The Impact of Process Effectiveness on User Interest in Contributing to the Open Source Software Projects

Amir Hossein Ghapanchi
Griffith University/School of Information and Communication Technology, Queensland, Australia
Email: amir.ghapanchi@gmail.com

Aybuke Aurum
The University of New South Wales/Australian School of Business, Sydney, Australia
Email: Aybuke@unsw.edu.au

Farhad Daneshgar
The University of New South Wales/Australian School of Business, Sydney, Australia
Visiting Fellow, Institute for Knowledge and Innovation/Bangkok University, Thailand
Email: f.daneshgar@unsw.edu.au

Abstract—**Unlike closed source software development, open source software (OSS) projects are not commonly driven by direct profit and do not offer developers monetary incentives. Instead, OSS development relies on volunteer developers and voluntary contributions from the user community. Thus, attracting voluntary user contributions to OSS projects is a challenging task. Defect fixing is one important area of OSS development that requires user contributions. Postulating upon the theory of competency rallying, this research examines the impact of the defect-fixing effectiveness on user interest in contributing to OSS projects. Analysis of data collected from 1481 OSS projects confirms that the effectiveness of the defect fixing process till any period of time has a positive significant effect on the user contribution in terms of defect submission as well as defect resolution in the following time period. The results of this study have several implications for OSS projects' managers as well as corporations interested in adopting OSS software.**

Index Terms—**Open source software success, competency rallying, defect-fixing process, user contribution.**

I. Introduction

In spite of increasing adoption of open source software, many OSS projects still fail in their early stages of development [1] [2]. According to Krishnamurthy [3], 63% of OSS projects on Sourceforge.net, the world's largest OSS host, experience failure. The reason for this failure is that a large majority of OSS projects cannot attract voluntary contributions from user community to further their development activities [4].

Having effective and healthy development processes (e.g. defect fixing process) has also been reported as a critical antecedent to the project success [5]. Defect-fixing is one of the key processes that characterize OSS development [6]. Our study focuses on defect-fixing process for three main reasons. First, an effective defect-fixing process is tied to users' perception of the project quality, activity and value [7] that in turn will affect project success [8]. Second, prior research on OSS projects has implied the importance of effective defect-fixing in impacting OSS projects positive outcomes, but has not empirically tested this relationship [6] [9] [10]. Third, defect-fixing process is within the control of OSS project team so understanding its impact on project success might be significant for OSS practitioners. If an effective defect resolution process leads to higher user interest in contributing to the project, then we would need to ask a practically and academically significant question which forms the main research question underlying this study: "Does the effectiveness of the defect-fixing process impact user interest in contributing to OSS projects?"

The consequence of poor responsiveness to customer needs in terms of defect-fixing ranges from user dissatisfaction, to unsafe software [11]. High responsiveness to defects has the potential to result not only in higher quality software, but also higher user contribution to the project. Although procedures have been proposed to increase user participation in software quality management, the relationship between responsiveness to user needs and user contributions has not been investigated [11]. Thus the objective of this study is to examine the relationship between defect-fixing effectiveness and user interest in contributing to the project.

Answering our research question is of critical importance because unlike commercial proprietary software development, OSS projects are not always driven by direct profits and do not offer developers monetary incentives [12], instead relying on input from

doi:10.4304/jsw.7.1.212-219

volunteer developers/users to further their development [1]. Moreover, user community can be viewed as a source of innovative ideas for further development of the current software [12] [13]. Therefore attracting contributions from user community is a critical issue for every OSS project.

The remainder of this paper is structured as follows. The next section provides background for the research and presents the research model. The research design is presented in Section 3. Data analysis and results are presented in Section 4. Discussion of the results is presented in Section 5. Section 6 shows the limitations of the study followed by concluding remarks in Section 7.

II. Background and Research Model

This section consists of four sub-sections. Firstly, literature on user interest in OSS projects is introduced. Subsequently, the defect-fixing process of OSS projects is reviewed. Finally the theory of competency rallying is introduced, resulting in a research model.

A. User Interest

User interest has been a key topic in information system (IS) research [14]. 'User interest' is the ability of an OSS project to attract the interest of community users to the project software [15] [16]. The extent to which an OSS project is able to attract community interest in using and contributing to the project is a key success factor for OSS projects [16].

