Evolution of Type-1 Clones

Nils Göde
University of Bremen

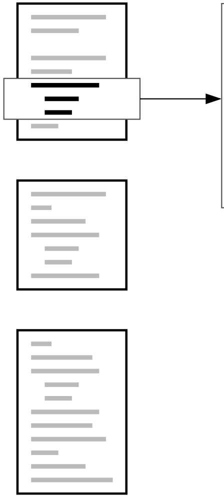
SCAM'09

9th IEEE International Working Conference
on Source Code Analysis and Manipulation

September 20th 2009

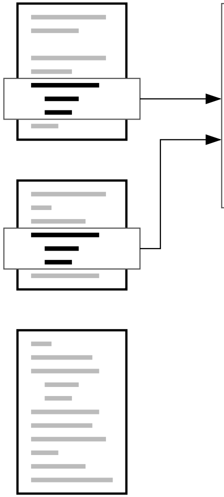
Software Clones

Software Clones



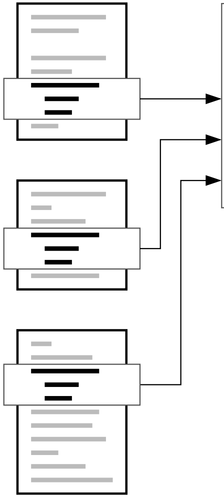
```
String nameParts[] = author[1].split(" ");
int i = 0;
for (i = 0; i < nameParts.length; i++) {
  c = nameParts[i].charAt(0);
  sb.append(c + ".");
}
```

Software Clones



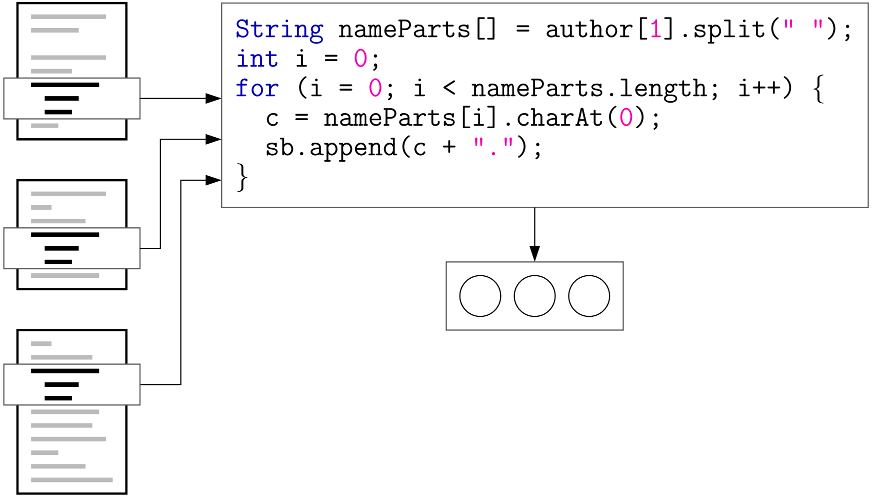
```
String nameParts[] = author[1].split(" ");
int i = 0;
for (i = 0; i < nameParts.length; i++) {
  c = nameParts[i].charAt(0);
  sb.append(c + ".");
}
```

Software Clones



```
String nameParts[] = author[1].split(" ");
int i = 0;
for (i = 0; i < nameParts.length; i++) {
  c = nameParts[i].charAt(0);
  sb.append(c + ".");
}
```

Software Clones



```
String nameParts[] = author[1].split(" ");
int i = 0;
for (i = 0; i < nameParts.length; i++) {
  c = nameParts[i].charAt(0);
  sb.append(c + ".");
}
```

Why Another Evolution Model?

- Overcome limitation to predefined patterns

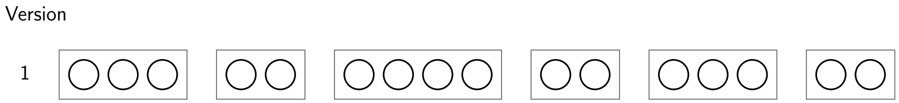
Why Another Evolution Model?

