Developers' Cooperation based on Terms of Project Description

Štěpán Minks, Jan Martinovič, Pavla Dráždilová, Alisa Babskova, and Kateřina Slaninová

VŠB - Technical University of Ostrava,
Faculty of Electrical Engineering and Computer Science,
17. listopadu 15/2172, 708 33 Ostrava, Czech Republic
{min111,pavla.drazdilova,jan.martinovic,alisa.babskova.st,
katerina.slaninova}@vsb.cz

Abstract. Very interesting specialized web portal for collaboration of developers is CodePlex [1]. Registered users can participate in multiple projects, discussions, adding and sharing source codes or documentations, issue a release, etc. In the article we deal with strength extraction between developers based on their association. The research presented in this article is motivated by our previous work [10]. From this paper we have used the approach for extraction of initial metadata, and we have used modified Jaccard coeficient for description of the strength of associations between developers. Method is usable for creation of derived collaborators network, where as input is used the set of words, which will describe the network (the developers used these words in project description).

1 Introduction

In the library science, keywords are used to describe the theme of the book and its inclusion in the catalog, mostly controlled by selecting words from the register. By using keywords, it is possible to search for books with similar content. In the same way, the keywords are used on the Web Search for websites with specific content.

Recently the concept of social networks and online communities is becoming still more and more popular. As a result, the number of their users significantly increasing. Reasons for communication between people and creation of social networks in our time are various: study, dating, travelling and tourism, work, games and programming is not the exception.

Many programmers on the Internet are looking for interesting ideas, or assistance when implementing their own solutions. Online collaboration is no longer a novelty in our times and it is run by people all over the world. However, searching for suitable and capable people who could implement a particular idea at reasonable deadlines and high quality is an eternal problem.

OSS (Open Source Software) is a example of a dynamic network, as well as a prototype of complex networks emerging on the Internet. By working through the Internet, interactions between developers can be considered as relations in the synthetic network

[1] CodePlex: http://codeplex.com

J. Pokorný, V. Snášel, K. Richta (Eds.): Dateso 2012, pp. 38–48, ISBN 978-80-7378-171-2.

of collaborators. These relations arise when the developers join the project and begin to communicate with others. OSS network consists of two entities - developers and projects. An examples of such OSS social network established on the basis of interaction between the participants is CodePlex.

In this paper we try to determine the strength of relationship or similarity between CodePlex developers in the context of projects they work on. To determine the context, we used project key words, which in the case of the CodePlex are extracted from project descriptions.

Some related work dealing with the terms extraction in the social network. In the article [11] author illustrate ontology emergence by a novel method for the extraction of community-based ontologies from Web pages. Other approache is in the articel [12], where authors examine the dynamics of social network structures in Open Source Software teams but data were extracted monthly from the bug tracking system in order to achive a longitudinal view of the interaction pattern of each project.

2 CODEPLEX

CodePlex is a specialized web portal operated by Microsoft. It is mainly used by developers for collaboration on projects, sharing source codes, communication and software development. Generally, registered users can participate in multiple projects, discussions, adding the source code and documentation, issue a release, etc. Some of the users have defined a specific role within the project for which they work. Each user has his own page, where he can share information about himself, his projects on which he currently works, and the most recent activities. The CodePlex projects themselves can be considered as a very interesting source of information. In addition to the list of users and roles, CodePlex enables register keywords, add description of the project, the number of visits, status, date of creation, url and other information about the project. All activities are carried out on CodePlex by a particular user within a specific project.

Database which was created as a result of data obtained from CodePlex.com, consists of 6 main tables: User - 96251 records, Project - 21184 records, Discussions - 397329 records, RecentActivity - 72285 records, Membership - 126759 records and SourceCode - 610917 records.

In CodePlex, we can see two types of entities: users and projects. Both are represented by tables that contain specific characteristics (see Table 1).

The undirect connection between the user and the project is implemented through activities within the scope of the project. These activities are in the database CodePlex divided into different types: SourceCode, Discussion, RecentActivity and Membership.

- In the **SourceCode**, there are records about added projects.
- **Discussion** describes discussions about the project and the responses of individual users.
- **RecentActivity** records activities such as check-in, task records, add project to the Wiki information, note about Release version etc.
- In the table **Membership**, we are able to trace the users' participation in the projects and their assigned role in them.

