A review of Requirement Elicitation techniques in OSSD

Shehla Saeed[1], Umbreen Fatima[2] and Faiza Iqbal[3]
[123]Department of CS & IT University of Lahore, Lahore, Pakistan

Summary
Open Source projects are getting very popular for more than last three decades. As Open Source Software development (OSSD) environment is globally dispersed, it is very important to determine and understand what particular approaches are being used for requirement elicitation in OSSD and how these approaches are different than traditional requirement elicitation. Requirement elicitation is human-centric approach done by interaction among various stakeholders to find out the needs of a project. Traditional RE process is a centralized approach involving social interaction of stakeholders. Whereas in OSSD stakeholder's interaction are decentralized and dynamic due to geographically dispersed locations of the stakeholders.
The focus of this study is review of requirement elicitation process in globally distributed OSSD. How it deviates from traditional software engineering phases? What challenges are faced during requirement elicitation phase in OSSD? What methods or procedures can be used to overcome these challenges?

Key words:
Requirement engineering, requirement- elicitation, requirement acquisition, open source.

1. Introduction

Trend of developing Open Source Software (OSS) has been growing over the years and revolutionizing the software industry for more than three decades. The term 'Open Source' was first coined in 1998. Cofounder of Foresight Institute of nanotechnology named Christine Peterson first time suggested the term 'open source' (David Bretthaueri, 2001). Open source products are different from freeware, shareware, unrestricted or royalty free software (Fitzgerald-2000).

Requirement engineering is a crucially important part of software development. The main purpose of requirement engineering is to gather requirements for developing quality software.

The cost of requirement engineering is varied from system to system depending on its size and type. For large system it will cost 15% and for smaller system it is 8-10% of the total budget (Sommerville, 1998, 2000).

Various studies attributed requirement elicitation as a cause of failure of software projects ranging from 12% to 71% (Davey & Parker,2015).

More over the studies (Lamsweerde, 2000; Briggs

Grunebacher, 2002; Eberlein & Leite,2002) justify that almost half of overruns and failures are due to poor requirement elicitation process. This outcome is also elaborated in some other studies in Europe (Molokken-Ostvold and Jorgensen, 2003; Boehm, 2000)

This paper is further describing the requirement elicitation techniques used in open source development and how they vary from traditional requirement elicitation techniques.

2. What defines Software as Open Source software (OSS)?

Open source software (OSS) is defined as a software available freely with its source code also with permissions of modifications and redistribution e.g., (Mozilla Firefox) (K.Crowstone, 2012).

OSS definition is bill of rights for computer users. It defines all specific rights granted by software license to be considered as Open source. Linux operating system and Netscape's web browser are successful projects of open source (B.Perens, 1999).

The Open Source Initiative (www.opensource.org) declares 9 characteristics for defining OSS. Three main characteristics are:

- Free distribution of the software
- Availability of source code
- Rights for modification of code.

The other six deals with licensing issues and declare that without any discrimination anyone can use these software for any field (Gracek & Arief, 2004; David Bretthauer, 2001).

OSSD is carried out by globally dispersed developers and contributors with different motivations behind their participation like need for certain features in the product, learning purpose, career development and fun (K. Crowston ,2012; J. Feller, 2000; S. K. Shah. 2006).

In majority of the OSSD the developers are the users of the product (Gracek & Arief, 2004; Scacchi, 2002; Fitzgerald, 2006).

Manuscript received March 5, 2018
Manuscript revised March 20, 2018

Mostly the terms free and OSS are considered the same however these can be differentiated on the basis of licenses assigned to particular software (Fitzgerald,2002,05-Walt Scacchi,2007).

Free software is licensed with GPL whereas OSS has various different licenses including GPL. Free software foundation categorized free software as a social movement while OSS as a particular software development technique (Walt Scacchi, 2007).

The history of free software is very long. Computers were first introduced in universities as research tools. Software was installed freely and programmers were paid for programming jobs not for the programs. When computers entered in the business world, programmers also started the business of their software by restricting the rights of the software and started charges for each copy (B.perens, 1999).

In 1985 ,Richard Stallman established the Free Software Foundation and created the GNU Public License. The FSF policy is “copyleft- all rights Reversed”.

Whereas, OSS is just introduced in 1998.The opensource.org domain was registered in 1999 define the OSS and developed an Open Source Initiative (OSI) certification and declare a list of licenses which meet the standards of open source certification(David Bretthauer,2001).

