Duplicate code

Duplicate code is a computer programming term for a sequence of source code that occurs more than once, either within a program or across different programs owned or maintained by the same entity. Duplicate code is generally considered undesirable for a number of reasons.[1] A minimum requirement is usually applied to the quantity of code that must appear in a sequence for it to be considered duplicate rather than coincidentally similar. Sequences of duplicate code are sometimes known as code clones or just clones, the automated process of finding duplications in source code is called clone detection.

Some of the ways in which two code sequences can be duplicates of each other are to be character-for-character identical, character-for-character identical with white space characters and comments being ignored, token-for-token identical, token-for-token identical with occasional variation or functionally identical.

Contents

How duplicates are created

Some of the reasons why duplicate code may be created include Copy and paste programming, which in academic settings may be done as part of plagiarism, or scrounging, in which a section of code is copied "because it works". In most cases this operation involves slight modifications in the cloned code such as renaming variables or inserting/deleting code. The language nearly always provides facilities to allow one copy of the code to serve multiple purposes, but a copy is created due to the programmer not truly knowing the language, not having the time to do it properly, or not caring about the increased active software rot.

It may also contain functionality that is very similar to that in another part of a program is required and a developer independently writes code that is very similar to what exists elsewhere. Studies suggest, that such independently rewritten code is typically not syntactically similar.[2]

Automatically generated code, where having duplicate code may be desired to increase speed or ease of development, is another reason for duplication. Note that the actual generator will not contain duplicates in its source code, only the output it produces.

Fixing

Duplicate code is most commonly fixed by moving the code into its own unit (function or module) and calling that unit from all of the places where it was originally used. Using a more open-source style of development, in which components are in centralized locations, may also help with duplication.

Costs and benefits

When code with a software vulnerability is copied, the vulnerability may continue to exist in the copied code if the developer is not aware of such copies.[3] Refactoring duplicate code can improve many software metrics, such as lines of code, cyclomatic complexity, and coupling. This may lead to shorter compilation times, lower cognitive load, less human error, and fewer forgotten or overlooked pieces of code. However, not all code duplication can be refactored.[4] Clones may be the most effective solution if the

programming language provides inadequate or overly complex abstractions, particularly if supported with user interface techniques such as simultaneous editing. Furthermore, the risks of breaking code when refactoring may outweigh any maintenance benefits.[5] Duplicated code does not seem to be significantly more error-prone than unduplicated code.[6]

Detecting duplicate code

A number of different algorithms have been proposed to detect duplicate code. For example:

- Baker's algorithm.[7]
- Rabin–Karp string search algorithm.
- Using Abstract Syntax Trees.[8]
- Visual clone detection.[9]
- Count Matrix Clone Detection.[10][11]
- Locality-sensitive hashing
- Anti-unification[12]

Example of functionally duplicate code

Consider the following code snippet for calculating the average of an array of integers

```
extern int array_a[];
extern int array_b[];

int sum_a = 0;

for (int i = 0; i < 4; i++)
   sum_a += array_a[i];

int average_a = sum_a / 4;

int sum_b = 0;

for (int i = 0; i < 4; i++)
   sum_b += array_b[i];

int average_b = sum_b / 4;
```

The two loops can be rewritten as the single function:

```
int calc_average_of_four(int* array) {
   int sum = 0;
   for (int i = 0; i < 4; i++)
       sum += array[i];

   return sum / 4;
}
```

Using the above function will give source code that has no loop duplication:

```
extern int array1[];
extern int array2[];

int average1 = calc_average_of_four(array1);
int average2 = calc_average_of_four(array2);
```

Note that in this trivial case, the compiler may choose to inline both calls to the function, such that the resulting machine code is identical for both the duplicated and non-duplicated examples above. If the function is not inlined, then the additional overhead of the function calls will probably take longer to run (on the order of 10 processor instructions for most high-performance languages). Theoretically, this additional time to run could matter.