Entity	Attribute	Type
User	login	character(32)
	personalStatement	character(255)
	createdOn	date
	lastVisit	date
	url	character(255)
Project	nameProject	character(255)
	tags	text
	createdOn	date
	status	character(255)
	license	character(255)
	pageViews	integer
	visits	integer
	url	character(255)
	description	text

Table 1. Tables User and Project

We can represent CodePlex as a bipartite graph of users and projects, where the edge between the user and the project is a user's activity in a project.

The Tables User, Project and Activity store in the CodePlex database information about time of occurrence, as well as the last modification or the last visit. Time of creation or last modification is not defined for all activities. Membership activities have no time component and Activities, SourceCode and RecentActivity do not track or change the last visit.

The execution time for casual activity component is often defined by verbal description, such as Today, Last Week, Monday. This data format is not suitable for analysis, and so the time was ignored for component-type Activities, Discussions and SourceCode. Projects and users have the time records in the correct format. Furthermore, all entities were analyzed for a better overview what data are available.

For each table were found a maximum and minimum time values of individual entities (users, projects and activities). We introduce a table of data creation and changes in the CodePlex database (see Table 2).

	createdOn/InsertTime		lastVisit/updateTime	
Entity	max	min	max	min
User	6/3/2011	14/4/2006	20/3/2011	15/1/2009
Project	28/2/2011	28/4/2006	not defined	
Discussions	can't be defined			
RecentActivity	20/3/2011	19/1/2011	not defined	
Membership	not defined			
SourceCode	can't be defined		not defined	

Table 2. Table Project

If we look at the data that we have in the Table User, we are not able to define the user's profile. It consists of the field of interest, what he deals with, the programming language he uses and at what level. PersonalStatement attribute is used to describe the user, but from the total set of our users downloaded, there was not a single one, who would fill it up. On the other hand, the project has enough information defined – which fields are concerned, how long it lasted, whether it is completed, which technology it is used, etc.

The main attribute, carrying the largest set of information, is the project Description – the description of the project itself.

Using activities such as user links to the projects, we are able to determine with some probability an area of specialization and a work of each user. For example, if a user is working on three projects written in .NET and one in Java, we could include him in .NET programmers with high probability, and less likely recommend him as a Java programmer.

In other words, terms or description of the project may not only help us to provide more information about projects, but also to determine the user's area of interests or abilities. As a result, the way we are able to compare user attributes determines the similarity to other network participants.

3 How to Construct Graph of Collaborators

Whenever we think about collaboration between two persons, we not only look at the relationship itself, but also at the context. It is clear that depending on context, the strength of relationship changes. Therefore, we divide collaboration into two main parts *Persons' Relationship* and *Relationship Context*.

Comparing with definition in article [10], where the basis is relation between a person and a term, and another colleague is seen as a context, we now consider the relation between persons as a main part, while the term describes the context. Although the computation process is almost the same, we think this reflects the reality better.

When we describe collaboration as a part of reality, we always start with defining main set of collaborated persons P. Although persons P could in fact represent any object in reality, process was designed specifically for the real persons.

Persons has additional attributes. Usually it could be publications, teams, organizations, projects, etc. We called it attribute domain. Let us define a sample organizations set as $D_O = \{Microsoft, Oracle, IBM, \ldots\}$. To specify organizations of person P_i. We can then specify the set of all persons' organizations as attribute set $O = \{O_{P_0}, O_{P_1}, \ldots, O_{P_n}\}$. If we want to express that person has some attribute, we create a subset from set which defines all the possible values. For example we can create a subset $O_{P_i} \subseteq D_O$, so O_{P_i} could be $\{Microsoft, Oracle\}$.

Generally, let D_X be a set of attribute domain, then X are attributes for all persons P, where object $X_{P_i} \in X$ is one person's attributes described as $X_{P_i} \subseteq D_X$.

3.1 Persons' Relationship

We describe a persons' relationship as commutative operation $\bullet$ on cartesian product of person's attribute $X\,x\,X$, where output is mapped to the set of real numbers $\mathbb{R}$.

$$Attribute_{Score}(X_{P_i}, X_{P_j}) = X_{P_i} \bullet X_{P_j} \in \mathbb{R} \tag{1}$$

An easy implementation of operation $\bullet$ are standard set operations like intersection or union. To get a real number, we just compute cardinality. Jaccard coefficient is the most typical operation we can use:

$$Attribute_{Score}(X_{P_i}, X_{P_j}) = \frac{|X_{P_i} \cap X_{P_j}|}{|X_{P_i} \cup X_{P_j}|} \tag{2}$$

It is clear to see that no matter what order of cartesian product we use; the result is the same. Other implementations could be simple matching coefficient, mutual information, Dice coefficient, overlap coefficient and many others.

