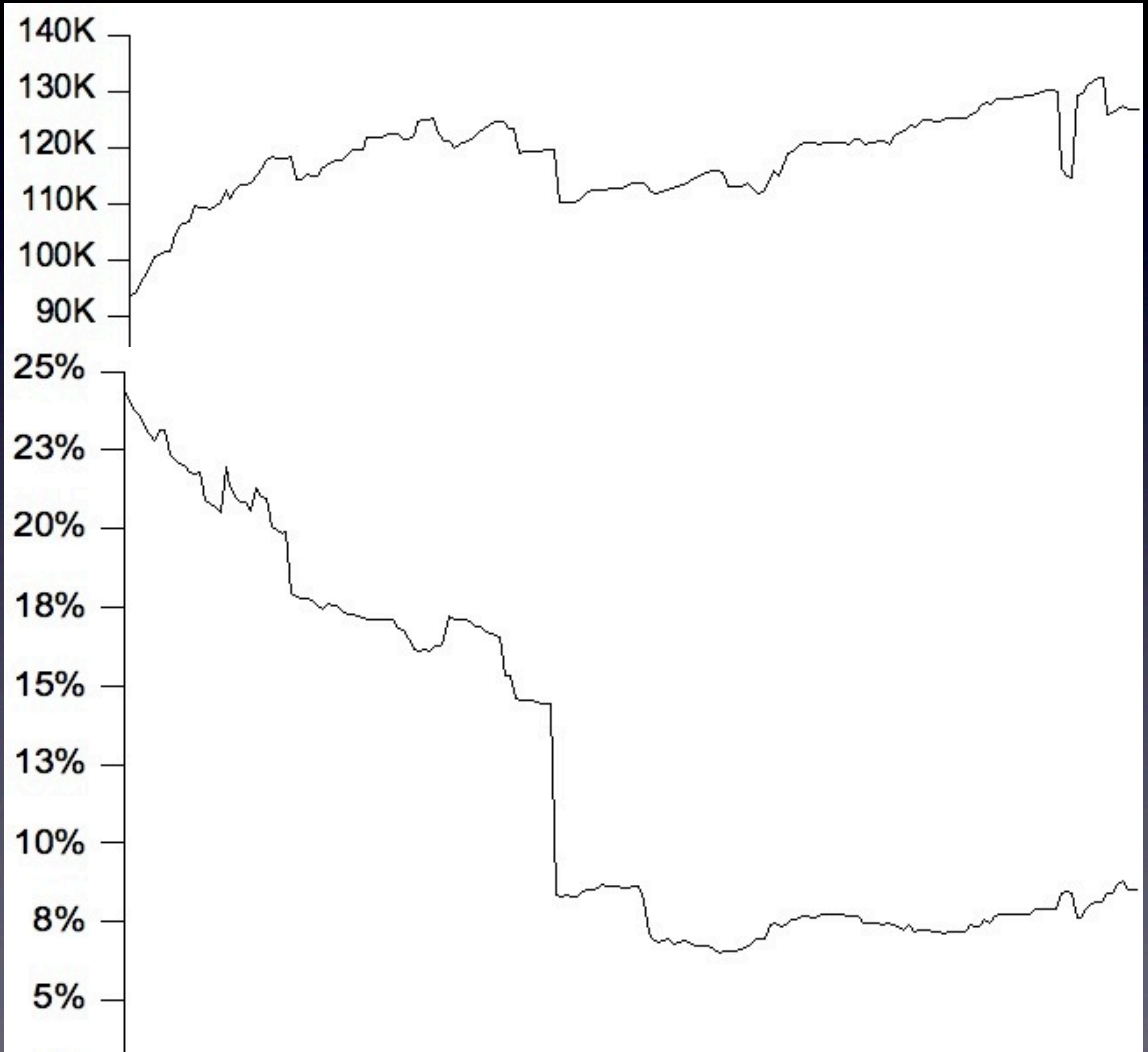


argouml
corejdt
emacs
filezilla
squirrel
0.00%
0.20%
0.40%
0.60%
0.80%
1.00%
AC%
AN%
DC%
DN%
CC%
CN%

