IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Volume-3, Issue-3 E-ISSN: 2347-2693

Applying GQM Approach towards Evaluation of Defect Management in Free/Open Source Software Projects

Dr. Anu Gupta

Department of Computer Science and Applications, Panjab University,Chandigarh, India
anugupta@pu.ac.in

www.ijcseonline.org

Received: Mar/02/2015 Revised: Mar /12/2015 Accepted: Mar /22/2015 Published: Mar /31/2015

***Abstract*—** Free/Open Source Software (F/OSS) has emerged as a novel model of software development and distribution during the last decade. An F/OSS project generally evolves by receiving submissions from its volunteers in form of source code, bug identification, feature request, support request, translation request, documentation etc. The present paper uses F/OSS defect data extracted from a research collaboratory. Then it applies Goal/Question/Metric approach to determine the effectiveness of Defect Reporting and efficiency of Defect Resolution. The research findings of present work provide empirical evidences about Defect Management which F/OSS Projects may use to improve software quality.

***Keywords*—**Free Software; Open Source; Defect Management; GQM;

I. INTRODUCTION

During the last decade, the expansion of Internet and related technologies has given impetus to Free/Open Source Software (F/OSS) model. It has transformed the way software projects are developed and distributed. Raymond has described distinction between the cathedral and the bazaar where cathedral is chosen to represent conventional software engineering approach with tightly coordinated, centralized teams following a rigorous development process [1]. In contrast, the bazaar metaphor reflects a development approach where projects are generally developed by geographically scattered volunteers, communicating using online tools and platforms. F/OSS Development involves a process where the entire source code is accessible for conducting peer review and defect identification [2]. This openness of source code has few key advantages for F/OSS users. Foremost advantage is the ability to test the system knowing exactly what goes on inside the software. Another advantage is the ability to fix defects without waiting for the community to catch up. Thus an F/OSS Project generally evolves at a fast pace by receiving submissions from various sources to address various aspects of the project like bug identification, feature request, support request, translation request, patch submission etc. Continuous and incremental product improvement through defect finding and fixing is a development hallmark of the F/OSS paradigm and is characterized by Eric Raymond as "Release early, Release often" [1].

Hence in most of the F/OSS Projects, substantial amount of defect related data gets accumulated in the Defect Management Systems over the period. This valuable historical defect data can be used to analyze the past experience as well as determine the responsiveness of F/OSS contributors towards users' feedback. Before adopting a particular F/OSS Product, the prospective users especially organizational ones need to evaluate the extensibility and maintainability. An analysis of historical defect data can greatly help them to evaluate how efficiently and effectively the requests for fixing bugs, enhancing features, translation requests, support requests etc. are being managed.

Moreover the availability of huge amount of information with diversity in size, development tools, methods etc. offers the possibility of detailed comparison among F/OSS Projects from which knowledge and experience can be gained by researchers as well as practitioners. In the current study, popular Goal/Question/Metric (GQM) approach is applied to analyze the valuable defect data of F/OSS Projects from various perspectives, specifically focusing on evaluating the efficiency and effectiveness in resolving defects and determining responsiveness towards users.

The rest of the paper is organized as follows; Section II reviews the work done in related areas. Section III presents research methodology where project selection, data collection and GQM approach are briefly described. It further highlights the quantitative results. Section IV discusses the results obtained. Finally, Section V concludes and provides directions for future work.

II. RELATED WORK

F/OSS project workspaces such as SourceForge provide access to project related information and offer specialized tools for mining this huge data [3]. Several studies have also been conducted which make use of historical data of F/OSS

Corresponding Author: *Anu Gupta, anugupta@pu.ac.in*
Department, of Computer Science and Applications, Panjab University,Chandigarh, India

projects. A study of Apache web server and Mozilla web browser quantified various aspects of developer participation, core team size etc. by using e-mail archives of source code change history and defect reports [4]. Another study analyzed the temporal changes among several F/OSS projects and discussed the distribution of defects among various categories on the basis of statistics provided by SourceForge [5].