Listing 1.1. Exported CodePlex data from XML file.

```
<codeplexProject key="...">
        <developer>hongmin0813</developer>
        <developer>lengleng3898</developer>
        <developer>lengleng38982nd</developer>
        <description>???</description>
        <year>29.4.2010 0:00:00</year>
        <url>http://lengleng3898.codeplex.com/</url>
        <meta>SS = 0</meta>
        <meta>RA = 0</meta>
        <meta>D = 0</meta>
        <meta>M = 0</meta>
</codeplexProject>
```

Applying to the projects in CodePlex, the base set Developers D is chosen as persons at first. We read sequentially the whole file and create sets D and CodePlex project CP as an attribute. For recording, we firstly read the author of the project, and if he is not in the set D, we add him as well the project he is working on, into CP_{D_i}. CP consists of all person's projects $\{CP_{D_0}, CP_{D_1}, \ldots, CP_{D_n}\} = CP$. After the whole analysis of the file, we can define AttributeScore computation for CodePlex:

$$Attribute_{Score}(CP_{D_i}, CP_{D_j}) = \frac{|CP_{D_i} \cap CP_{D_j}|}{|CP_{D_i} \cup CP_{D_j}|} \tag{3}$$

3.2 Relationship Context

As we discussed above, every person has it's attributes. Moreover, each person has a description text. If we use lexical analysis on this text, we can define a term set (or a m-gram set) for every person as T_{P_i}. Term set T consists of all persons term sets $\{T_{P_0}, T_{P_1}, \ldots, T_{P_n}\} = T$, when the domain for terms D_T could be easily obtained as union of all terms extracted for each person $D_T = T_{P_0} \cup T_{P_1} \cup \ldots \cup T_{P_n}$.

The whole process of obtaining term sets is described in [10], so we just remiding $(t_k \; in \; T_{P_i})$ stands for the number of terms t_k in the titles of articles by T_{P_i} and $(t_k \; in \; T)$ for the number of terms in titles in all articles.

We can evaluate association between the selected term $t_k \in D_T$ and a person $P_i \in P$:

$$R(T_{P_i}, t_k) = \frac{(t_k \; in \; T_{P_i})}{(t_k \; in \; T) + |T_{P_i}| - (t_k \; in \; T_{P_i})} \tag{4}$$

$$R_{Norm}(T_{P_i}, t_k) = \frac{R(T_{P_i}, t_k)}{MAX(R(T_{P_i}, t_1), \ldots, R(T_{P_i}, t_{|T_{P_i}|}))} \tag{5}$$

Evaluation of the whole relationship context of two persons P_i and P_j has two steps. First, we compute association between P_i and selecte term t_k, and between the second person P_j and t_k separately. Afterwards, because each part is already evaluated by real number, we combine both results in the same way; we can combine the whole result in equation one. However, the most usual is again muliplication, so we could write:

$$Context_{Score}(T_{P_i}, T_{P_j}, t_k) = R_{Norm}(T_{P_i}, t_k) \, R_{Norm}(T_{P_j}, t_k) \tag{6}$$

In CodePlex we see the description text for the developer as the all description of all projects he is working on, joined together.

$$Context_{Score}(T_{D_i}, T_{D_j}, t_k) = R_{Norm}(T_{D_i}, t_k) \, R_{Norm}(T_{D_j}, t_k) \tag{7}$$

3.3 Collaboration – Whole Score

The last step is to define Score, which consists of $Attribute_{Score}$ and $Context_{Score}$:

$$Score(X_{P_i}, X_{P_j}, T_{P_i}, T_{P_j}, t_k) = Attribute_{Score}(X_{P_i}, X_{P_j}) \, Context_{Score}(T_{P_i}, T_{P_j}, t_k) \tag{8}$$

We obtain for CodePlex:

$$Score(CP_{D_i}, CP_{D_j}, T_{D_i}, T_{D_j}, t_k) = Attribute_{Score}(CP_{D_i}, CP_{D_j}) \, Context_{Score}(T_{D_i}, T_{D_j}, t_k) \tag{9}$$

3.4 Building the Graph

To describe the network of collaboration, we use standard weighted graph $G(V,E)$, where weighted function is defined as $w : E(G) \mapsto \mathbb{R}$, when $w(e) \geq 0$.