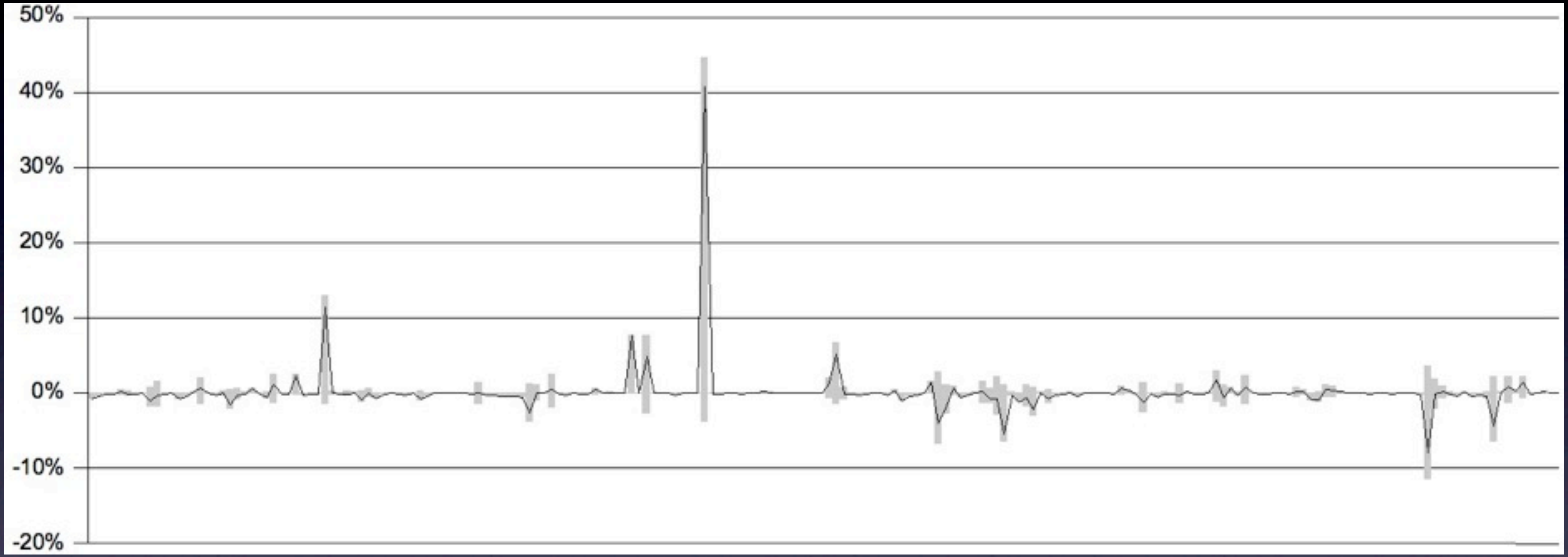
ArgoUML



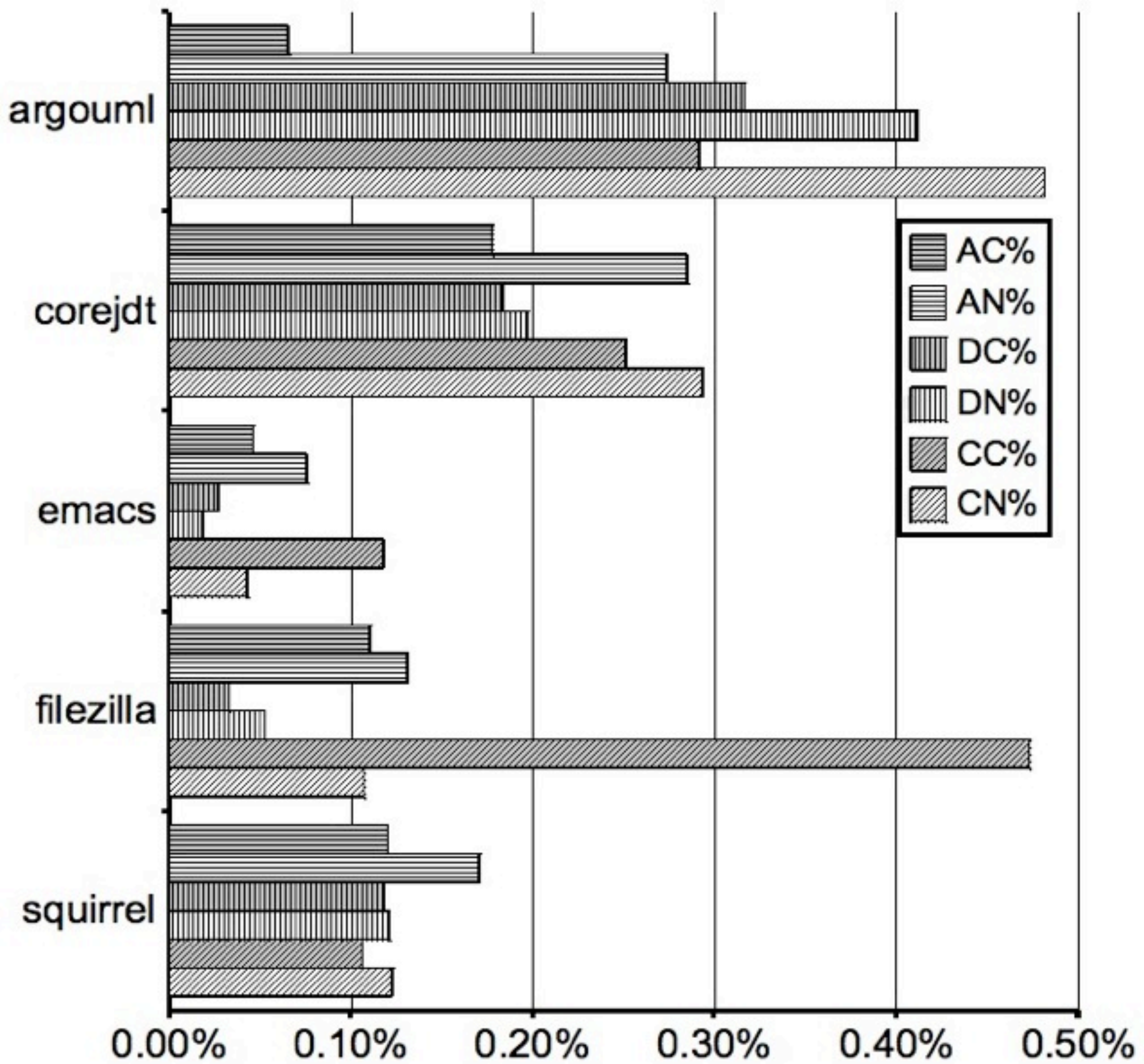
ArgoUML



Deletions Dominate



Cleaner Results



RQ #1:
Is Clone Code more stable?

- The average percentage of additions, deletions, or other changes to cloned code is lower than the average percentage for non-cloned code
- More often a higher percentage of non-cloned code is added, deleted, or changed in comparison to cloned code
- **Cloned code is more stable than Non-Cloned Code**

RQ #2:
Are Clones changed consistently?

- If cloned code is changed consistently, it evolves together
- If cloned code is changed consistently, inconsistent changes may be bugs

Hypothesis #1

During the evolution of a system,
code clones of a clone group
are changed consistently

- Two studies suggest that consistent changes do not appear as often as expected [Kim et al, Aversano et al]
- Both studies analyzed small Java systems

Hypothesis #2

During the evolution of a system,
if code clones of a clone group
are not changed consistently,
missing changes will appear in a later version

Changes in Clones

- A clone is identified by file, start & end line
- A change is identified by file, start line, number of deleted line and added lines
- match changes to clones

➡ if all changes to the clones of a group are the same, the group has consistent changes