A study analyzed the debugging process of nine popular F/OSS Projects and highlighted four types of bug fixing processes that can be distinguished by considering temporal continuity and efficiency dimensions [6]. A number of statistical analyses have been recorded about the F/OSS Project Debian to analyze the effectiveness of the F/OSS development process used by projects such as Debian [7].
Another study has investigated the coordination practices adopted within four F/OSS development teams focusing particularly on bug fixing process and confirmed the validity of Raymond's Bazaar metaphor for bug fixing process [8]. Various methods have also been developed that allow automated analyses to evaluate and interpret CVS and change log data [9] [10].

Some quantitative analyses has been carried out to examine the code quality about GNOME and various other projects [11] [12]. The Open Source Maturity Model (OSMM) helps in evaluating the maturity level of F/OSS Projects based upon support, documentation, training, product integration etc. [13] [14]. Atos Origin has also developed the method for Qualification and Selection of Open Source Software (QSOS) [15].

The Open Business Readiness Rating (OpenBRR), initiated by Intel Corporation, the Centre for Open Source Investigation at Carnegie Mellon University and SpikeSource also supports evaluation of F/OSS in a standardized way [16]. The Open Source Software Quality Observation (SQO-OSS) is also providing for the evaluation and quantification of F/OSS project quality [17]. This system is based on the automated analysis of the available data sources of the project (such as CVS, mailing lists and Defect management databases) to find out a quality metric for F/OSS Projects.
Even though there are number of studies about F/OSS, little work has been done to utilize the data stored in Defect Management System of F/OSS Projects. It is also found that only some very successful projects have caught the attention of researchers generally. There is a considerable requirement of efforts to that can facilitate prospective users to choose the most appropriate F/OSS Product.

III. RESEARCH METHODOLOGY

A. Project Selection and Data Collection

In the current study, F/OSS Projects are selected from SourceForge, a centralized place for F/OSS developers to host their projects [3]. It is the world's largest F/OSS Projects repository with more than 430,000 F/OSS projects and over 3.7 million registered users. A single source is chosen to select projects in order to control for differences in available tools and project visibility. In spite of large number of projects hosted, only a small proportion of these projects are actually active. Also many of the F/OSS Projects either do not use or do not allow public access to Defect Management System. Hence those projects are considered for which defect related data is publicly accessible and is being maintained completely at SourceForge. Another criterion used for selection of projects is the project development stage (1-6 where 1 is the planning and 6 is a mature stage). A cut-off of 5 is chosen which indicates that the selected projects are at similar stage of development and are not in the early stage of development lifecycle. A total of 20 projects are selected. Selection of limited number of projects has helped to carry out in-depth study. For all the selected F/OSS projects, detailed defect data is downloaded from SourceForge Research Data Archive (SRDA) [18]. The defect data is downloaded on the basis of unique Project ID assigned to each project at SourceForge and is stored in the local repository (mySQL) comprising more than 60,000 defect records. A conceptual framewoek for Defect Data extraction and analysis has been proposed [19].

B. Goal/Question/Metric (GQM) Approach

Goal/Question/Metric (GQM) method was originally developed by V. Basili and D. Weiss and further expanded by D. Rombach [20]. The GQM is set up in three steps. First, the reasons to measure quality are identified, i.e., the Goals are determined at conceptual level. The second step enumerates Questions at operational level that will help to assess whether a quality goal has been achieved or not. Finally, the third step tends to transform the Questions into Metrics in a quantitative way. The current study applies GQM approach towards evaluation of Defect Management of F/OSS projects in the following manner (Table 1):