The determination of set V is generally simple, because objects of vertices set V match with objects of set P, so $V = P$. However, we can do the same with all the possible pairs from set P to assign a set of edges E; it is better to design the algorithm to each implementation at first, and to reduct the number of useless computations. In addition, we must choose term t_k for function w, which reflects the context. Because only the commutative operations are used, we do not need to take into consideration the order of attribute objects in function parameters. Moreover E is two-object set, where the order of objects does not matter, so the evaluating is done just once.

When we construct graph based on developers' projects relationship, we use $Attribute_{Score}(CP_{D_i}, CP_{D_j})$ as w, where no term is needed, then simply $V = D$, which means that every developer is a vertex in the graph. Then, for each developer $D_i \in D$ we find collaborators D_{i_C} and for each collaborator $D_j \in D_{i_C}$ we create two-object set $\{D_i, D_j\}$, which corresponds with an edge in the graph. Equation 3 is then used to evaluate the edge.

The function $Score(CO_{D_i}, CO_{D_j}, T_{D_i}, T_{D_j}, t_k)$ is used for evaluating the edges in the context of the term. The only difference is, that majority of developers has not chosen term in their description text, so the result will be 0 and no edge would exists. Hence, we first determine subset of developers $D_{t_k} \subseteq D$ for those that have a term in their description text, followed by the same steps described in the last paragraph to compute developers' projects relationship. Then, the term t_k is used for computation of the second part in $Context_{Score}(T_{D_i}, T_{D_j}, t_k)$. Finally, we calculate the whole *Score* by multiplication of both parts.

4 Experiments

For the basic computation of the collaboration, we chose the term "team" and apply it to the formula 5. The results were limited to the collaborators with whose the person has worked together on the project at least once. We show in the Table 3 values of $Attribute_{Score}$ for person with nickname CareBear and in the Table 4 for person with nickname shanselman.

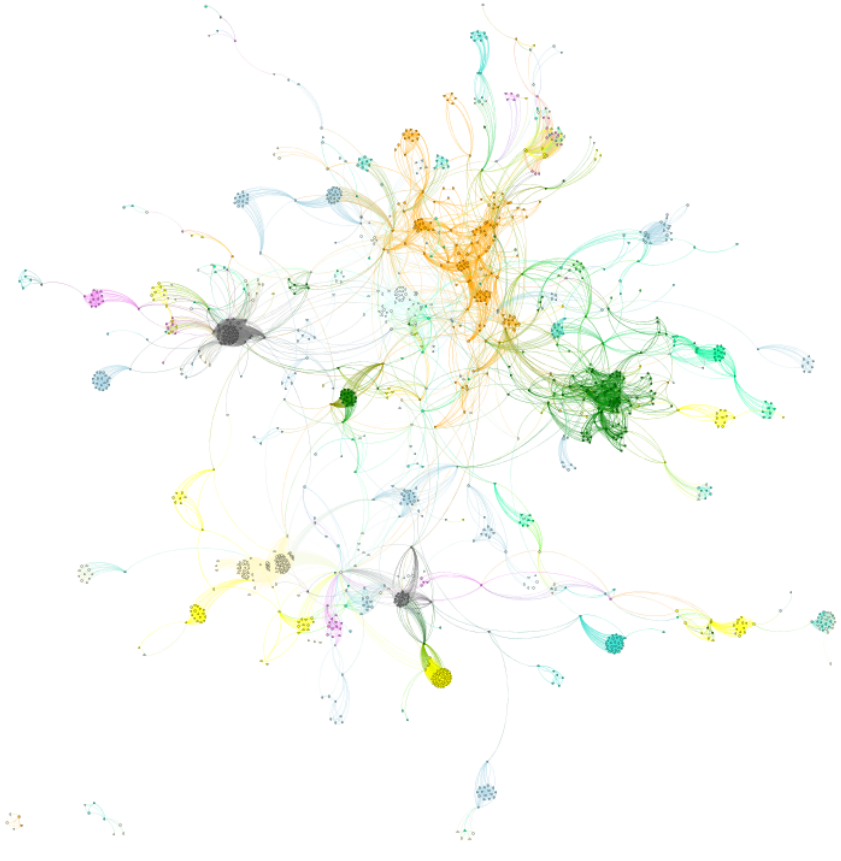

Fig. 1. Synthetic collaborators network for the term *team* - edge weights are computed by *Score*