Prior research on antecedents of user interest has resulted in interesting findings. Stewart et al. [15] showed that license restrictiveness is negatively associated with user interest, while having a sponsor has a positive impact. Subramaniam et al. [16] showed that developer interest, user interest and project activity are correlated. Additionally, they concluded that project activity, developer interest, project license and development status impact user interest. Moreover, Colazo [17] surveyed 121 open source projects and found that product popularity impacts on user contribution. Long [18] discovered that contributions from community impact the success or failure of OSS projects.

A review of prior studies has identified that there is a lack of literature exploring the software development process considerations that predict OSS project success [19]. Furthermore, the current literature lacks studies that examine the impact of the defect-fixing process on user contribution. As a result, the current study seeks to explore the influence of the effectiveness of the defect-fixing process on user interest in contributing to OSS projects.

'User contribution' is the contribution that the community makes to an OSS project, like reporting or fixing a defect. User contribution is the ideal state of 'user interest' because a given user doesn't contribute to an OSS project unless s/he has already adopted and used that particular project. Therefore, the current research suggests user contribution to an OSS project as a post-hoc usage behavior which shows user interest in that particular project.

B. Defect-Fixing Process

One of the most important areas of study in information system development is software quality. A significant dimension of software quality is responsiveness to user/customer needs [11]. According to Hsu et al. [11], responsiveness to customer needs is considered as an external quality dimension, identified by individual or organizational users.

The defect-fixing process is a process in which bugs and defects observed in the software are handled and resolved to improve the quality of the software. In OSS projects, this process is normally handled through a defect tracking system.

The defect-fixing process has been studied by few OSS researchers. Herbsleb and Mockus [20] found that the progress in fixing defects reported influences the positive outcomes in OSS projects. Stewart and Gosain [4] used the percentage of defect reports completed as an indicator of OSS project effectiveness and found that communication quality and team effort impact the quantity of defect reports completed. They suggest that OSS project success comprises the extent to which a project receives input from the community, and the extent to which it creates an observable output such as a defect fixed. Stewart and Gosain [21] also proved that the percentage of defect reports completed impacts perceived effectiveness of OSS projects defined by how well an OSS project succeeds to accomplish its goals.

The setting chosen for this research is the largest OSS repository, Sourceforge.net. Sourceforge.net doesn't clearly specify the process of defect fixing through its defect tracking system, defect pre-defines statuses for a defect. "Open" status is used when a defect is first reported. Subsequently, someone (e.g. a project administrator) either assigns it to a developer to be fixed, or rejects if it is a duplicate, out of date or not legitimate etc; the "Pending" status is also used when the defect is legitimate but it is better to be fixed at a point of time in future. Finally, when the defect report is completed, the status is changed to "closed". As the name proposes, the status "fixed" also used when a defect is resolved.

C. Theory of Competency Rallying

Theory of competency rallying (TCR) was first introduced by Katzy and Crowston [22]. Crowston and Scozzi [23] and Katzy and Crowston [24] then applied the TCR in two different contexts to demonstrate its rigor in explaining project success in the context of virtual organizations. The TCR introduces four sets of antecedents for project success including: (1) identification of market needs; (2) management of a short-term co-operative effort; (3) marshalling of competencies; and (4) identification and development of competencies. These four capabilities (See Figure 1) are all necessary for the success of a project undertaken in a virtual organization.