Goal	Question	Metric
G1. To determine effectiveness of Defect Reporting	**Q1**. What is the rate of defect arrival? **Q2**. Does it show some trend over the period?	**M1**. Defect Arrival Pattern (Monthly) **M2.** Linear Trend Lines
G1. To determine the	**Q3.** What is the defect resolution	**M3.** Cumulative Defect Arrival Pattern and Defect Closure pattern

efficiency of Defect Resolution	pattern over the period? **Q4.** How is the backlog management capability of the F/OSS community? **Q5.** Do new software releases have some relationship with the backlog management of F/OSS Projects? **Q6.** How much time is taken to resolve the defects? **Q7.** How long does a defect remain pending?	over time interval (Monthly) **M4.** Backlog Management Index (BMI) values in Control Charts **M5.** Plotting of Vertical Lines for Software Releases (Major/Minor) in BMI Control Charts **M6.** Average Age of Defects Resolved *DRA(d_i)=Defect Closing Date(d_i)-Defect Opening Date(d_i)* **M7.** Average age of the pending defects *DPA(d_i)=Current Date-Defect Opening Date(d_i)*

TABLE 1
Goal/Question/Metric for Evaluation of Defect Management

C. Analysis and Results

The metrics mentioned in Table 1 are applied to the defect reports gathered for various projects. The detailed analysis and results obtained are being presented as follows:

- *Defect Arrival*

Defect Arrivals refers to all the defects that have been identified and reported by F/OSS users at the hosting site. Such defects help to determine the efforts of geographically distributed community, their interest and feedback towards the project. Live Defect arrival data for each of sampled Project is consolidated on monthly basis and is plotted in form of line graphs to observe the defect arrival pattern. Fig. 1 shows one such graph for some of the F/OSS Projects. It is found that most of projects have very few defects being reported initially. This is related to the fact that the projects have less number of downloads as well as less users in the beginning. With the passage of time, number of downloads as well as active users increase. This generally tends to increase Defect arrivals. It has also been observed that defect arrivals start decreasing after some time. It is due to the fact that projects start moving towards stable status. It is also seen that the defect arrival pattern is quite inconsistent and fluctuating in all the projects. The linear trend lines are also plotted corresponding to inconsistent and fluctuating defects arrival. To see whether aggregate defect reports are changing significantly over the period or not, a standard analysis of variance (ANOVA) is carried out on aggregate defect reports submitted during various months over all the years for all the projects taken together. The results also confirm that defect reports change significantly over the years.

- *Defect Resolution*

Defect Resolution refers to the process of fixing/closing of reported defects. Such resolution may cause changes in the source code of the project. Certain defects may be closed without any corresponding code change also. Cumulative Defects arrived over the period are represented in Defect arrival curve. Defect closure curve is related to the resolution and closing of defects by F/OSS community, represented by Cumulative Defects closed over the period. The gaps between these two curves indicate the pending defects at that point of time. An ideal defect resolution process needs to be Continuous (when cumulative defect closed curve is quite smooth without having any peaks or steps) and Efficient (when cumulative closed curve stays near to the cumulative open curve). Various F/OSS projects show variations in the Defect resolution process. In Fig. 2, defect closure curve is discontinuous and inefficient initially. But later it denotes a project with high quality of defect resolution process. Fig. 3 highlights a project with lower quality of defect resolution process where gaps i.e. pending defects go on increasing.

- *Backlog Management*

Backlog management refers to the capability of F/OSS volunteer developers to handle the pending defects [20]. It is measured using Backlog Management Index (BMI) which is calculated as ratio of number of defects closed to number of defects arrived during the period.

$$BMI = \frac{Number\ of\ defects\ closed\ during\ the\ period}{Number\ of\ defects\ arrived\ during\ the\ period} \times 100$$

The backlog is reduced if defects are being closed at the same or higher rate at which the defects are arriving (If BMI is larger than 100). If BMI is less than 100, the backlog gets increased. The technique of control charting can help to calculate the overall backlog management capability of the software process [21]. In fact BMI chart is a pseudo-control chart because BMI data are auto correlated and assumption of independence for control charts is violated. As the BMI values are in wide range, *c* control chart is more suitable [21]. In this case, three kinds of control lines are calculated as follows:

