

WHITEPAPER FRONTENDART



ATTACK OF THE CLONES

Source code clones

Source code cloning, also known as copy and paste programming is believed to be a major threat to software systems, mostly the ones currently in their implementation phase. The problem is often overlooked due to the true danger lying in the clones' long-term existence, specifically in their unmonitored evolution – hence the immediate negative impact cannot be observed. When a clone instance of a given method needs maintenance (bug fixing) or further code development (enhancement), usually the tasks would require to be performed on the other clone instances as well for the consistency to remain.

Neglecting to maintain other clone instances can have drastically negative influence on software maintainability in the long run, which finally concludes in very high maintenance costs –especially in the later stages of software evolution, where maintenance is an expensive task on its own.

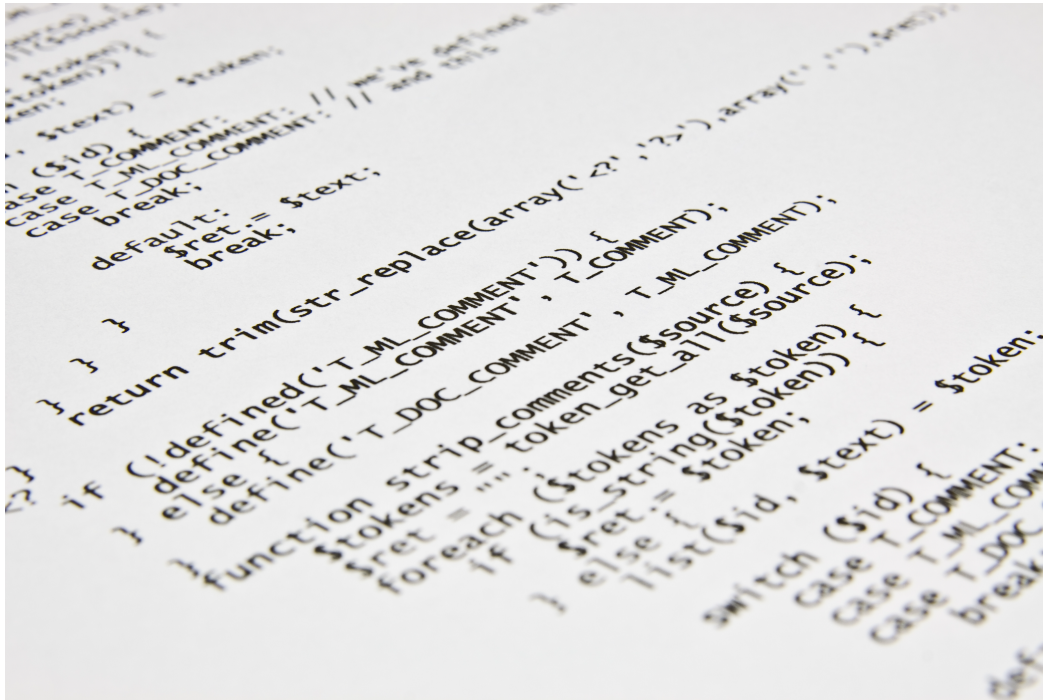
Clone instances are usually not left out of maintenance intentionally (because of irresponsibility or ignorance), the true reason rather lies in the complexity of the software system. After a certain size, time or change in source code or even staff members, the clone instances can be a hard, if not even impossible task to track and maintain by hand.

Also, while basically clones have their negative effects, but sometimes they are necessary components to a project. This is because total omission of clones may result in a source code with high coupling (which should also be avoided) and in some cases, it may lead to overly complicated, virtually unmanageable sections of the code.

Also, removing every existing clone is contraindicated for further reasons; it can simply be too big of a task and can over-encumber the programmers while also being harmful. Clones should be tracked and maintained with an appropriate strategy.

Cause		Σ
Consistent changes	Instances deleted	1
	Intentional refactoring	3
	Instances are newly created	10
	Σ	14
Inconsistent changes	Some instances deleted	1
	Inconsistent changes	21
	New instances added	4
	Σ	26
Σ		40

Table: Consistent and inconsistent changes in jEdit



In a research of the Department of Software Engineering, University of Szeged on clone tracking, our state-of-the-art clone tracking algorithm was applied on 84 consecutive versions of jEdit, a Java-based, open-source text editor software. It found 50 different suspicious clone evolution patterns and 65% of the so-called clone smells (see Table for details) pointed out inconsistent changes in source code (which then later led to bug fixes, as they were discovered).¹

Our research led to the conclusion that detecting clones is insufficient on its own to retain suitable maintainability level. Clone instances need to be tracked and maintained along the development process – by a dedicated and automatized solution if possible.

¹ http://link.springer.com/chapter/10.1007%2F978-3-642-18381-2_7