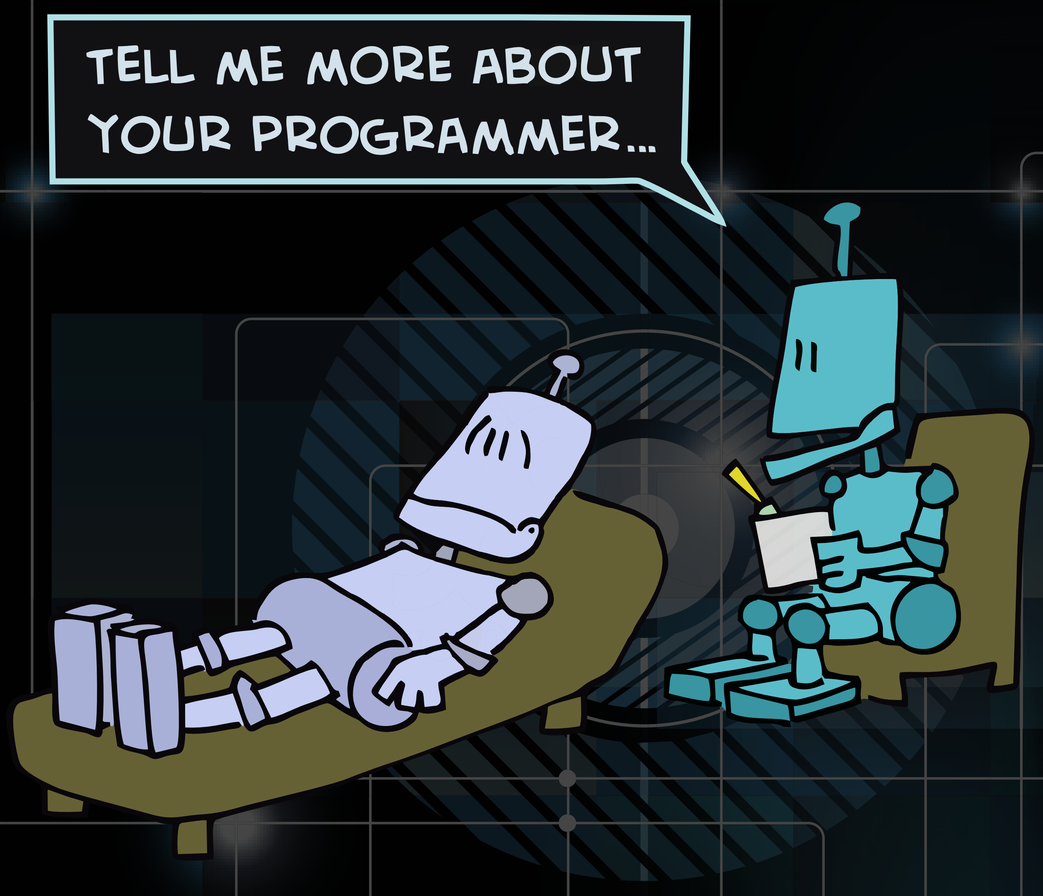
Activity-Based Programmer's Knowledge Model for Personalized Search in Source Code

Eduard Kuric, prof. Mária Bieliková



> Motivation

To support search-driven development it is not sufficient to implement a "mere" full text search over a base of source code. When a programmer reuses code he has to trust the work of external programmers that are unknown to him. **Reputation ranking** can be a plausible way to rank code results. It can be supported by using an externalized model of each programmer's knowledge of a particular code.

> Goals ... calculation of ...

- programmer's know-how about **used technologies**
- programmer's karma based on **importance of components**

Calculation of programmer's know-how about used technologies

I. identification of the author of the source code and technologies which were used
II. construction of the programmer's model of used technologies and the calculation of the know-how score for each technology

„If the programmer uses a method with the longer time difference he will probably have better experience about it than another programmer, who uses it with the shorter time difference (even though he is not currently using it)."

(1) Average time of a method usage:

$$t_{T_i,C_j,M_k,AVG} = \frac{\sum_l \left(t_{T_i,C_j,M_k} * x_{T_i,C_j,M_k}\right)_l}{\sum_l \left(x_{T_i,C_j,M_k}\right)_l}.$$

(2) Dispersion of the method usage by the programmer p in time t:

$$r_{T_i,C_j,M_k} = \sum_l \left(t_{T_i,C_j,M_k,AVG} - \left(t_{T_i,C_j,M_k}\right)_l\right)^2 * \left(x_{T_i,C_j,M'_k}\right)_l$$

(3) Experience with the component/technology:

$$kC_{T_i,j} = \frac{\sum_k r_{T_i,C_j,M_k}}{|M_{T_i,C_j}|} \qquad kT_i = \frac{\sum_j kC_{T_i,j}}{|C_{T_i}|}$$

Technology (library, package)			Component (class, interface)			Method (function)	
T_1	$\|C_{T1}\|$	kT_1	$C_{T1,1}$	$\|M_{T1,C1}\|$	$kC_{T1,1}$	$M_{T1,C1,1} = \{ (t_{111}, x_{111})_1, (t_{111}, x_{111})_2, \ldots\}$	$t_{111,AVG}$
							r_{111}
						$M_{T1,C1,n} = \{ (t_{11n}, x_{11n})_1, (t_{11n}, x_{11n})_2, \ldots\}$	$t_{11n,AVG}$
							r_{11n}
			$C_{T1,2}$	$\|M_{T1,C2}\|$	$kC_{T1,2}$	...	...
			...	...	...	...	...
T_2	$\|C_{T2}\|$	kT_2	...	...	...	...	...
...	...	...	...	...	...	...	...

Calculation of programmer's karma based on importance of components

I. construction of a graph of method dependencies from code of a project and calculation of PageRank score for each method
II. 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

Calculation of the karma value:

$$kv_{p_j} = \sum_{i \in M_{p_j}} PRS_{M_i} * \frac{LLOC_{M_i,p_j}}{LLOC_{M_i}}$$

PRS - PageRank score
LLOC - Logical Lines of Code
M_{pj} - a set of all the method-IDs which programmer p_j (co)authored

Reputation ranking

I. a programmer enters a query
II. an ordered list of relevant methods is retrieved based on calculating a cosine distance between programmer's query and all methods (at least one query concept occurs in each method)

Calculation of the ranking score for each method candidate:

$$score_{M_k} = \text{cosSim}_{M_k} + d * \left(@kh_{\hat{p_A}} M_k + @kv_{\hat{p_B}}\right)$$

$@kh_{\hat{p_A}} M_k$ - the maximal know-how score for M_k
$@kv_{\hat{p_B}}$ - the maximal karma value for for M_k

kuric@fiit.stuba.sk
www.fiit.stuba.sk/~kuric
bielik@fiit.stuba.sk
www.fiit.stuba.sk/~bielik

STU FIIT · iit.src · PeWe@FIIT personalized web group · PerOnIK Personalized Conveying of Information and Knowledge