EVOLUTION OF SOCIAL DEVELOPER NETWORK IN OSS: SURVEY

Anil Kumar[1]

[1]*PG Scholar, Galgotias University, Greater Noida, UP, India*

Abstract

Social Developer Networks are built of complex hierarchical social network which forms relationship information between software entities. Recent research trends suggested many different successful methods on Social Developer Network to analyze its growth and advancement of network structure. Aim is to present brief survey on the analysis, method, algorithm used in developer social network and techniques and technologies available to analyze performance of the developers associated network. Further discuss the growth, extension of its research field, existing methodologies and approach, extended tasks and techniques in developer social network and OSS.

Keywords: *Social Developer Network, Open Source Software, OSS, Social Network, Community Structure, developers, bug resolving process, defects*

---***---

I. INTRODUCTION

Social Developer Network (SDN) is building block of software entities network and combine several software project entities. Social Group can be reinterpreted as SDN which is behavior of several node-edge relationships. It consists of nodes for example Developer, Project, File, Author, Watcher, User and edge which represent relations as works, mange, commit, comment, bug-report etc. These developer networks are illustrated graphically to visualize network relationship where nodes are represented having core role in the network and edge are represented as task performed. We perform network analysis by performing operations on dataset, execute algorithm and showing node-edge relational activities through visual graphical representation. While analyzing social network, Central node measurement, Ranking pages and prioritizing measures comes into picture. Several measure exists to analyze hidden network properties. SDN have been interesting area over past several years. Analyzing social developer network might point out experts in understanding the social structure and assists new comers or active managers to manage their social activities.

Our paper deals with single point views about previous and current activities in the Social Developer Network (SDN). Growth of open access project movement started several years ago. Continuous growth of software business activities in demonstrates financial success . Several software product vendor like Google, RedHat are active organization to facilitate OSS. Categorizing several different platform of OSS for example Operating System (Linux), DBMS (MySQL), Web Content Management (Zoomla and Wordpress), Business Software tool (Magento).

2. AN OVERVIEW OF SOCIAL DEVELOPER NETWORK TYPES

2.1 Heterogeneous Social Network

A heterogeneous social network holds topological information and relational information and consists of more entity types. Each vertex is recognized using its neighbor relations. A relation is combination of directly and indirectly connected nodes and connectors. Heterogeneous Social Network HSN(Vertices, Edge, Label) is directed labeled graph, where Vertices is a limited group of nodes, Label is a limited set and Vertices × Label × Vertices is a limited group of edges [9]. Most algorithm for example Meta-path analysis, ranking, similarity analysis and group similar activities are operation mainly performed in this network.

2.2 Homogeneous Social Network

Homogeneous social network assumes single type of vertices and edge relations. This kind of analysis method generally produces loss of information in the process. In homogeneous social network, vertices denote entity and edge shows relations. Available algorithm and methods example ranking, similarity search, clustering and group similar activities and association relation prediction

2.3 Multidimensional Social Network

Denotes multiple kinds of relationships Multi-dimensional network containing every dimension constituting user association at each site e.g. Facebook, Twitter, YouTube and Orkut etc Lei Tang et al. [13] Users relatedness with each other on several activities leads to multiple network . Xutao Li et al. [26] establish community relation in multi-dimensional network by calculating the similarity between two items in same dimension (entity) or different dimension (entity) from the network based on probability distribution of each dimension or entity. Their result shows that purposed algorithm is effective and efficient.

3. LITERATURE SURVEY

Social developer network has proceeded in a decade. In this survey paper, I intended to show the related work of crucial author's views and their contribution towards social developer networks. Overall survey can be categorized in different sections like community structure detection, bug localization, social developer network and collaboration in bug fixing process, bug triaging.