See also

- Abstraction principle (programming)
- Anti-pattern

- Don't repeat yourself
- List of tools for static code analysis
- Redundant code
- Rule of three (computer programming)

Example of duplicate code fix via code replaced by the method

References

1. Spinellis, Diomidis. "The Bad Code Spotter's Guide" (http://www.informit.com/articles/article.aspx?p=457502&seqNum=5). InformIT.com. Retrieved 2008-06-06.
2. Code similarities beyond copy & paste (https://www.cqse.eu/publications/2010-code-similarities-beyond-copy-paste.pdf) by Elmar Juergens, Florian Deissenboeck, Benjamin Hummel.
3. Li, Hongzhe; Kwon, Hyuckmin; Kwon, Jonghoon; Lee, Heejo (25 April 2016). "CLORIFI: software vulnerability discovery using code clone verification". *Concurrency and Computation: Practice and Experience*. **28** (6): 1900–1917. doi:10.1002/cpe.3532 (https://doi.org/10.1002%2Fcpe.3532).
4. Arcelli Fontana, Francesca; Zanoni, Marco; Ranchetti, Andrea; Ranchetti, Davide (2013). "Software Clone Detection and Refactoring". *ISRN Software Engineering*. **2013**: 1–8. doi:10.1155/2013/129437 (https://doi.org/10.1155%2F2013%2F129437).
5. Kapser, C.; Godfrey, M.W., ""Cloning Considered Harmful" Considered Harmful (http://plg2.cs.uwaterloo.ca/~migod/papers/2006/wcre06-clonePatterns.pdf)," 13th Working Conference on Reverse Engineering (WCRE), pp. 19-28, Oct. 2006
6. Wagner, Stefan; Abdulkhaleq, Asim; Kaya, Kamer; Paar, Alexander (2016). "On the relationship of inconsistent software clones and faults: an empirical study" (http://elib.uni-stuttgart.de/opus/volltexte/2016/10526/). *Proc. 23rd IEEE International Conference on Software Analysis, Evolution, and Reengineering (SANER 2016)*.
7. Brenda S. Baker. *A Program for Identifying Duplicated Code.* Computing Science and Statistics, 24:49–57, 1992.
8. Ira D. Baxter, et al. Clone Detection Using Abstract Syntax Trees (http://www.semanticdesigns.com/Company/Publications/ICSM98.pdf)
9. Visual Detection of Duplicated Code (http://www.iam.unibe.ch/~scg/Archive/Papers/Rieg98aEcoopWorkshop.pdf) Archived (https://web.archive.org/web/20060629083352/http://www.iam.unibe.ch/~scg/Archive/Papers/Rieg98aEcoopWorkshop.pdf) 2006-06-29 at the Wayback Machine by Matthias Rieger, Stephane Ducasse.
10. Yuan, Y. and Guo, Y. *CMCD: Count Matrix Based Code* Clone Detection, *in 2011 18th Asia-Pacific Software Engineering Conference. IEEE, Dec. 2011, pp. 250–257.*
11. Chen, X., Wang, A. Y., & Tempero, E. D. (2014). A Replication and Reproduction of Code Clone Detection Studies (http://www.qualitascorpus.com/pubs/ChenWangTemperoClones.pdf). In ACSC (pp. 105-114).
12. Bulychev, Peter, and Marius Minea. "Duplicate code detection using anti-unification (https://cyberleninka.ru/article/n/duplicate-code-detection-using-anti-unification)." Proceedings of the Spring/Summer Young Researchers' Colloquium on Software Engineering. No. 2. Федеральное государственное бюджетное учреждение науки Институт системного программирования Российской академии наук, 2008.

External links

- The University of Alabama at Birmingham: Code Clones Literature (https://archive.is/20121211121637/http://students.cis.uab.edu/tairasr/clones/literature/)
- Finding duplicate code in C#, VB.Net, ASPX, Ruby, Python, Java, C, C++, ActionScript, or XAML (http://alexdresko.com/2010/09/09/finding-duplicate-code-in-c-vb-net-aspx-ruby-python-java-c-c-actionscript-or-xaml)