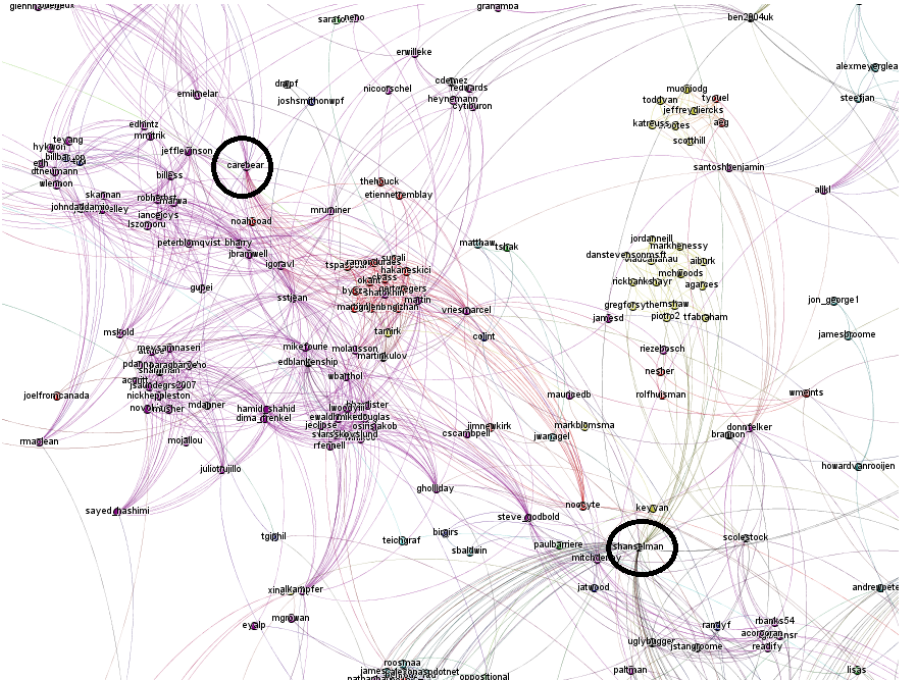

Fig. 2. Selected subnetwork with developers Carebare and shanelsman

Number	Coworkers	Projects	Common projects	$Attribute_{Score}$
1	CareBear	13	13	1
2	EmilMelar	2	2	0,1538462
3	Maggie	2	2	0,1538462
4	Kudzu2	2	2	0,1538462
5	kudzu	12	3	0,1363636
6	hhariri	6	2	0,1176471
7	amccool	1	1	0,07692308
8	arundeep	1	1	0,07692308
9	badmaash	1	1	0,07692308
10	frasse	1	1	0,07692308
...				
145	shanselman	20	1	0,03125
146	Microsoft	537	1	0,001821494

Table 3. Coworkers of CareBear

We can immediately notice that even though shanselman do not participate on many projects with CareBear (they have one common project), the AtributeScore is 0.03125. Conversely then, although shanselman (or CareBear) has with Microsoft 4 common projects, Microsoft cooperate with many other persons. Therefore, the shanselman (or CareBear) has not such a strong AtributScore with Microsoft.

Number	Coworkers	Projects	Common projects	$Attribute_{Score}$
1	shanselman	20	20	1
2	Haacked	14	3	0,09677419
3	jongalloway	14	3	0,09677419
4	SteveSanderson	4	2	0,09090909
5	shahineo	4	2	0,09090909
6	JasonHaley	6	2	0,08333334
7	ben2004uk	7	2	0,08
8	bsimser	8	2	0,07692308
9	AArnott	8	2	0,07692308
10	dcazzulino	20	2	0,05263158
11	agsmith	1	1	0,05
12	BartRead	1	17 0,05	
...				
147	CareBear	13	1	0,03125
...				
152	ReedMe	31	1	0,02
153	Microsoft	537	4	0,007233273