Key Components

- Social Network and Community in OSS
- Social Developer Network and Bug Resolving Process
- Bug Prediction
- Bug Localization
- Bug Triaging

3.1 Social Network and Community in OSS

W Scacchi [4] studies the expected relationship in social structure of development team and source code in open source software. Open source software does not always adhere traditional software engineering practice but kind of complex structure of socio-technical processes. Q Hong et al. [1] discover that developers and their relationships in social developer network change continuously made in core developer team. They compared structure and evolution of developer as coder social network with popular general social network (GSN) such as Twitter, Facebook and Amazon. Their study also analysed how developer social network grow over time and effect of DSN and growth of topological relation of the developer network. Their results reveal strong evidence of the community structure, maintains world characteristics over time and denote a gradual enhancement of community structure inside developer social network. Andrejs Jermakovics et al. [11] Approach for visualizing software developer network for V C S repositories. They calculates similarity among developer based on similar file modification. Built collaborating developers network and applies filtering methods to increase the modularity of the developer network. They assign weight to all files and compute developers similarity. Zhou et al. [12] analyzed different dimension of software modification including change significance or source code dependency levels. They took a set of features from the two different source and proposed a Bayesian network for change prediction. Merging close changed entities and dependency relation, approach can predict underlying uncertainty. The study on two medium-sized OSS project indicates the feasibility and productiveness of our approach compared to previous work. Matthew Van Antwerp and Greg Madey [10] describes significant consideration of OSS in projects popularity. Their results show existing developer-developer relations are mark of previous and coming project popularity. M A Balieiro et al. [14] developed user-defined functional option called OSS-Network, which performs study on network communities using social network. OSS-Network assists to decrease the difficulties of researchers in code, bug resolving process, and user support. Provides an user defined and functional option of retrieving data, performing observation and analysis. Thung Ferdian et al. [7] They investigated source code construction forming network in GitHub. Distinguished leading developers and projects GitHub sub-network. It helped developers in understanding developers and projects association. Found out relationship between projects, developers and influential projects . They analyzed that projects networks in GitHub. Project networks only need common developer establishes relation between projects. Finding influential projects and developer on the basis of page rank gives more developer with many projects. Their results shows that distribution of project-project network graph generally follows power law while developer-developer does not. P Bhattacharya et al. [8] Visualize evolution in software project . Analyzed software evolution and build forecaster to facilitate development work and maintenance. Their results analysis on eleven major projects including Eclipse, Firefox, shows forecaster can detect significant structural modification. It might assist developer to measure bug severity, prioritizing dub efforts and predict defects-prone release. Nicolas Lopez [5] explored topic-wise model can assist understanding evolution of software system. He studied evolution of topics taking developers relation in account. He analyzed topic pattern of developer project works. Considered modification that expand OSS projects in eco-systems. His research demonstrates that developers usually take part in multiple projects that relates ecosystem and assists developers to get experienced working on specific topic. Hierarchical directories structure and files of software program is pin point for research work

3.2 Social Developer Network and Bug Resolving Process

K. Crowston and J. Howison [2] studied people of network in bug resolving process. They formed social structured links between developers on their several co-occurrence information in bug reports in the bug checking system. Their work targets 120 different projects from SourceForge to study the centralization problem and aims to gain communication pattern in Open software system projects. Result shows software team varies in communication centralization in the project. Amit and Avdesh [6] studied evolution and growth of social developer network for Eclipse project using Bugzilla database. They demonstrated how network property like Average Path length, Modularity etc has impact on attributes characterizing productiveness of bug resolving process. Result demonstrate good relation between developer network and properties distinguishing bug resolving process. Igor Steinmacher et al. [3] presents study on new comers joining projects process since they are important and potential contributor to the project growth and productivity. Their study analyze the difficulties faced by newly joined employee and tried to verify politeness, usefulness of the queries or replies received by the new comers in the communication with team manager. Their results shows that retention rate is as below as 18% in the mailing list and 13% in the issue manager and pointed out some possible cause for leaving the project . Overall analysis shows new comer are not much interested to join project due to lack of proper response or lack of clarification on specific doubt. They tried to understand how developers collaborate on project and how the newcomers behavior changes in communities because it helps management to take decision on retention. Yasutaka Kamei et al. [16]. identified developers resolving bug who takes considerable time to drill down function or some part of source code to review, fix and test it even after critical files are known. Developers can recheck risky changes at time of updating. They studied six open source and some business projects from various domains based on characteristics of software change, number of added lines and developer expertise.

3.3 Bug Prediction

Zimmerman and Nagappan [17] A dependency graphs compared relation between source code. Inexperienced practitioner have less knowledge. Predicting failure in post release lacks effective allocation of resource and inexperience in code complexity. Managers recognize expert units to tackle critical defects. Their work was examined on server platform. Recall result was 10 percent greater than expected for complexity matrix model over network measures. In the network measure, developers experienced critical twice of complexity metrics. Kim Herzig et al. [18] All predicted bugs are not expected actual defects but results in new features. Bug reports are classified wrongly due to misclassification of defects and feature. S. Shivaji et al. [19] demonstrate better bug prediction methodology. Implemented feature identification technique in fault prediction. They investigated several features collection methods to improve performance. Implemented significance attributed evaluation filter method with Bayes classifier.