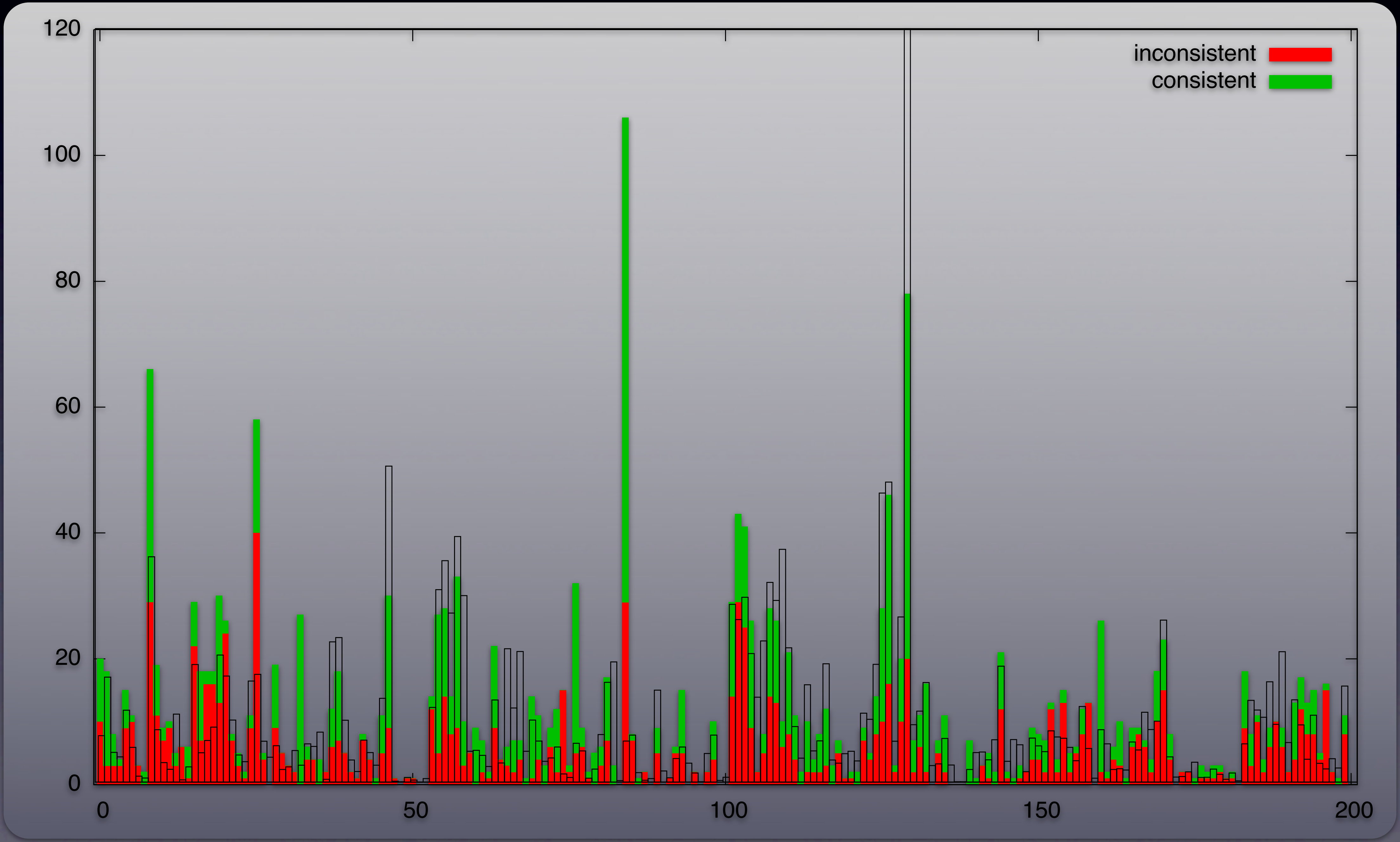
Analyzed Systems

	Source	Changes	Clones		
System	LOC	LOC	LOC		Groups
ArgoUML	118366	2816	14862	13%	313
CAROL	9824	248	601	6%	17
jdt.core	192930	2478	29438	15%	644
Emacs	227964	578	22966	10%	528
FileZilla	90138	698	14362	16%	210

Results

	$\|GC\|$	$\|GI\|$	$\frac{\|GC\|}{\|GI\|+\|GC\|}$
ArgoUML	1049	1050	50%
CAROL	66	69	49%
jdt.core	1375	1124	55%
Emacs	440	543	45%
FileZilla	246	204	55%

ArgoUML



Influence of Parameters

- Impact of change detection (diff): whitespace and indentation is ignored

➡ Manual inspection

  - Most changes are similar
  - Changes in arguments and predicates

Influence of Parameters

	Transformed			Original		
	$\|GC\|$	$\|GI\|$	$\frac{\|GI\|}{\|GD\|}$	$\|GC\|$	$\|GI\|$	$\frac{\|GI\|}{\|GD\|}$
ArgoUML	1049	1050	50%	1266	2988	30%
CAROL	66	69	49%	77	170	31%
jdt.core	1375	1124	55%	1416	2194	39%
Emacs	440	543	45%	480	1006	32%
FileZilla	246	204	55%	270	316	46%

Hypothesis #1 (invalidated)

During the evolution of a system,
code clones of a clone group
are changed consistently

✘ is only valid half of the time

Hypothesis #2

During the evolution of a system,
if code clones of a clone group
are not changed consistently,
missing changes will appear in a later version