The common traits of both free software and OSS are source code availability, modification and redistribution to others with rights to insure these freedoms (W.Scacchi, 2007).

3. Adoption towards OSS

Organizations are moving towards OSSD. There are various motivations behind this emerging trend of OSSD. However, the dominant driver behind this motivation is innovation and economic turn

(Monica Divitini, 2003).

In software development OSS is considered as paradigm shift that can solve the “software crisis” (i.e. , development time and cost of software is too much despite that software is not of good quality) The OSS software are more reliable and its rapid release schedule and little or no cost is very striking (Smith and Kollock ,1999).

Software code access, software cost, technological factors, support factors are the main reasons for adoption of OSS in education sector of various countries (Gangadharan, G. R, 2012).

Many developers are attracting towards OSS to get its economic benefits and to get good learning experience as there is no pressure of time and budget schedule (Lakhani KR, 2003; Johnson JP, 2001). Others consider the OSSD as a good experience for managing their career (Lerner J, 2002).

Apache and Linux are the most successful open source projects. However, a lot of others are being used commonly including projects on Internet infrastructure (e.g., bind, sendmail), database systems (e.g., MySQL,POSTGRE SQL), user applications (e.g., OpenOffice), programming languages (e.g., Python, Perl, gcc), and games (e.g., Paradise)

4. Categories of OSSD

OSSD projects are identified in a wide range of domains. OSS researchers categories the OSSD communities into various domains like networked computer games, web infrastructures, higher education, military computing and bioinformatics (Scacchi and Alspaugh, 2008; Scacchi, 2009). Open-source related products and processes are explored in domains such as public policy, law, geography, physics, biology, organization science, art, anthropology, management, economics, and information systems (Scacchi et al. 2009).

Rather than using complex traditional approaches of software development OSSD involves simple development processes and a few resources to produce quality software.

Sourceforge the largest open source services web portal hosting 430,000 projects and connecting 41.8 million customers classify the OSS in communication projects, development projects, multimedia projects, home and educational projects , games, system administration projects, science , security projects and engineering domains.

5. OSS development process

Raymond (1999) uses the metaphor of bazaar for OSSD process like merchants in bazaar OSS developers voluntarily decide how and when to contribute, and cathedral for the traditional software development model where all processing is controlled by a central master. However, this bazaar metaphor was broadly criticized by Bezroukov (1999).

Traditional software development process generically carried out by four broad phases like planning, analysis, design and implementation. However, in OSS development process the first three phases of planning, analysis and design are concatenated and performed by a single developer or small core group. The planning phase is best summarized by Raymond (1999) good projects are developed due to developer's personal itch. This leads to an initial prototype open to globally disperse developers of different domain and abilities to contribute on it. Requirements are taken as understood by the developers without interactions among the developers and users (Fitzgerald, 2006).

As the phenomenon of OSS has grown enormously, researchers have focused their attention to it resulting in a rich base of literature about various issues in OSSD.

Laurent and Cleland (2009) discussed two common OSS models: user-based model and vendor-led approach. In the former, software is developed by collaborating end users whereas in the later approach, a vendor runs development procedure and maintenance activities. Primarily vendor is responsible for development activities of projects however source code is open to the users for improvements.

The focus of this study is requirement elicitation process in OSSD. How it deviates from traditional software engineering phases? What challenges are faced during requirement elicitation phase in OSSD? What methods or procedures can be used to overcome these challenges?

6. Traditional Requirement Engineering Practices

RE phase of software development consist of

following processes: elicitation, analysis, describing requirement, system modeling, requirement validation and management (Sommervillie, 2000; Nuseibeh & Easterbrook,2000; Lucia & Qusef, 2010; Paetsch, Eberlein& Maurer).

Requirements elicitation process involves discovering requirements and identifying system boundaries by consulting stakeholders.

Sawyer and Kotonya (2004) describe requirement elicitation is the process of finding requirements from different source using various elicitation approaches.

Fox expressed requirement elicitation an art of the discovering stakeholder needs for the product

(Fox, 2007).

Output of requirement elicitation phase is a requirement document called SRS.