3.4 Bug Localization

The basic concept behind the Bug Localization is to calculate statistics characterizing a software program's runtime behavior over multiple executions. Erroneous source code contains critical bug which results in failure at runtime. It causes increased in resource cost and decreased productivity. Bug localization assists program manager and practitioner to identify source code and resolve bug in minimized time to increase productivity. Localizing or locating defects in the code have been challenging task. Zimmerman et al. [15] Analyzed Bugzilla and C V S to predict bugs. Differentiated defects between bug repository and each code locations in eclipse project . Figured out pre and post release. calculated total defects based on each location since source location was fixed. Experiment shows complex codes throw higher defects. Dongsun Kim et al. [20] presented two-phased defect assumption model. Checks whether input bug found in existing file. Results shows 70 % of the assumption were correct to point bug containing files. [Wong et al., 2007] proposed many spectra metrics, assign less weight to pass test case if particular statement executed by more pass test cases. Used Siemens Test tool as benchmark. [XiaoYuan Xie et al., 2010] proposed a post-ranking of program statements evaluation. They grouped statements into two suspicious groups according to heuristics.

3.5 Bug Triaging

A bug information is assigned, developer verify and resolves the bug. Potential developer is promoted with new issue. Gaeul Jeong et al. [28] purposed markov chain graph model to capture bug reassigned history. It discovered team hierarchy and selected appropriate developer for new activities. Their model raised the prediction accuracy by 23 percent in compare with traditional bug triaging approach. John Anvik and Gail C. Murphy [29] demonstrated approach to generate recommenders for time-saving development process. Their approach results in more accurate when performed on five open source code projects in bug resolving process. Weigin Zou et al. [30] used feature and instance selection technique for increasing correctness of bug triage. They analyzed on Eclipse data set. Results shows new and small training set can generate better accuracy than the existing original one.

4. VISUAL ILLUSTRATION OF SOCIAL DEVELOPER NETWORK

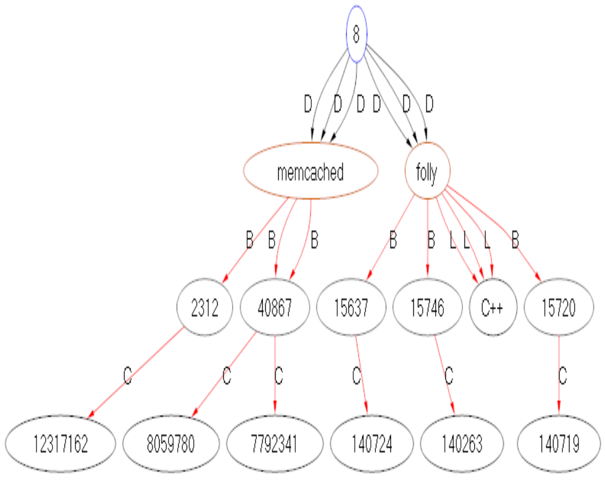

Fig 1 Social Developer Network

Above directed network graph contains five distinct types of labeled node, edges and relationships. This representation uses developer network to show social relationship among several distinct node. Visual representation can be understood from Top node as Developer node followed by Project node, Bug node and Comment node respectively. Relationships between nodes as Developer->Project->Bug->Comment and Developer->Project->Language. Developer to Project (labeled as D), Project to Language (labeled as L), Project to Bug (labeled as B) and Project to Comment (labeled as C). Depicted social network diagram demonstrates that Developer Id (8) work on Project (memcached) and (Folly), write code in language (C++), Bug Ids (2312, 40867) and (15637, 15746, 15720) respectively having several comments in each projects.

5. SUMMARY SOCIAL DEVELOPER NETWORK TECHNIQUES AND PURPOSED APPROACH

Table 1: Summary of surveyed information of several author's tasks, techniques.