- the rate of consistently changed groups will increase for longer durations

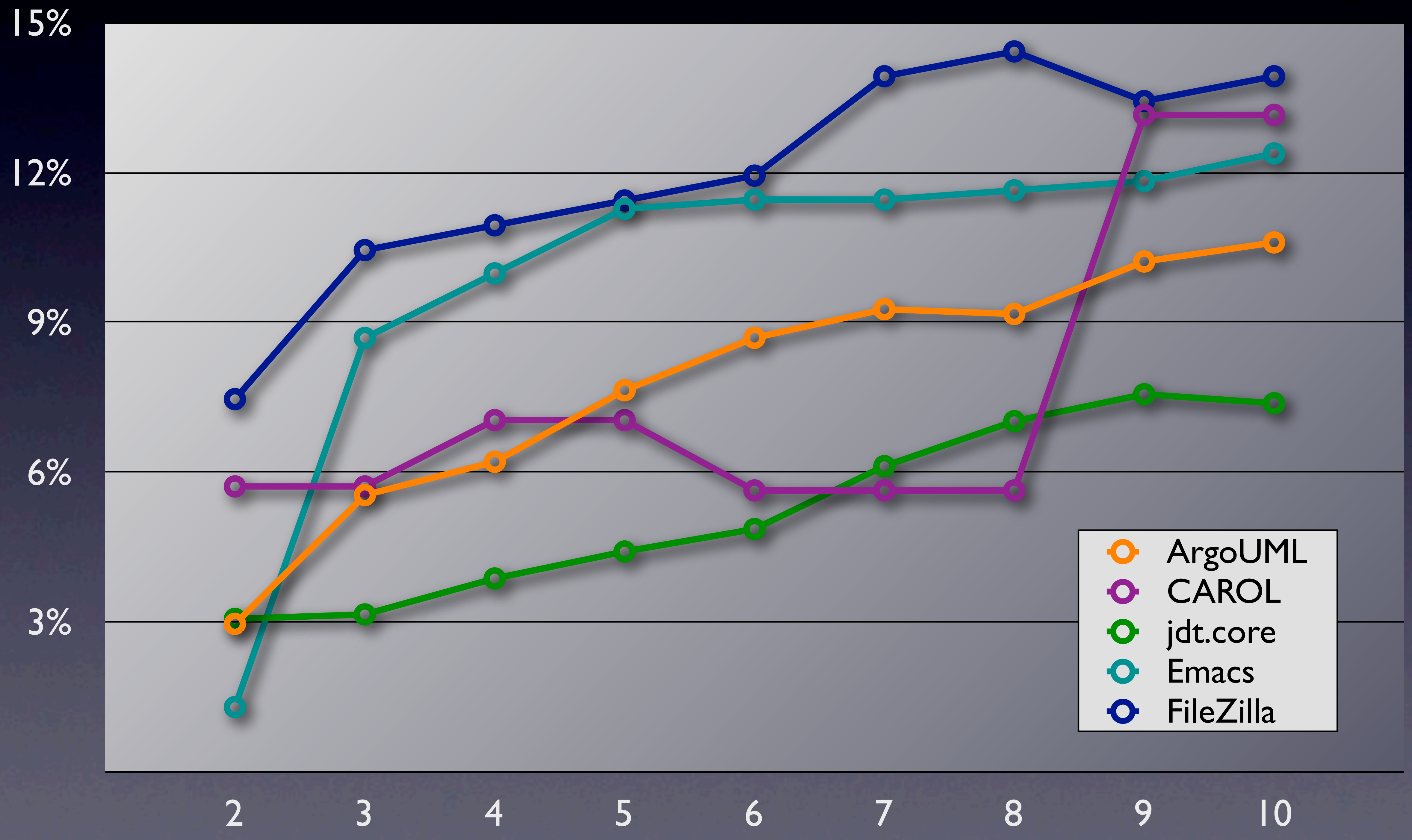
✘ no significant change observed!

Hypothesis #2

During the evolution of a system,
if code clones of a clone group
are not changed consistently,
missing changes will appear in a later version

- compute the probability that inconsistent changes turn into consistent changes