Requirement Elicitation techniques Zhang (2007) classified requirement elicitation techniques into four categories: conversational, observational, analytical and synthetic. Conversational techniques are face to face interactions like interviews, brainstorming etc. Observational techniques are require analyst for monitoring the work going on like ethnography Analytical techniques where analyst extract information from documentation or existing code like requirement reuse, documentation etc. Synthetic techniques are combination of conversation, observation and analytic techniques like JAD, prototypes, scenarios/storyboard etc. (Zhang, Z., 2007).

Others mentioned hundreds of requirement elicitation techniques are practiced in industry including questionnaire, interviewing, domain analysis, task analysis, card sorting, group work,

goal based approaches, brainstorming, requirement workshops, ethnography, observation, prototyping, joint application development(JAD), apprenticing, scenarios and viewpoints etc. (Zowghi and Coulin, 2005).

An alternative term for requirement elicitation "Trawling for requirements" involves the techniques of business use case, apprenticing, understanding the situations and modeling it, interviewing, solving the right problem, outcomes, business rules, brainstorming, wallpaper, personas, wikis, blogs , family therapy, photographs , discussion forums, document archaeology and viewpoints (Robertson and Robertson, 2006).

Requirement analysis involves the reasoning about requirement, conflicts resolution, prioritize and classify requirements.

Requirements specification is to keep record of elicited requirements in the form of text or diagrams.

System Modeling is developing of system models, including system's environment, system architecture etc. it makes requirements easy to understand. Data flow models, object oriented approaches, semantic data models are mostly used for system modeling (Sommerville, 2000).

Requirement validation confirms that requirement document meets the customer needs. Requirements reviews and requirements testing Techniques are used for requirements validation. Requirement management process includes managing all the information about requirements. Requirements are identified uniquely and various policies are defined for requirement management and traceability.

7. Requirement Elicitation Problems

A detailed literature survey by Davey & Parker (2015) categorized RE problems very concisely into following nine points:

- Various human aspects prohibit communication between client and consultant like person's cognitive limitations, different cultures etc.
- Change of requirement occurs during life time of project.
- Incomplete requirements
- Human language is not always appropriate for technological solution.
- Clients demand for the requirement that is not required by organization.
- Client cannot properly explain the business needs.
- The client representatives are not helpful to elicit requirements due to conflict of interest.
- Lack of professional behavior in consultant or analyst
- RE is not deterministic.

8. Requirements Engineering in Open Source Software

OSSD does not follow the Software Engineering approaches elaborated in modern research as mentioned above. However, it does not mean that in OSS SE is done unsuccessfully. It is an entirely unique approach in which software development is quite transparent and development artifacts are freely available over the internet (Scacchi, 2007). RE processes in OSS are different from those followed in traditional software development (Crowston, 2007).

Various literature studies shown that OSSD projects does not follow the traditional RE process (Noll and Liu 2010; Scacchi 2002, 2009).

9. Techniques used for Requirement elicitation in OSSD

Renzel, Klamma and Jarke (2015) identified issue trackers as common web based tools for elicitation and negotiation of requirements. They described that these tools cause low participation of end users and proposed a requirement bazaar, a web based open source social media tool as a solution providing communication between developers and end users.

Laurent & Huang (2009) described that websites with online forums and project wikis are used for requirement gathering. They evaluated effectiveness of these tools for gathering requirements and managing feature requests. They proposed use of data mining tools for automatic topic discovery and use of recommender systems to keep users up-to-date about relevant discussions and avoid redundant discussion threads.

Kuriakose & Parson(2015) pointed out that literature about OSS describe that requirement elicitation is informal and ad hoc through web based approaches like issue databases, discussion forums . They perform an empirical study of requirement elicitation by involving the OSS developers and discussed various challenges regarding OSSD. They also proposed an enhancement to requirement gathering interface, the repository of reusable requirement patterns. Vlas & Robinson (2011) illustrated that requirement in OSSD is gathered informally through feature requests and discussion forums. They design an automated natural language requirement classifier.

Llanos and Castillo (2012) elaborated that requirements are asserted via web based artifacts. Scacchi(2002) conduct a research on four different

open source software development communities to find out their techniques of requirement elicitation. Initially he point out eight kinds of software informalisms for RE process in OSSD.

Alspaugh and Scacchi (2013) conducted a research to find out the mystery behind the success of OSS without using the classical requirement engineering processes. The focus was to find out how extensively OSSD use classical requirement? And what are the alternate methods instead of classical requirements. They described that requirement gatherings are mostly through feature requests or software bugs reported to Bugzilla and further discussed in e-bulletin boards and email lists.