- Overcome limitation to predefined patterns
- Map fragments instead of classes

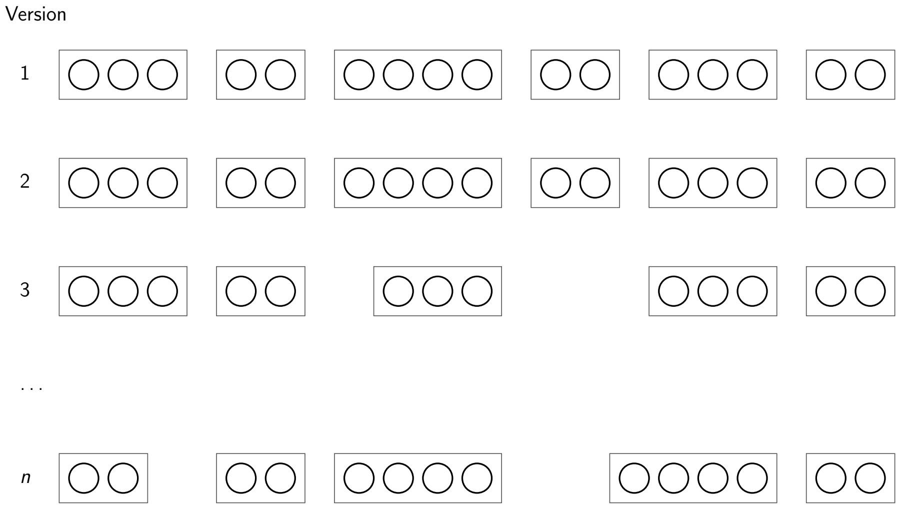
Why Another Evolution Model?

- Overcome limitation to predefined patterns
- Map fragments instead of classes
- Reduce computational effort