- if group is changed inconsistently in week *w*

Hypothesis #2



RQ #2:
Are Clones changed consistently?

- half of changed clone groups are inconsistently changed
- if a clone group is inconsistently changed, there is an increasing probability that it is consistently changed later

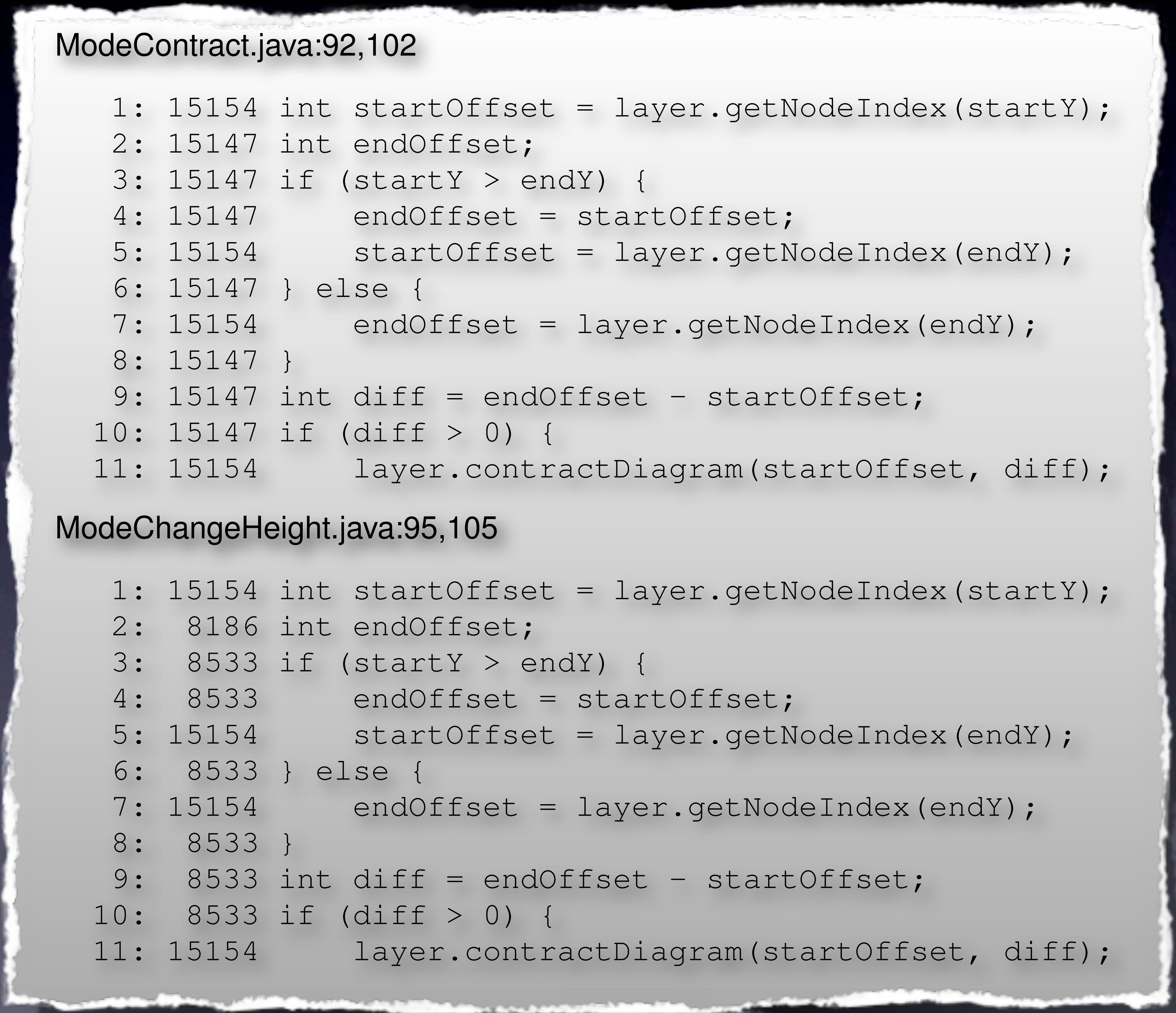
Do Clones lead to Bugs?