- Central Line (CL) equal to Mean BMI

- Lower Control Limit $LCL = (CL - 3\times\sqrt{\text{CL}})$
- Upper Control Limit $UCL = (CL + 3\times\sqrt{\text{CL}})$

If a process is mature and under statistical process control, all values should lie within the LCL and UCL. The process is said to be out of statistical process control if any value falls out of the control limits. Fig. 4 shows poor backlog management throughout the period for one of the sampled projects. It is observed that BMI curves for most of the F/OSS projects are very fluctuating. To find out the reasons for such behavior, a detailed analysis of release data with BMI curves was also carried out. It is explored that the F/OSS Projects are releasing their minor/major versions very frequently confirming the premise "*Release Early, Release Often*" [1]. In the Fig. 4, efforts are also made to trace back the shapes of BMI curves with release history of the projects. It is found that more than 90% of spikes in BMI curves are matching with the version releases. This phenomenon refers that generally F/OSS developer community do not resolve the defects regularly, instead put additional efforts to resolve defects just before each release.

- *Defect Resolution Age*

Defect Resolution Age (DRA) refers to the number of days elapsed between a defect arrival date and defect resolution date. The average defect resolution age should be short as well as quite consistent for efficient defect resolution.

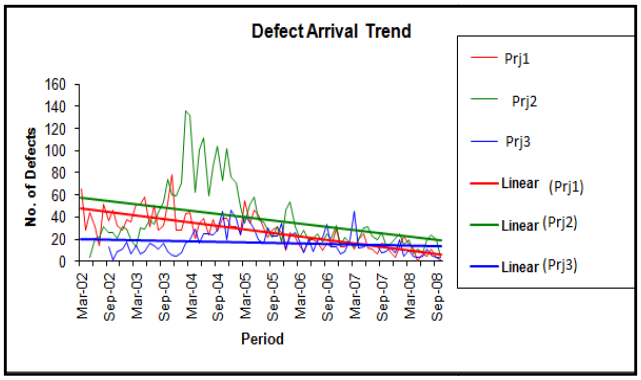

Fig 1: Defect Arrival Trend

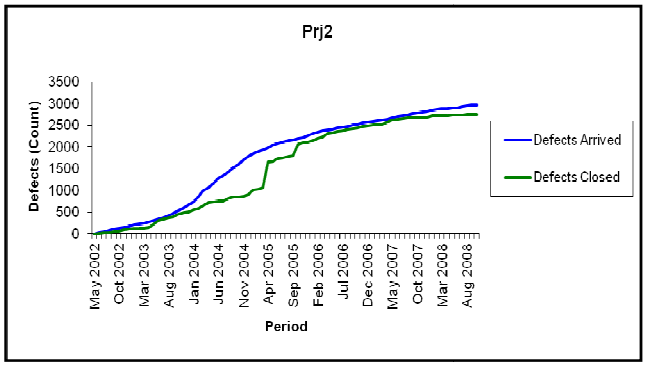

Fig 2: Defect Resolution

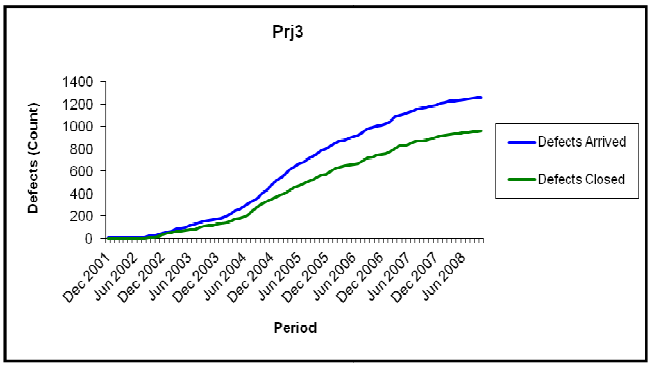

Fig 3: Defect Resolution

The monthly average of defect resolution age (MADRA) is computed using the following formula:

$$MADRA = \frac{\sum_{i=1}^{N} DRA\,(di)}{N}$$

Where d_i refers to a defect closed

For the F/OSS Projects under study, none of the projects has shown decreasing trend, very few projects are having near to constant trend lines and most of the projects are showing upward trends in average defect resolution age over the period (Fig. 5). To analyze the overall defect resolution age for all the selected projects together during the investigation period, average resolution age for each of the 20 projects for various years is taken into consideration and standard analysis of variance (ANOVA) is applied which also shows that there is significant change in defect resolution age over the period.