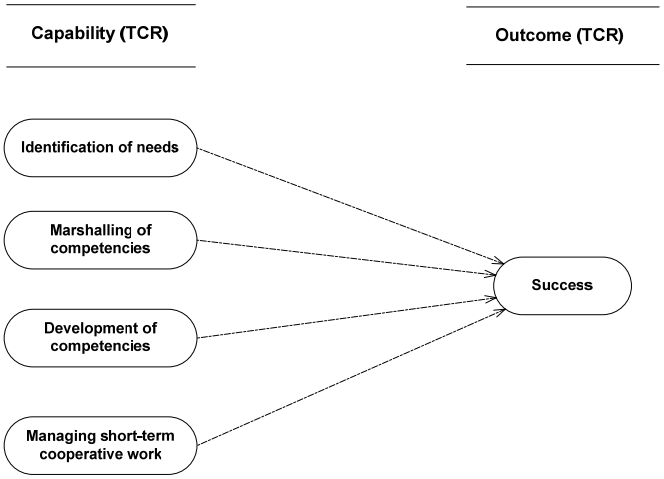

Figure 1. TCR's capabilities and outcome

In this research, we will employ the TCR to explain how competency rallying for the defect-fixing process might leads to project success in terms of procuring contributions from community users. Current paper applies the TCR at a process level meaning that capabilities in the TCR are related to the capabilities required for the defect-fixing process and the outcome of the TCR (i.e. success) is related to the project success in attaining community contributions to this process. In the following sections, we will explain the four sets of capabilities proposed by the TCR and demonstrate how each competency corresponds to various tasks of the defect-fixing process.

The first capability in the TCR refers to the ability of identifying a market need that could be fulfilled by the virtual organisation. There are four tasks involved in the defect-fixing process of an OSS project: reporting a defect, assigning the defect, fixing the defect, and managing and closing the defect. A defect reported by a user obviously shows some quality needs holding by the software community. Thus, we relate the first task involving in the defect-fixing process, reporting defects to the first competency in the TCR, identification of needs.

The second capability in the TCR refers to the ability of the virtual organization to recognize and bring together actors with the competencies required to meet an identified market need. If a reported defect is decided to be resolved, the defect is assigned to a developer to resolve. This needs comprehension of the assigner on who possess the expertise required to fix a defect. Therefore, similar to the first competency, we relate the second task involving defect-fixing process, assigning defects, to the second competency in the TCR, marshalling of competencies.

The third capability in the TCR shows the ability of the virtual organization for ongoing development and refinement of competencies. Competencies can be viewed as residing in the skills and expertise of individuals and groups within the organisation. By fixing more and more defects, a given developer can develop his/her competencies in the course of real-practice, "… [Project developers] autonomously [can] develop their competencies in the course of their on-going 'real world' activities and sharpen them in the course of an OSS project [p. 5]" [23]. Accordingly, we relate the third task involving in the defect-fixing process, fixing defects, to the third competency in the TCR, development of competencies.

The fourth capability in the TCR is the ability of a virtual organization to manage short-term co-operative work. Each defect reported on the defect tracking system of a particular OSS project can be attributed to short-term cooperative work because different people (e.g. submitter, assigner, fixer, and etc.) are involved in it. A defect lives from the moment the submitter reports it until the moment that it gets terminated (i.e. defect's status is set to "closed"). Accordingly, we relate the fourth task of the defect-fixing process, closing and managing defect, to the fourth competency in the TCR, managing short-term cooperative work.

D. Research Model

Given the justification presented in previous section, we developed a model of relationships (See Figure 2) to be tested through empirical data. The dependent variables of the model include user contribution to defect submission and defect resolution. There is one independent variable: the effectiveness of the defect-fixing process. There are also four variables in the conceptual model that, according to prior research, can impact our dependent variable including: project size, development status, project age, and project audience. However, as we were not interested in their impact, we controlled for them in the study.

As mentioned earlier, this research applies the TCR at a process level. In this context, capability means capabilities used in the course of the defect-fixing process. Inspired by the TCR, we argue that rallying competencies for the defect-fixing process might result in project success in terms of procuring user contributions to this process (See hypotheses 1 and 2). One key point here is that there is a lagging time between independent variable and dependent variables in this study. Subramaniam et al. [16] also mentioned this fact. Thus, we measure defect-fixing effectiveness till a point of time (from start of project till t1), and measure user contribution in the following time period (i.e. 8 months after t1).

H1. The effectiveness of the defect-fixing process till any period of time has a positive impact on user contribution to defect submission in the following time period.

H2. The effectiveness of the defect-fixing process till any period of time has a positive impact on user interest in contributing to defect resolution in the following period.