- Rahman et al., “Clones: What is that Smell” MSR 2010
- most bugs have very little to do with clones
- cloned code contains less buggy code
- larger clone groups don’t have more bugs than smaller clone groups
- making more copies of code doesn’t introduce more defects

RQ #3: Can Originals and Copies be identified?

- Where is my code coming from?
- Who is the original author?
- Are licenses violated by external code?

Version Controls Systems can ‘blame’



ModeContract.java:92,102

```
 1: 15154 int startOffset = layer.getNodeIndex(startY);
 2: 15147 int endOffset;
 3: 15147 if (startY > endY) {
 4: 15147     endOffset = startOffset;
 5: 15154     startOffset = layer.getNodeIndex(endY);
 6: 15147 } else {
 7: 15154     endOffset = layer.getNodeIndex(endY);
 8: 15147 }
 9: 15147 int diff = endOffset - startOffset;
10: 15147 if (diff > 0) {
11: 15154     layer.contractDiagram(startOffset, diff);
```

ModeChangeHeight.java:95,105

```
 1: 15154 int startOffset = layer.getNodeIndex(startY);
 2:  8186 int endOffset;
 3:  8533 if (startY > endY) {
 4:  8533     endOffset = startOffset;
 5: 15154     startOffset = layer.getNodeIndex(endY);
 6:  8533 } else {
 7: 15154     endOffset = layer.getNodeIndex(endY);
 8:  8533 }
 9:  8533 int diff = endOffset - startOffset;
10:  8533 if (diff > 0) {
11: 15154     layer.contractDiagram(startOffset, diff);
```

