Activity-Based Programmer's Knowledge Model for Personalized Search

Eduard KURIC*

Slovak University of Technology in Bratislava
Faculty of Informatics and Information Technologies
Ilkovičova, 842 16 Bratislava, Slovakia
kuric@fiit.stuba.sk

Every day, a programmer needs to answer several questions for the purpose of finding solutions and making decisions. It requires the integration of different kinds of project (software system) information, as well as, it depends on the programmer's knowledge, experience, skills and inference.

The main advantage of search-driven development should be that programmers save time and resources by reusing (external) source code (components) in their software projects. To support search-driven development it is not sufficient to implement a "mere" full text search over a base of source code. In [2], authors investigated what functionality programmers would ideally like to have in a source code search engine. They included characteristics as: *review by other independent programmers - programmer's feedback*, *reviews/ranking for the source code*, *satisfaction level of other programmers*, and *number of programmers actively using it.*

When a programmer reuses source code he has to trust the work of external programmers that are unknown to him. It is possible to solve using a trustability metric so that the programmers assess the quality of source code search results. Existing approaches are often based on collaborative filtering of programmers' votes and project activity of programmers [1]. It would be helpful for programmers to see not only some activity statistics about software project (components) but also a karma value of each author (programmer). If a target programmer will easily see in the search results that an experienced programmer with good reputation has participated in writing the source code (component) then the target programmer will be more likely to think about reusing. Thus, author's reputation can provide information to support programmers' decisions.

Reputation ranking can be a plausible way to rank source code search results, i.e., if we determine programmers' karma values, we can prefer software components based on reputation of their authors. To model programmer's reputation (to calculate programmer's karma value), we need to investigate software components which he

* Supervisor: Mária Bieliková, Institute of Informatics and Software Engineering

Spring 2013 PeWe Workshop, April 5, 2013, pp. 55–56.

created. We propose programmer's knowledge model and methods for its automatic retrieving. Currently, we focus on two factors, namely, calculation of programmer's know-how about used technologies and calculation of programmer's karma based on importance of components.

Our focus is to automatically find out, using source code, which technologies (libraries) programmers really use, when they used them and their depth (degree) of usage. Based on the result, we find out the programmer's know-how with the specific technology in the comparison with other programmers who used (use) this technology as well (in a software system).

Let's consider the programmer who knows the platform Java which includes various technologies. The programmer's know-how about used technologies is represented with so-called *know-how score* in interval $\langle 0,1\rangle \cup \{-1\}$, where 0 means that the programmer does not have any know-how (experience) with the selected technology comparing to other programmers, and 1 means that the programmer's know-how about the technology is the highest in the comparison with other programmers. The value -1 means that the programmer used the selected technology as the only one and therefore we are not able exactly to express his relative experience.

The method for automatic calculation of programmer's know-how about used technologies consists of these steps: (1) identification of the author of the source code and technologies which were used, (2) construction of the programmer's model of used technologies and the calculation of the know-how score for each used technology.

The next factor is calculation of programmer's karma based on importance of components which he authored/co-authored. We suppose that the more source code fragments of a program (project) refer to a method then its importance is greater. For the calculation of method importance, we inspired by PageRank algorithm, where the importance of a method is determined by how many methods call it.

The method for automatic calculation of programmer's karma consists of three main steps, namely, (1) construction of a graph of method dependencies from source code of a project and calculation of PageRank score for each method; (2) construction of an index which contains a list of all the methods, the number of their Logical Lines of Code, calculated PageRank score, authors with determining their degree of authorship; and (3) calculation of programmers' karma.

Acknowledgement. This contribution is the partial result of the Research & Development Operational Programme for the project Research of methods for acquisition, analysis and personalized conveying of information and knowledge, ITMS 26240220039, co-funded by the ERDF.

References

[1] Gysin, F., S., Kuhn, A.: A trustability metric for code search based on developer karma. In: *Proc. of ICSE Workshop on Search-driven Development - Users, Infrastructure, Tools and Evaluation*, ACM, New York, 2010, pp. 41-44.

[2] Sim, S., E., et al.: How well do search engines support code retrieval on the web? In: *ACM Trans. Softw. Eng. Methodol.*, ACM, New York, 2011, pp. 1-25.