Elkins and Dupée quoted informal requirements elicitation, developer led requirements in OSSD. Massey (2002) argue that source of requirements in OSSD is different from that of traditional requirements. Open source requirements come from developers, users, from emulation of implicit standards, by implementing explicit standards, by developing learning prototypes.

Scacchi (2009) pointed out about two dozen different types of informal software being used in OSSD. Almost 5-10 informalisms with different combinations are used to elicit requirements in different projects.

The informalisms used to elicit requirements were instant messaging; blogs; e-bulletin boards, and online forums;

news postings; project email; project digest;to-do lists ;hyperlinked web information ; how-to- guides; FAQs ; project Wikis; external publications; traditional system documentation; project licenses; open software architecture diagrams; reuse of software modules; plug-ins; project portals; source code directories on Web ;particular repositories of projects; bug reports; and issue databases such as Bugzilla. Provisionments may be found in many of these informalisms. Provisionment is a way of expressing qualities and features in the existing version of software, a competitor product, or developer's produced prototypes that can go under various changes.

Gill,Raza ,Zaidi and Kiani proposed a semi-automated ambiguity resolution in requirement elicitation from natural language for OSSD.

Noll (2011) conduct an empirical study on Firefox 2.0 and elaborated requirements are asserted by developers according to their experience and knowledge, from competitor product etc.

10. Conclusions

From the literature review, it is concluded that OSSD is opening the new ways of constructing, deploying and evolving complex software systems. However, informal ways of eliciting requirements are not yet part of traditional requirement engineering. OSSD requirement elicitation does not adhere to traditional requirement elicitation techniques.

Some of the researchers find out some challenges faced during requirement elicitation in OSSD as discussed in tabular form (table.1) and some of them proposed some solutions against these challenges. As this is a review of requirement elicitations in OSSD, the future work will be an empirical study of requirement elicitation in open source software development and an SLR.

Table 1: Requirement Elicitation in OSSD and its challenges

Paper	Current Requirement elicitation methods in OSS	Challenges discussed	Proposed elicitation methods
[37]	Common web based tools like issue trackers for elicitation and negotiation of requirement	Elicitation tools have excessive formality, technical jargon, and unmatched user experience causing low participation of end users.	Requirement bazaar
[15]	Websites with online forums are used for requirement gathering i.e., feature requests and bugs reported	Ineffective communication ways, difficulties in identifying user priorities, feature request management in forums, synchronous communication between users & developers.	Use of data mining methods to enhance the online requirement gathering and recommender systems.
[38]	Users and stakeholders use plain text boxes in requirement gathering interfaces in OSSD (e.g. Sourceforge) to submit requirement information.	Incomplete and invalid information, cognitive limitations of human participants.	Addition of Repository of reusable requirement pattern in interfaces used for requirement gathering in OSSD
[48]	Forums and feature requests in natural language.	Manual analysis of requirement in Natural language is time consuming and error prone.	Automated requirement classifier to deal with NL(natural language) text.
[41]	Require-ments are asserted by the developers	Informal elicitation e.g., README files, Barriers in RE: Differences in languages, time zone, culture, education. synchronous communication	Not proposed
[39]	Web based approaches used for requirement elicitation, analysis and prioritization	Not mentioned	Not proposed
[47]	Online forums, chats, project wikis, web based hyperlink information	Not mentioned	Ethnographic hypermedia to carry on requirement elicitation activities of OSSD
[46]	Informal text descriptions in forums and feature requests.	Manual analysis of requirement in Natural language is time consuming and error prone	Requirement classifier for NL(Natural language)
[45]	Requirement asserted by developers on the basis of their knowledge and experience, feature requests by end users	Not mentioned	Not mentioned
[5]	Describe Informalism to elicit requirement: Community communications, websites used for issue tracking and bugs reporting	Not mentioned	Various communication and collaboration tools, and use of web based portals and repositories for social interactions
[40]	Informal requirement elicitation, developer led requirement	Not mentioned	Not proposed
[44]	Informal ways of communicat-ion to interact in OSSD	Machines can not completely understand the domain terminologies and communication gestures so fully automated elicitation tools are not reliable.	Semi- automated ambiguity resolution in requirement elicitation from natural language.
[43]	Provision-ment, bug reports,feature request	Not mentioned	Not proposed
[42]	Asserted by developers, by competing products, users, extension implement-tations of products.	Not mentioned	Not proposed

References

[1] Feller, J. and Fitzgerald, B. (2000) A Framework Analysis of the Open Source Software Development Paradigm, in W. Orlikowski, P. Weill, S. Ang & H. Krcmar (Eds) Proc. of 21st Annual International Conference on Information Systems, (ICIS2000), Brisbane, Australia, Dec 2000. (Winner of the ICIS Best Conference Theme paper award)

[2] Bretthauer, D. (2002). Open source software: A history. Information Technology and Libraries, 21(1), 3.