- *Defect Pending Age*

Defect Pending Age (DPA) refers to the number of days elapsed since a defect arrived and still remained pending at the end of the month. For all the selected F/OSS Projects, monthly average of defect pending age (MADPA) is computed using the following formula:

$$MADPA = \frac{\sum_{i=1}^{N} DPA\,(di)}{N}$$

Where d_i refers to a pending defect

It is found that most of the projects are showing increasing trend of monthly average defect pending age. Further detailed analysis of defects pending age is carried out by classifying the pending defects according to their pending age (Less than 10 days, 11 to 30 days, 31 to 90 days, 91 to 365 days and More than 365 days). Fig. 6 shows curves for the overall monthly average pending age of all the pending defects as well as monthly average pending age for defects

falling in each of the categories. By observing the pattern of defect pending age over the period, it is found that the average pending age is increasing in almost all the projects. But this increase in defect pending age trend is attributed mainly by those defects whose average pending age is 90 days or more. While in lower age categories, trend remains either constant or slightly fluctuating.

.

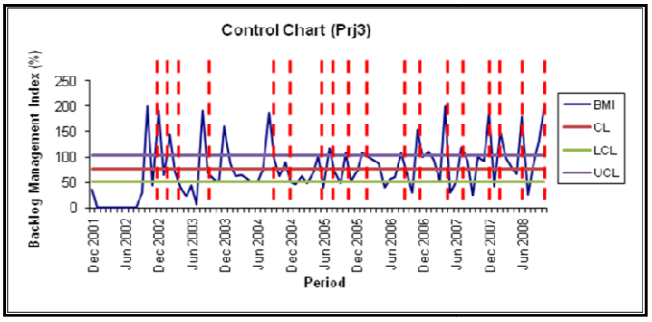

Fig 4: Software Release and Backlog Management of Defects

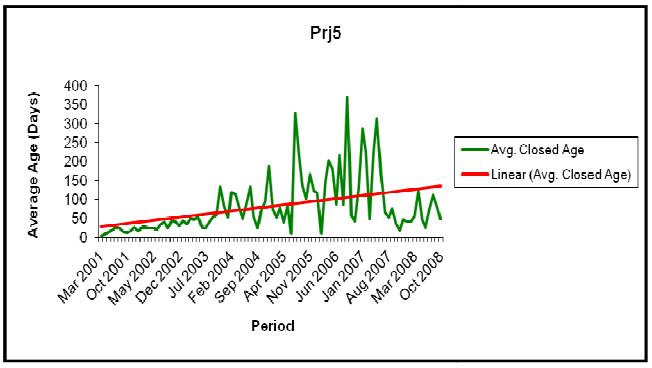

Fig 5: Defect Resolution Age (Increasing Trend)

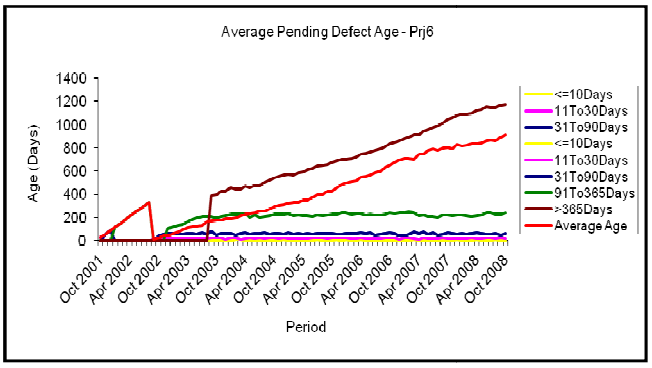

Fig 6: Defect Pending Age

To analyze the overall defect pending age for all the selected projects together during the investigation period, average pending age for each of the 20 projects for various years is taken into consideration and standard analysis of variance (ANOVA) is applied which confirms that there is significant change in defect pending age over the period.