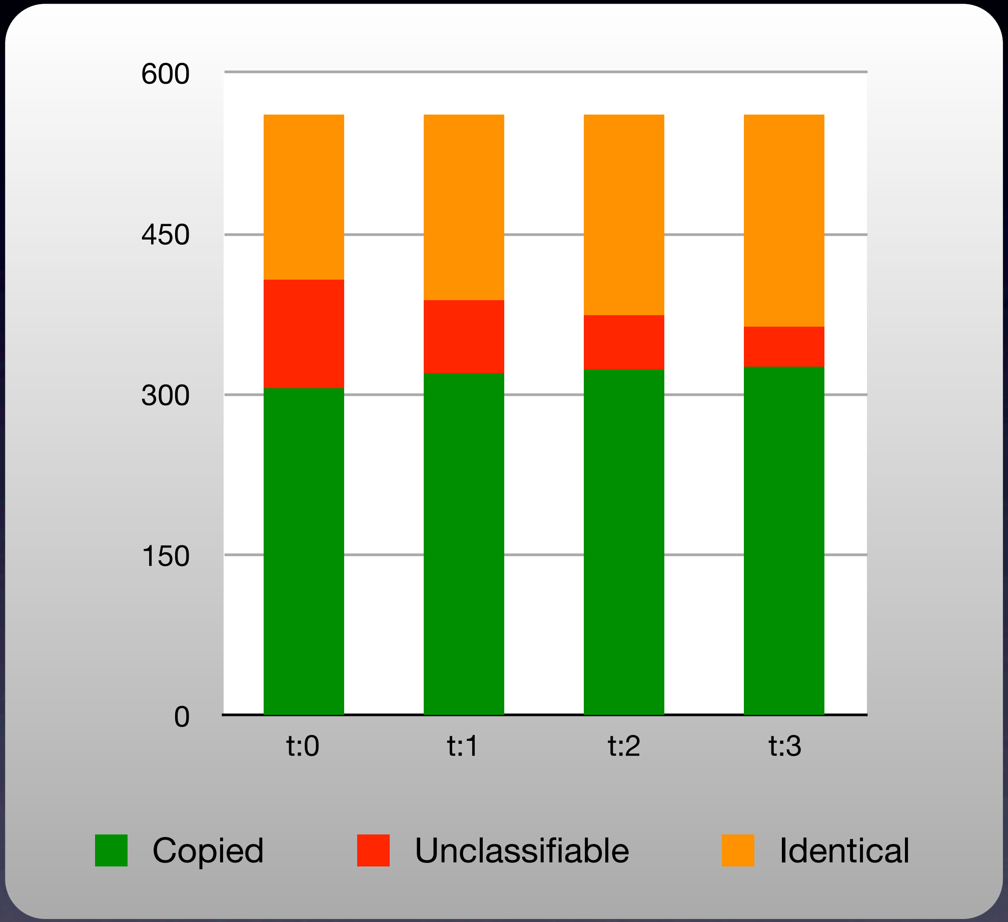
Classification

- A clone pair is *identical* if all corresponding lines have the same version.
- A clone pair is *copied* if the versions of all lines are either larger or smaller than the corresponding lines' versions.
- A clone pair is *unclassifiable* if it is neither identical nor copied.

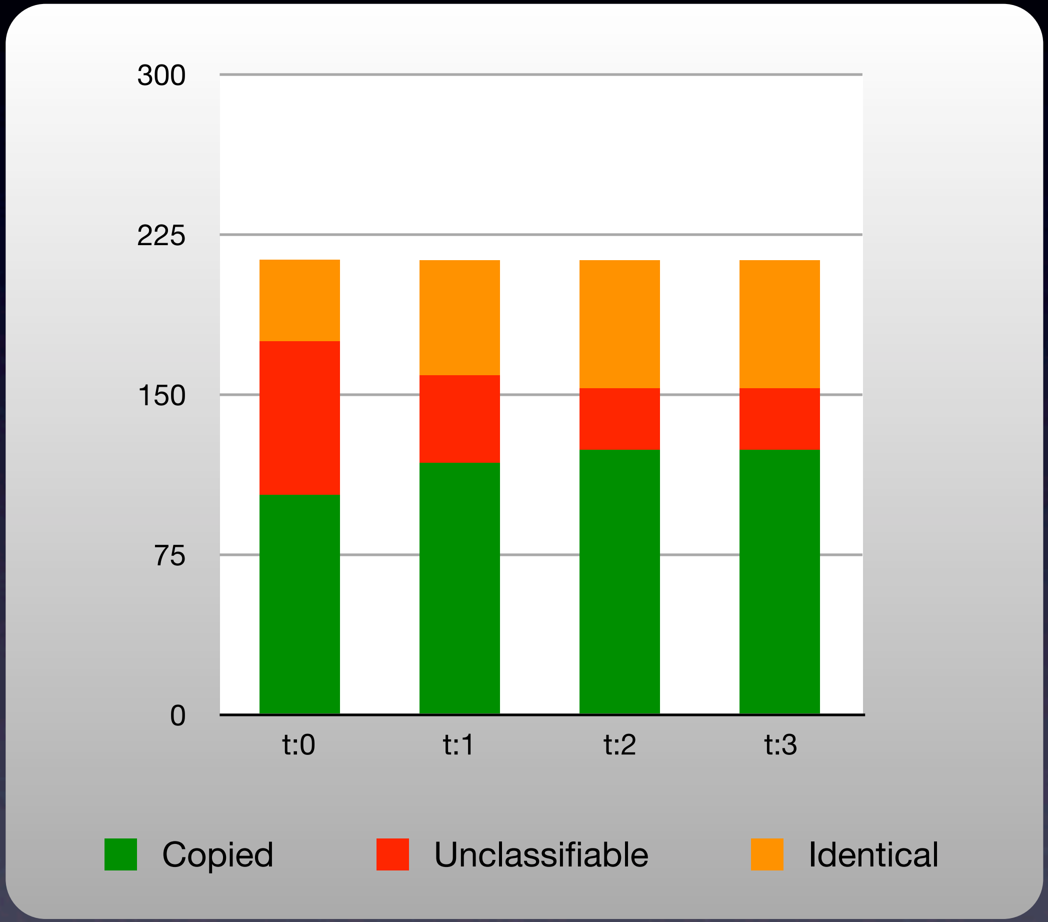
Tolerance

- The clones of a clone pair are said to be classifiable with a tolerance of *t* if after removing *t* source lines the resulting pair can be classified as copied or identical.
- Compute the Levenshtein Distance between the strings of versions.