[3] Scacchi, Walt. "Free/open source software development: Recent research results and methods." Advances in Computers 69 (2007): 243-295.

[4] Scacchi, W., Feller, J., Fitzgerald, B., Hissam, S., & Lakhani, K. (2006). Understanding free/open source software development processes. Software Process: Improvement and Practice, 11(2), 95-105.

[5] Scacchi, W. (2009). Understanding requirements for open source software. Design Requirements Engineering: A Ten-Year Perspective, (14), 467-494.

[6] Gacek, C., & Arief, B. (2004). The many meanings of open source. IEEE software, 21(1), 34-40.

[7] Divitini, M., Jaccheri, L., Monteiro, E., & Trætteberg, H. (2003, May). Open source processes: no place for politics. In Proceedings of ICSE 2003 workshop on Open source (pp. 39-44).

[8] Perens, B. (1999). The open source definition. Open sources: voices from the open source revolution, 1, 171-188.

[9] Vlas, R. "A requirements based exploration of open source software development projects: Towards a natural language processing software analysis framework". PhD dissertation. Georgia State University. 2012

[10] Bhowmik, T., Niu, N., Singhania, P., & Wang, W. (2015). On the role of structural holes in requirements identification: an exploratory study on open-source software development. ACM Transactions on Management Information Systems (TMIS), 6(3), 10

[11] Jensen, C. J. (2010). Discovering and Modeling Open Source Software Processes. University of California, Irvine.

[12] Weber, S. (2004). The success of open source. Harvard University Press.

[13] Johnson JP. Economics of open source software. Working paper, Cornell University; 2001.

[14] Lakhani KR, Wolf RG. Why hackers do what they do: understanding motivation and effort in free/open source software project. Working paper, MIT Sloan School of Management/The Boston Consulting Group; 2003.

[15] Laurent, P., & Cleland-Huang, J. (2009, June). Lessons learned from open source projects for facilitating online requirements processes. In International Working Conference on Requirements Engineering: Foundation for Software Quality (pp. 240-255). Springer, Berlin, Heidelberg.

[16] Raymond, E. (1999). The cathedral and the bazaar. Philosophy & Technology, 12(3), 23.

[17] Bezroukov, N. (1999). A second look at the cathedral and the bazaar. First Monday, 4(12).

[18] Bezroukov, N. (1999). Open source software development as a special type of academic research: Critique of vulgar Raymondism. First Monday, 4(10).

[19] Lerner J, Tirole J. Some simple economics of open source. The Journal of Industrial Economics 2002;50(2):197–234.

[20] https://sourceforge.net/about accessed on 24.08.2017

[21] Gangadharan, G. R., & Butler, M. (2012, September). Free and Open Source Software Adoption in Emerging Markets: An Empirical Study in the Education Sector. In IFIP International Conference on Open Source Systems (pp. 244-249). Springer, Berlin, Heidelberg.

[22] Fitzgerald, B. (2006). The transformation of open source software. Mis Quarterly, 587-598.

[23] G. Kotonya and I. Sommerville, Requirements Engineering – Processes and Techniques, John Wiley & Sons, Chichester, UK, 1998.

[24] Lamsweerde, A. V. (2000). Requirements engineering in the year 00: A research perspective. Paper presented at the 22nd International Conference on Software Engineering, Limerick, Ireland

[25] Davey, B., & Parker, K. (2015). Requirements elicitation problems: A literature analysis. Issues in Informing Science and Information Technology, 12, 71-82.Retrieved from http://iisit.org/Vol12/IISITv12p071082Davey1929.pdf

[26] Moløkken-Østvold, K., & Jørgensen, M. (2003). A review of software surveys on software effort estimation.Paper presented at the 2003 International Symposium on Empirical Software Engineering, 2003. ISESE 2003

[27] Briggs, R. O., & Gruenbacher, P. (2002). EasyWinWin: Managing complexity in requirements negotiation with GSS. 35th Hawaii International Conference on System Sciences, Hawaii, IEEE.