IV. DISCUSSION

F/OSS Projects tend to gather large amount of defect related data because of interaction among volunteers facilitated by Hosting site. The present paper applies GQM approach to analyze determine how various variables associated with defects change over time such as defect arrival defect resolution, pending defects etc. and brings forth many important insights.

Generally an F/OSS Project is developed by a core team comprising few developers that is further surrounded by a globally distributed community of active as well as passive users. The active users communicate their feedback in form of bugs, feature requests, patch submissions etc. through Defect Management System provided by project hosting site. Because of frequent releases in F/OSS projects, defect arrival pattern is normally inconsistent. But an overall downward trend indicates gradual progress in stability and quality of the F/OSS Projects.

During the analyses, it has been found that generally defect resolution is not performed very consistently. This tends to decline defect removal rate and increase the average resolution age of defect. This problem needs to be addressed timely otherwise important user feedback can not used to enhance the software project. It is also observed that defects get accumulated gradually and then additional efforts are put to resolve them near the forthcoming software releases. It is also found that a few defects remain pending for fairly long period of time in the Defect Management System. They are neither resolved nor their status is updated, if resolved. Such ignored defects keep on accumulating and result in increasing trend in overall defect pending age.

The inefficient defect resolution may have serious implications on the growth of F/OSS projects in the long term. Reproduction of defect also becomes difficult with its increasing pending age. Finally, users will also be discouraged to provide further feedback if due consideration is not given to reported defects. This reduces the benefits that a Project can obtain from peer review and volunteer contribution, the hallmark of F/OSS.

V. CONCLUSION AND FUTURE SCOPE

F/OSS repositories contain huge amount of valuable data that can be used to determine the progress of projects as well as to facilitate the users in evaluating the products. The

research findings of present work contribute to an understanding of Defect Management practices from practitioner perspective as well as provide empirical evidences about effectiveness and efficiency in defect resolution, which F/OSS Projects may use to improve software quality. F/OSS is an evolving paradigm of software development. A variety of Defect Management Systems are being used among F/OSS Projects. The current study is focused on one of them being used at SourceForge; future work would be carried out on other Defect Management Systems to analyze more F/OSS Projects. Moreover Defect Data can also be combined with data from other publicly accessible repositories for further research.

ACKNOWLEDGMENTS

We are thankful to the University of Notre Dame for providing access to Sourceforge Research Data Archive (SRDA) for retrieving data on F/OSS projects.

REFERENCES

[1] Eric S. Raymond, "The Cathedral and the Bazaar", *First Monday*, Volume -**3**, No. **3**, **1998**. *http://firstmonday.org/htbin/cgiwrap/bin/ojs/index.php/fm/article/view/578/499*

[2] Joseph Feller, Brian Fitzgerald, Scott A. Hissam and Karim R. Lakhani, "Perspectives on Free and Open Source Software", **2005**, The MIT Press. *http://mitpress.mit.edu/books/chapters/0262562278.pdf*

[3] "SourceForge", *http://sourceforge.net/*

[4] Audris Mockus, Roy Fielding and James D. Herbsleb, "Two Case Studies of Open Source Software Development: Apache and Mozilla" *ACM Transactions on Software Engineering and Methodology*, Volume- **11**, No.-**3**, Page No. - (309-324),**2002.**

[5] Dawid Weiss, "A Large Crawl and Quantitative Analysis Of Open Source Projects Hosted On Sourceforge", *Research Report ra-001/05(2005)*, Institute of Computing Science, Pozna University of Technology, Poland. *http://www.cs.put.poznan.pl/dweiss/xml/publications/index.xml*

[6] Chiara Francalanci and Francesco Merlo, "Empirical Analysis of the Bug Fixing Process in Open Source Projects", *Open Source Development, Communities and Quality*, Springer Boston, Volume- **275**, Page No.- (**187-196**), **2008**.