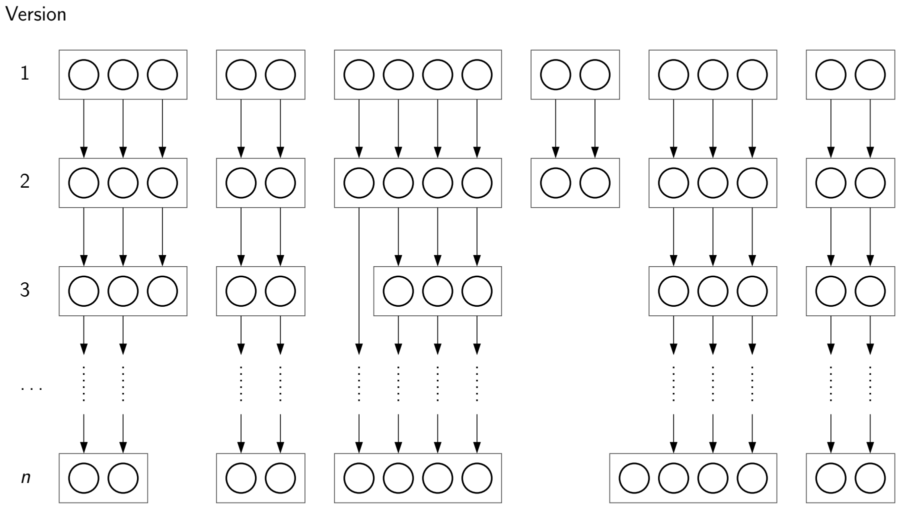
Clone Evolution Model



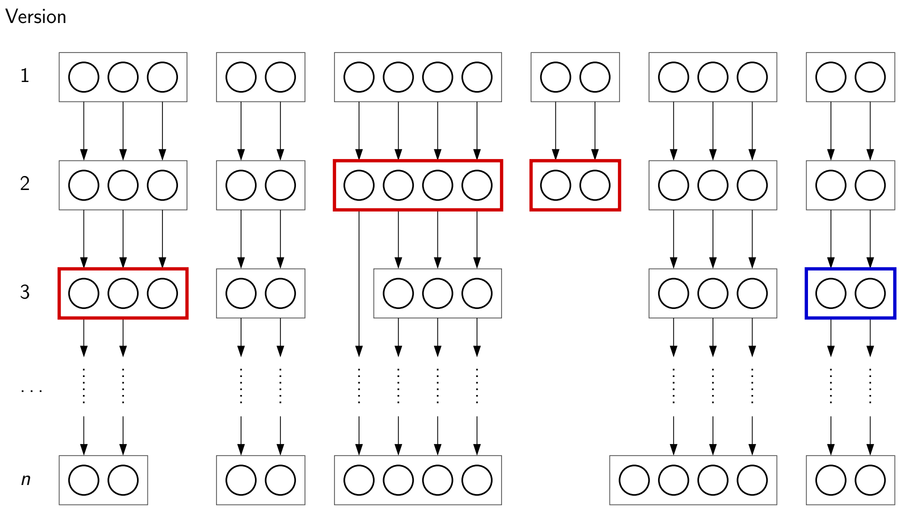
Clone Evolution Model



Clone Evolution Model



Clone Evolution Model



Why More Case Studies?

Why More Case Studies?

- Controversial results

Why More Case Studies?

- Controversial results
- Results contradict our experience

Why More Case Studies?

- Controversial results
- Results contradict our experience
- Limited diversity of subject systems

Why More Case Studies?

- Controversial results
- Results contradict our experience
- Limited diversity of subject systems

Our case studies covered. . .

Why More Case Studies?

- Controversial results
- Results contradict our experience
- Limited diversity of subject systems

Our case studies covered. . .

$\longrightarrow$ 9 subject systems

Why More Case Studies?

- Controversial results
- Results contradict our experience
- Limited diversity of subject systems

Our case studies covered. . .

$\rightarrow$ 9 subject systems

$\rightarrow$ 200 versions each ($\approx$ 4 years)

Why More Case Studies?

- Controversial results
- Results contradict our experience
- Limited diversity of subject systems

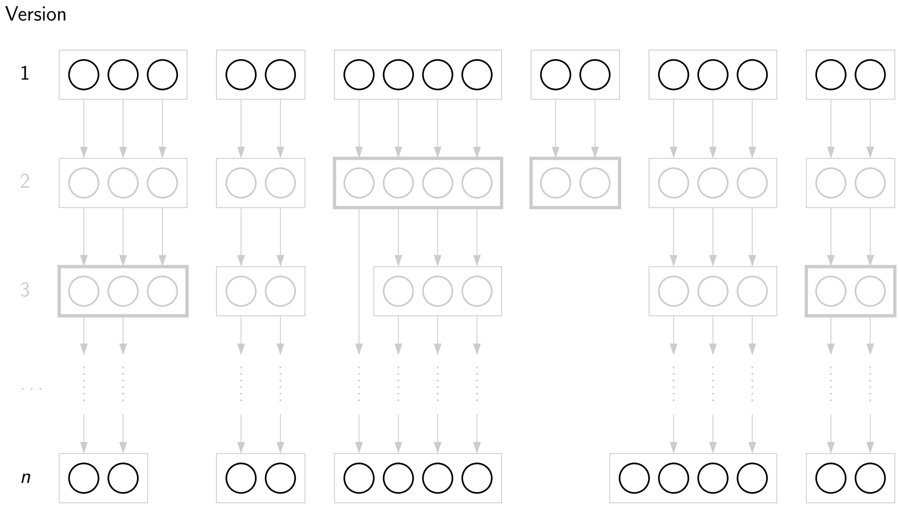
Our case studies covered. . .

$\rightarrow$ 9 subject systems

$\rightarrow$ 200 versions each ($\approx$ 4 years)

$\rightarrow$ 3 programming languages (C++, Java, C)

Clone Ratio



Clone Ratio

"[. . .] the overall ratio of function clones remains stable [. . .]"

[Laguë et al., 1997]

Clone Ratio

"[...] the overall ratio of function clones remains stable [...]"

[Laguë et al., 1997]

"[...] the CP% remains relatively stable over the several recent versions [...]"

[Li et al., 2006]

Clone Ratio

"[. . .] the overall ratio of function clones remains stable [. . .]"

[Laguë et al., 1997]

"[. . .] the CP% remains relatively stable over the several recent versions [. . .]"