[28] Eberlein, A., & Leite, J. (2002). Agile requirements definition: A view from requirements engineering. Paper presented at the International Workshop on Time-Constrained Requirements Engineering (TCRE'02), Essen, Germany

[29] Boehm, B. (2000). The art of expectations management. Computer, 33(1), 122-124.

[30] I. Sommerville and P. Sawyer, Requirements Engineering – A Good Practice Guide, John Wiley & Sons, Chichester, UK, 2000.

[31] B. Nuseibeh and S. Easterbrook, "Requirements Engineering: A Roadmap", Department of Computing, Imperial College, London, 2000.

[32] A. D. Lucia and A. Qusef "Requirements Engineering in Agile Software Development", journal of emerging technologies in web intelligence, vol. 2, no. 3, august 2010.

[33] F. Paetsch, A. Eberlein, F. Maurer, "Requirements Engineering and Agile Software Development", Twelfth IEEE International workshop, 2003.

[34] Zhang, Z., 2007. Effective Requirements Development- A Comparison of Requirement Elicitation Techniques: In Proceedings of Software Quality Management Conference : Software Quality in the Knowledge Society, British Computer Society pp: 225-240.

[35] Paetsch, Frauke, Armin Eberlein, and Frank Maurer. "Requirements engineering and agile software development." 2012 IEEE 21st International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises. IEEE Computer Society, 2003.

[36] Fox C. (2007). Introduction to Software Engineering Design, Processes, Principles, and Patterns with UML 2. Boston, Massachusetts: Pearson/Addison Wesley.
[37] Renzel, D., Klamma, R., & Jarke, M. (2015). Requirements Bazaar: Experiences, Added-Value and Acceptance of Requirements Negotiation between End-Users and Open Source Software Developers. In Software Engineering & Management(pp. 122-123).
[38] Kuriakose, J., & Parsons, J. (2015, August). An enhanced requirements gathering interface for open source software development environments. In Requirements Engineering Conference (RE), 2015 IEEE 23rd International (pp. 288-289). IEEE.
[39] Llanos, J. W. C., & Castillo, S. T. A. (2012, May). Differences between Traditional and Open Source Development Activities. In PROFES (pp. 131-144).
[40] Elkins, M., & Dupée, B. The state of the art of Open Source Requirements Elicitation.
[41] Kuriakose, J., & Parsons, J. (2015, August). How do open source software (OSS) developers practice and perceive requirements engineering? An empirical study. In Empirical Requirements Engineering (EmpiRE), 2015 IEEE Fifth International Workshop on (pp. 49-56). IEEE.
[42] Noll, J. (2008, September). Requirements acquisition in open source development: Firefox 2.0. In IFIP International Conference on Open Source Systems (pp. 69-79). Springer, Boston, MA.
[43] Alspaugh, T. A., & Scacchi, W. (2013, July). Ongoing software development without classical requirements. In Requirements Engineering Conference (RE), 2013 21st IEEE International (pp. 165-174). IEEE.
[44] Gill, K. D., Raza, A., Zaidi, A. M., & Kiani, M. M. (2014, May). Semi-automation for ambiguity resolution in Open Source Software requirements. In Electrical and Computer Engineering (CCECE), 2014 IEEE 27th Canadian Conference on (pp. 1-6). IEEE.
[45] Noll, J., & Liu, W. M. (2010, May). Requirements elicitation in open source software development: a case study. In Proceedings of the 3rd International Workshop on Emerging Trends in Free/Libre/Open Source Software Research and Development(pp. 35-40). ACM.
[46] Vlas, R. E., & Robinson, W. N. (2012). Two rule-based natural language strategies for requirements discovery and classification in open source software development projects. Journal of Management Information Systems, 28(4), 11-38.
[47] Scacchi, W., Jensen, C., Noll, J., & Elliott, M. (2009). Multi-Modal Modeling, Analysis, and Validation of Open Source Software Development Processes.
[48] Vlas, R., & Robinson, W. N. (2011, January). A rule-based natural language technique for requirements discovery and classification in open-source software development projects. In System Sciences (HICSS), 2011 44th Hawaii International Conference on (pp. 1-10). IEEE.