[7] Martin Michlmayr and Anthony Senyard, "A Statistical Analysis of Defects in Debian and Strategies for Improving Quality in Free Software Projects", *The Economics of Open Source Software Development*, Elsevier B.V., Page No.- (**131–148), 2006**.

[8] Kevin Crowston and Barbara Scozzi, "Bug Fixing Practices within Free/Libre Open Source Software Development Teams", *Journal of Database Management*, Volume- **19**, No.-**2**,Page No. -(**1-30**), **2008**.

[9] David A. Wheeler, "Estimating Linux's Size Version 1.04", May **2001**. *http://www.dwheeler.com/sloc/*

[10] Daniel German and Audris Mockus, " Automating the Measurement of Open Source Projects", *Proceedings of the 3rd Workshop on Open Source Software Engineering*, *International Conference on Software Engineering*, May **2003**, Portland, Oregon, USA.

[11] Stefan Koch, "Effort Modeling and Programmer Participation in Open Source Software Projects ", *Information Economics and Policy,* Volume- **20**, No. **4**, Page No. – (**345-355), 2008**.

[12] Ionic Stamelos, Lefteris Angelis, Apostolos Oikonomou and Georgios L. Bleris, "Code Quality Analysis in Open Source Software Development", *Information Systems Journal*, Volume - **12**, No. **1**,Page No. (**43-60**), **2002**.

[13] Navica's Open Source Maturity Model (OSMM)", *http://www.navicasoft.com/pages/osmm.htm*

[14] Frans-Willem Duijnhouwer and Chris Widdows, "Capgemini Expert Letter Open Source Maturity Model", Capgemini, **2003**. *http://pascal.case.unibz.it/retrieve/1097/GB_Expert_Letter_Open_Source_Maturity_Model_1.5.31.pdf*

[15] "Qualification and Selection of Open Source Software (QSOS)", *http://www.qsos.org/methode.php*

[16] "Open Business Readiness Rating", *http://www.openbrr.org/wiki/index.php/Home*

[17] "Software Quality Observatory for Open Source Software (SQO-OSS)", *http://www.sqo-oss.eu/*

[18] G. Madey, The SourceForge Research Data Archive (SRDA), University of Notre Dame, *http://zerlot.cse.nd.edu/*

[19] A. Gupta, R.K. Singla," Qualitative Evaluation of Defect Resolution in Free/Open Source Software Projects", International Journal HIT Transactions on ECCN, ISSN: 0973-6875 Volume - **3**, No. **9**, Page No.- (**27-36**), **2009**.

[20] V. Basili, G. Caldiera, and H. D. Rombach. The Goal Question Metric Approach, John Wiley & Sons Inc., **1994**.

[21] Stephen H. Kan, "Metrics and Models in Software Quality Engineering", Second Edition, Pearson Education, **2003**.

AUTHORS PROFILE

Dr. Anu Gupta has been working as Associate Professor in Computer Science and Applications at Panjab University, Chandigarh since July 1998. She has also held the position of Chairperson, Department of Computer Science & Applications, Panjab University, Chandigarh (Feb. 2008- Jan. 2011) .She was awarded University medal for securing first position in M.C.A. at Punjabi University, Patiala, Punjab in the year 1997. She has the experience of working on several platforms using a variety of development tools and application packages. She has completed the Doctor of Philosophy Degree from Panjab University in the area of Free/Open Source Software. Her research interests include Networking, Multimedia Technologies, E-Commerce and Software Engineering. She is a life-member of 'Computer Society of India' and 'Indian Academy of Science'. She has published several research papers in various journals and conferences.