[Li et al., 2006]

"[. . .] the number of code clones [. . .] is somewhat proportional to the size [. . .]"

[Livieri et al., 2007]

Clone Ratio

"[. . .] the overall ratio of function clones remains stable [. . .]"

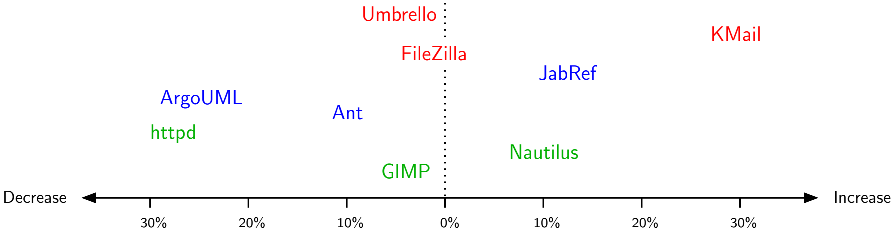
[Laguë et al., 1997]

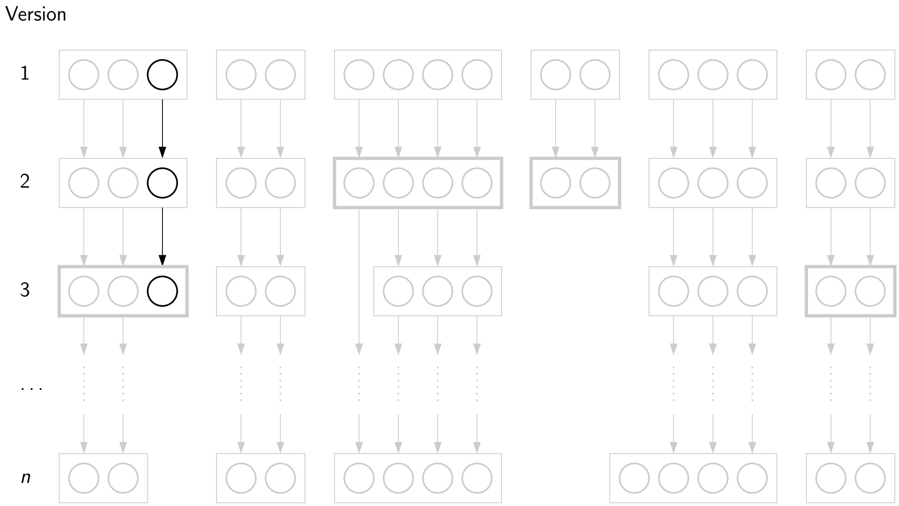
"[. . .] the CP% remains relatively stable over the several recent versions [. . .]"

[Li et al., 2006]

"[. . .] the number of code clones [. . .] is somewhat proportional to the size [. . .]"

[Livieri et al., 2007]




Version
1
2
3
...
n

Mean Fragment Lifetime

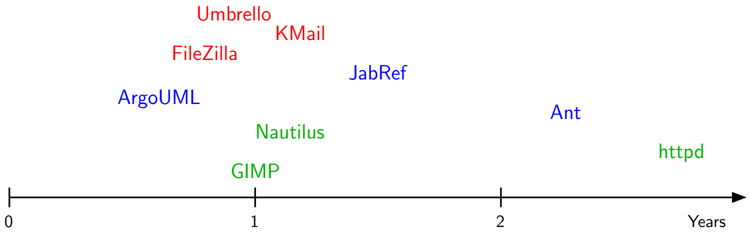
"[. . .] a large number of clones were volatile."