Characteristics	Data Source	Method	Activities	Approach
Bug Reports Information	Bugzilla dataset	Classification Supervised Learning	Bug Triaging	Anvik
Bug Report	Five OSS Projects	Machine Learning	Bug Triaging	John Anvik and Gail C. Murphy [29]
Bug Report	Bugzilla	Complexity Metrics	Predict Post-release failure	Zimmerman and Nagappan [17]
Source Code	Mozilla, Apache	Classification	Bug prediction	Kim Herzig et al. [18]
File and Function Variable	Source code and CVS	Metadata analysis	Change Prediction	Hassan and Holt
Source code and Bug Report	Bugzilla and CVS	Bug tracking system	Bug Localization	Zimmerman et al. [15]
Bug Report	Mozilla and Eclipse	Markov chain graph model	Bug Triage	Gaeul Jeong et al. [28]
Bug Report	Eclipse	Feature, instance selection	Bug Triage	Weigin Zou et al. [30]

6. CONCLUSIONS

Social Developer Network and OSS are compatible and distributed. OSS is distributed software environment. OSS facilitate reduced cost, good performance, better quality, reduced time, increased security in critical operations and project management economically and financially. Social Developer Network community structure built to make relationship between several open source project. This survey demonstrated different developer networks, developer relationship and associated relationship with bug resolving process, localization of bug and predicting bug in social developer network.

REFERENCES

[1] Qiaona Hong, Sunghun Kim, S.C. Cheung, "Christian Bird, Understanding a Developer Social Network and its Evolution", Proceeding ICSM '11 Proceedings of the 2011 27th IEEE International Conference on Software Maintenance, Pages 323-332

[2] K. Crowston and J. Howison, "The social structure of free and open source software development," First Monday, vol. 10, no. 2, 2005.

[3] Igor Steinmacher, Igor Wiese, Ana Paula Chaves, Marco Aurélio Gerosa , "Why do newcomers abandon open source software projects?" ,Cooperative and Human Aspects of Software Engineering (CHASE), 2013 6th International Workshop on IEEE, 2013, page. 25-32

[4] Walt Scacchi, "Software Development Practices in Open Software Development Communities: A Comparative Case Study" In Proceedings of the First Workshop on Open Source Software Engineering, 2001,

[5] Nicolas Lopez, "Using topic models to understand the evolution of a software ecosystem", ESEC/FSE 2013 Proceedings of the 2013 9th Joint Meeting on Foundations of Software Engineering, Pages 723-726

[6] Amit Kumar, Avdesh Gupta, "Evolution of Developer Social Network and Its Impact on Bug Fixing Process", ISEC'13, Feb 2013, 21-23

[7] Thung, Ferdian; LO, David; and JIANG, Lingxiao, "Network Structure of Social Coding in GitHub" (2013). *Research Collection School of Information Systems (Open Access).*

[8] Pamela Bhattacharya, Marios Iliofotou, Iulian Neamtiu, Michalis Faloutsos, "Graph-Based Analysis and Prediction for Software Evolution", Proceeding ICSE '12 Proceedings of the 34th International Conference on Software Engineering, Pages 419-429

[9] Cheng-Te Li and Shou-De Lin, "Centrality Analysis, Role-based Clustering, and Egocentric Abstraction for Hetrogeneous Social Networks" 2012 International Conference on Social Computing ((PASSAT, SocialCom), IEEE, Sept 2012

[10] Matthew Van Antwerp, Greg Madey, "The Importance of Social Network Structure in the OpenSource Software Developer Community", Proceedings of the 43rd Hawaii International Conference on System Sciences - 2010

[11] Andrejs Jermakovics, Alberto Sillitti, Giancarlo Succi, "Mining and visualizing developer networks from version control systems", Proceeding CHASE '11 Proceedings of the 4th International Workshop on Cooperative and Human Aspects of Software Engineering,Pages 24-31

[12] Zhou Y,W¨urschM, Giger E, Gall H, L¨u J., "A bayesian network based approach for change coupling prediction", In Proceedings of the Working Conference on Reverse Engineering (WCRE), IEEE Computer Society: Washington, DC, USA, 2008;

[13] Lei Tang, Xufei Wang, and Huan Liu. Community Detection via Heterogeneous Interaction Analysis. Knowledge Discovery and Data Mining (DMKD), 2012

[14] Marco A. Balieiro, Samuel F. de Sousa Júnior, and Cleidson R. B. de Souza, "Facilitating Social Network Studies of FLOSS using the OSSNetwork Environment", Open Source Development, Communities and Quality IFIP – The International Federation for Information Processing Volume 275, 2008, pp 343-350

[15] Zimmermann, T. Premraj, R., Zeller, A, "Predicting defects for Eclipse", In Proc. 3rd International Workshop on Predictor Models in Software Engineering (Minneapolis, MN, USA, May, 2007), PROMISE'07.