Table 4. Coworkers of shanselman

4.1 Key Terms Computation

At first, we have calculated the keywords for the CareBear and shanselman. We have selected only the first 15 terms for illustration (see Table 5 and Table 6). For comparison we marked some terms (bold text).

number	t_k	t_k *in* $TP_{CareBear}$
1	flickr	1
2	cosmo	0,9894736
3	automaton	0,9415065
4	ovik	0,8872708
5	downloadr	0,8238943
6	weeb	0,7605178
7	scrum	0,571498
8	photo	0,5248883
9	**team**	0,4844927
10	tf	0,404665
11	associ	0,3172817
12	flickr downloadr	0,3168824
13	**store**	0,3070892
14	process templat	0,2949297
15	team foundat	0,2901235
...		
926	set	0,007442362

Table 5. Key Terms for the person CareBear

number	t_k	t_k *in* $T_{P_{shanselman}}$
1	syndic	1
2	administr	0,75
3	peer	0,6934211
4	creatur	0,6413794
5	terrarium	0,5994475
6	argot	0,5570145
7	reflector	0,5359043
8	ecosystem	0,5102881
9	nuget	0,3803681
10	browser	0,3582435
11	assembl	0,3461412
12	consum	0,3358614
13	mobil	0,3216281
14	ad	0,309705
15	nerddinn	0,2954391
...		
82	**team**	0,1306442
...		
174	**store**	0,08637504
1444	system	0,007958922

Table 6. Key Terms for the person shanselman

In the Figure 1 is whole network of collaborators for the term *team*. Here is 31 connected components (communities) with collaborating developers. Figure 2 shows graphs of synthetic collaborators network generated for the term "team" and for selected developers.

5 Conclusion

Research presented in this article is oriented to the strength extraction between persons based on their context in the CodePlex. The method was presented using the data collection from the CodePlex database, which contains information of the activities of developers in the project. The proposed method is usable for the development of collaboration network. The description of this network is based on the set of terms (as the input), which are used in the description of projects by the given developer. Using this method, we have obtained the new weight in the synthetic collaborators network. By means of the set of selected term, belonging to one (or more) persons, we can construct the subnetwork with only the context-related collaborators. This subnetwork can be very helpful in searching of the persons who are interested in the same area, defined by the selected term. It is usable for members of the project management, who need to find suitable developers specialized to certain area. It follows that this method can be used to a certain extent for prediction as well.

Acknowledgment

This work was supported by SGS, VSB – Technical University of Ostrava, Czech Republic, under the grant No. SP2012/151 Large graph analysis and processing.

References

1. E. Deza and M. Deza. Dictionary of distances. 1-391, 2006.
2. Ch. Jacquemin and B. Didier. Term Extraction and Automatic Indexing. Handbook of Computational Linguistics. Oxford University Press, 599-615, 2003.
3. M. Konchady. Text Mining Application Programming (Programming Se-ries). Charles River Media, Rockland, MA, USA, May 2006.
4. A. H. Lashkari, F. Mahdavi, V. Ghomi. A Boolean Model in Information Retrieval for Search Engines, 2009.
5. P. Lopez and R. Laurent. HUMB: Automatic Key Term Extraction from Scientific Articles in GROBID. Computational Linguistics July, 248-251, 2010.
6. J. Mori, Y. Matsuo, M. Ishizuka, B. Faltings. Keyword extraction from the Web for FOAF metadata, In Proceedings of the 1st Workshop on Friend of a Friend, Social Networking and the (Semantic) Web, 2004.
7. M. F. Porter. An algorithm for suffix stripping. Program, 14:130-137, 1980.
8. Y. Ding. Scientific collaboration and endorsement: Network analysis of coauthorship and citation networks. Journal of informetrics 5.1,187-203, 2011.
9. R. R. T. Santamaría. Overlapping Clustered Graphs: Co-authorship Networks Visualization. Lecture Notes in Computer Science 5166, 190-199, 2008.
10. S. Minks, J. Martinovic, P. Drazdilova and K. Slaninova. Author Cooperation based on Terms of Article Titles from DBLP, IHCI 2011, Praha, 2011.
11. P. Mika. Ontologies are us: A unified model of social networks and semantics. Web Semantics Science Services and Agents on the World Wide Web 5, 5-15, 2007.
12. Y. Long, K. Siau. Social Network Structures in Open Source Software Development Teams. Journal of Database Management 18, 25-40, 2007.