[Kim et al., 2005]

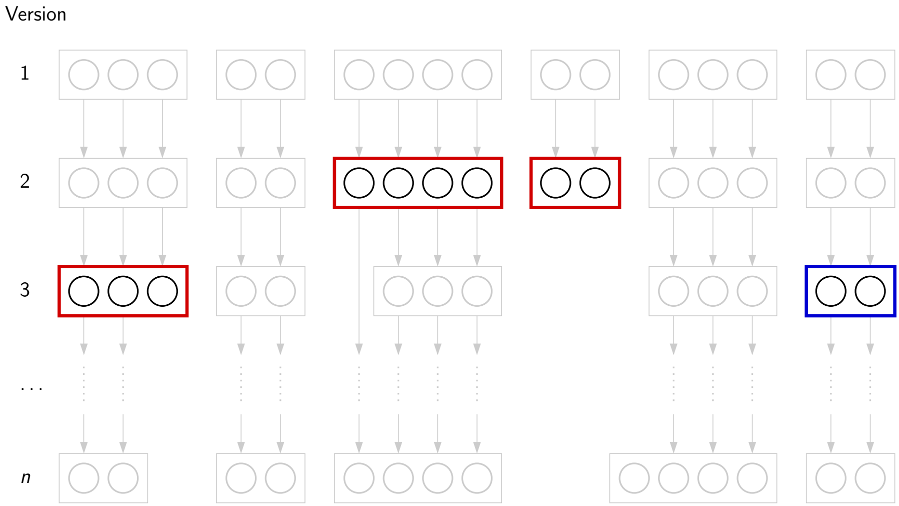
Mean Fragment Lifetime

"[. . .] a large number of clones were volatile."

[Kim et al., 2005]



Inconsistent Changes



Inconsistent Changes

“[. . .] the majority of clone classes is always maintained consistently.”

[Aversano et al., 2007]

Inconsistent Changes

"[. . .] the majority of clone classes is always maintained consistently."

[Aversano et al., 2007]

"[. . .] clone groups are consistently changed in roughly half of the time [. . .]"

[Krinke, 2007]

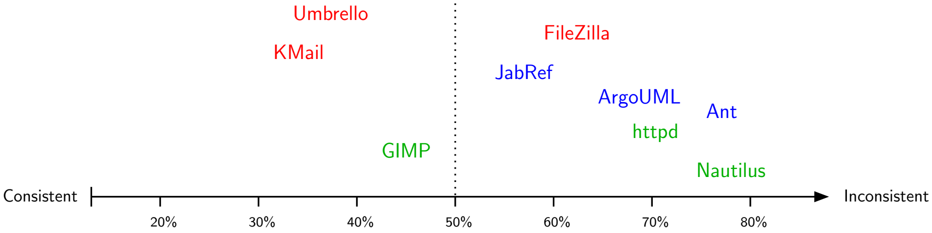
Inconsistent Changes

"[. . .] the majority of clone classes is always maintained consistently."

[Aversano et al., 2007]

"[. . .] clone groups are consistently changed in roughly half of the time [. . .]"

[Krinke, 2007]



“The harmfulness of clones cannot be judged without considering their evolution.”

L. Aversano, L. Cerulo, and M. Di Penta. How clones are maintained: An empirical study. In *Proceedings of the 11th European Conference on Software Maintenance and Reengineering*, pages 81–90. IEEE Computer Society, 2007.

M. Kim, V. Sazawal, D. Notkin, and G. C. Murphy. An empirical study of code clone genealogies. In *Proceedings of the Joint 10th European Software Engineering Conference and the 13th ACM SIGSOFT Symposium on the Foundations of Software Engineering*, pages 187–196. ACM, 2005.

J. Krinke. A study of consistent and inconsistent changes to code clones. In *Proceedings of the 14th Working Conference on Reverse Engineering*, pages 170–178. IEEE Computer Society, 2007.

B. Laguë, D. Proulx, J. Mayrand, E. Merlo, and J. Hudepohl. Assessing the benefits of incorporating function clone detection in a development process. In *Proceedings of the International Conference on Software Maintenance*, pages 314–321. IEEE Computer Society, 1997.

Z. Li, S. Lu, S. Myagmar, and Y. Zhou. CP-Miner: Finding copy-paste and related bugs in large-scale software code. *IEEE Transactions on Software Engineering*, 32(3):176–192, 2006.

S. Livieri, Y. Higo, M. Matsushita, and K. Inoue. Analysis of the linux kernel evolution using code clone coverage. In *Proceedings of the 4th International Workshop on Mining Software Repositories*, pages 22–25. IEEE Computer Society, 2007.