[16] Yasutaka Kamei, Emad Shihab, Bram Adams, Ahmed E. Hassan, Audris Mockus, Anand Sinha, and Naoyasu Ubayashi, "A Large-Scale Empirical Study of Just-in-Time Quality Assurance" IEEE Transactions On Software Engineering, Vol. 39, No. 6, June 2013

[17] Zimmermann, T. and Nagappan, N., "Predicting Defects using Network Analysis on Dependency Graphs," in 29th International Conference on Software Engineering, 2007

[18] Kim Herzig, Sascha Just, Andreas Zeller, "It's not a bug, it's a feature: how misclassification impacts bug prediction", Proceeding ICSE '13 Proceedings of the 2013 International Conference on Software Engineering, Pages 392-401

[19] Shivkumar Shivaji, E. James Whitehead, Jr., Ram Akella, Sunghun Kim, "Reducing Features to Improve Code Change Based Bug Prediction", Journal IEEE Transactions on Software Engineering archive Volume 39 Issue 4, April 2013, Pages 552-569

[20] Dongsun Kim, Yida Tao, Sunghun Kim, Andreas Zeller, "Where Should We Fix This Bug? A Two-Phase Recommendation Model", IEEE Transaction on Software Engineering, VOL. 39, NO. 11, NOVEMBER 2013

[21] Moser, R., Abrahamsson, P., Pedrycz, W., Sillitti, A., Succi,G., "A case study on the impact of refactoring on quality and productivity in an agile team", In Proc. of the 2nd IFIP Central and East European Conference on Software Engineering Techniques CEE-SET 2007, Poznan, Poland (2007).

[22] Huzefa Kagdi, Michael L. Collard and Jonathan I. Maletic1, A survey and taxonomy of approaches for mining software repositories in the context of software evolution, Journal of Software maintenance and evolution : Research and Practice J. Softw. Maint. Evol.: Res. Pract. 2007; 19:77–131

[23] Xiao Yuan Xie and Tsong Yueh Chen and BaoWen Xu "Isolating Suspiciousness from Spectrum-Based Fault Localization Techniques", In Proceedings of the 2010 International Conference on Quality Software (QSIC), 2010, page 385-392, IEEE

[24] Wong, W.E. and Shi, Y. and Qi, Y. and Golden, R. "Using an RBF neural network to Locate Program Bugs", In Proceedings of the 19th International Symposium on Software Reliability Engineering, 2008, page 27-36, 2008, IEEE/ACM.

[25] Santelices, R. and Jones, J.A. and Yu, Y. and Harrold, M.J. "Lightweight fault-localization using multiple coverage types ", In Proceedings of the 31st International Conference on Software Engineering, 2009, page 56-66, 2009, IEEE.

[26] Xutao Li, Ng, M.K., Yunming Ye, " MultiComm: finding community structure in multi-dimensional networks", Knowledge and Data Engineering, IEEE Transactions on Volume:26,Issue: 4, April 2014, pp. 929 - 941, IEEE

[27] Huzefa Kagdi, Michael L. Collard and Jonathan I. Maletic, "A survey and taxonomy of approaches for mining software repositories in the context of software evolution", Journal of Software maintenance and evolution : Research and Practice J. Softw. Maint. Evol.: Res. Pract. 2007; 19:77–131

[28] Gaeul Jeong, Sunghun Kim, Thomas Zimmermann, "Improving Bug Triage with Bug Tossing Graphs", Proceeding ESEC/FSE '09 Proceedings of the the 7th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering , Pages 111-120

[29] John Anvik, Gail C. Murphy , "Reducing the effort of bug report triage: Recommenders for development-oriented decisions", Journal ACM Transactions on Software Engineering and Methodology (TOSEM) TOSEM Homepage archive Volume 20 Issue 3, August 2011, Article No. 10

[30] Weiqin Zou, Yan Hu,Jifeng Xuan ,He Jiang , "Towards Training Set Reduction for Bug Triage",Computer Software and Applications Conference (COMPSAC), 2011 IEEE 35th Annual, Pages 576 - 581