Classification Results



ArgoUML



Apache

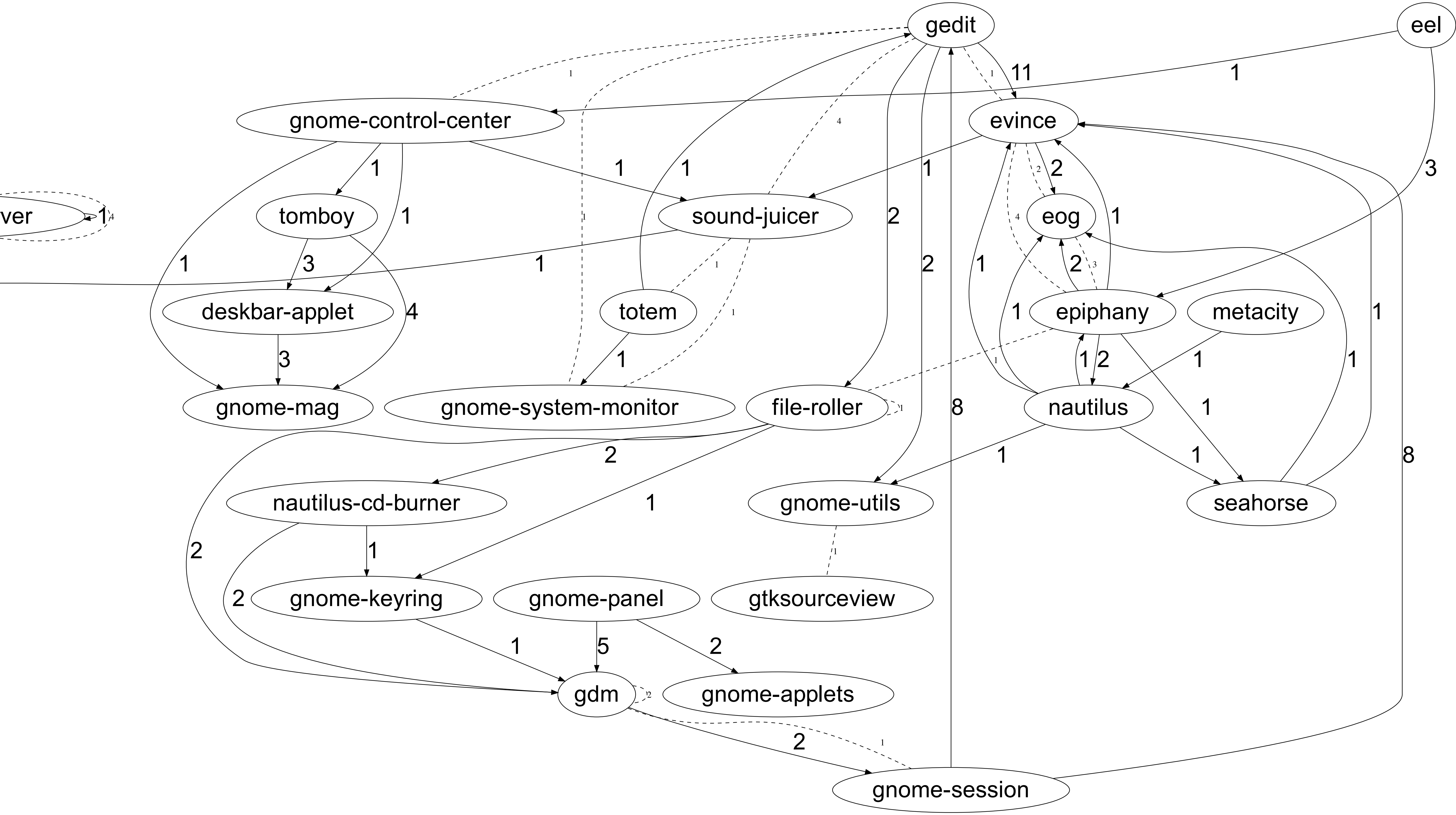
RQ #3:
Can Originals and Copies be identified?

- When comments are ignored and a small tolerance is accepted, the majority of clone pairs can automatically be distinguished between the original and the copy.

Flow between Projects

- The GNOME Desktop Suite consists of 68 projects, written in C.
- 4494 source files (*.c)
- 2.6 MLOC

Flow between Projects



Conclusions

- Cloned code is more stable
- Clone groups are inconsistently changed half of the time
- If a clone group is inconsistently changed, it may consistently changed later
- For the majority of clone pairs, the original can automatically be distinguished from the copy.