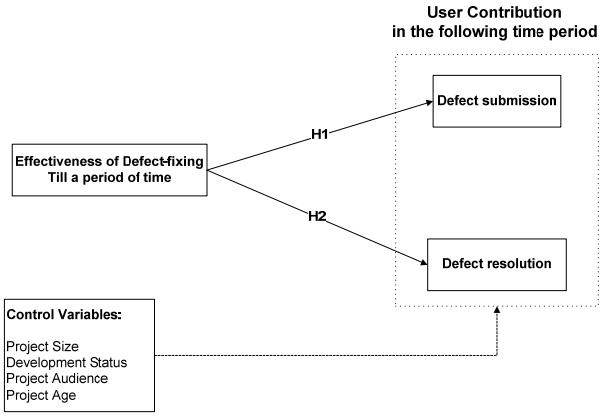

Figure 2. Research model

III. RESEARCH DESIGN

A. Sampling

Sourceforge.net divides OSS projects into various categories including: communication, database, desktop, education, formats and protocols, games and entertainments, Internet, multimedia, office/business, religion and philosophy, scientific/engineering, security, social science, software development, system, terminal, and text editor. In order to increase generalizability of the results, we decided to take sample from various categories. Thus, we chose to focus on five categories namely: communication, security, software development, scientific/engineering, and game and entertainment. In order to narrow down our sample, we then imposed a number restrictions such as:

- exclude projects that have not had any release within last 2 years (to discard inactive projects);
- exclude projects whose development status is planning, pre-alpha, or alpha (because they normally don't have any software release);
- exclude projects whose development status is mature (because they normally don't have much activity and are already mature in terms of defects and less activity is done on them for these purposes);
- focus on those projects that have had at least 5 records in their defect-tracking system.

The total number of projects in these 5 categories that satisfied our sampling criteria was 1481. We collected data on all of the 1481 projects. Table 1 shows the distribution of the projects based on the number of times each project had been downloaded.

TABLE I.
DISTRIBUTION OF THE PROJECTS IN TERMS OF NUMBER OF DOWNLOADS

Number of downloads	Frequency	Percentage
50-1000	78	5%
1000-20,000	601	41%
20,000-100,000	415	28%
>100,000	387	26%
Total	1481	100%

B. Operationalization of the Constructs and Data Collection

In order to operationalize the effectiveness of the defect-fixing process, we use the measures introduced by Ghapanchi and Aurum [25] namely: the number of defects submitted by team members; total number of defects assigned to team members; number of defects fixed by team members, and number of defects closed by team members. Applying a rigorous scale development methodology, Ghapanchi and Aurum [25] reported a high validity and reliability for the measures discussed. This operationalization is in line with the correspondence between the TCR capabilities and tasks underlying the defect-fixing process (See Section II, B).

We will operationalize our dependent variable by user contribution to the project in terms of submitting and resolving defects. User contribution to defect submission will be measured by the number of defects submitted by users. User contribution to defect resolution will also be measured by the number of defects closed by users.

There are four control variables in this study: project size, development status, project audience, and project age. Project size will be measured by the number of project team members. Project audience is a dummy variable that is set to 1 if the project is developer targeted and is set to zero otherwise. Project age is measured by the number of months a project has existed. With respect to development status, development status of the project on Sourceforge (e.g. beta, stable and etc) will be used. Data on control variables (project age, development status, project audience, and team size) were directly collected from the projects' profiles page on Sourceforge.

As mentioned before, we collected data in 2 snap shots. Data on the measures of the effectiveness of the defect-fixing process were collected at a point of time (t1), while data on user contribution to defect-fixing process were collected once at t1 and again at t2 which was 8 months after t1. We then subtracted the value of the measure at t2 from that of t1.

Data on the measures of the effectiveness of the defect-fixing process as well as data on user interest in contributing to defect submission and resolution were extracted using Sourceforge advanced search on the projects' defect-tracking systems. For example, for 'defects submitted by team', we selected 'Bug' as 'tracker' and we highlighted all team members for 'submitted by', then the advanced search retrieves the defect reports which have been submitted by team members. As another example, for 'defects submitted by users', we subtracted the number of 'defects submitted by team' from the total number of defects submitted. Figure

3 shows a snapshot of a defect tracking system on Sourceforge.

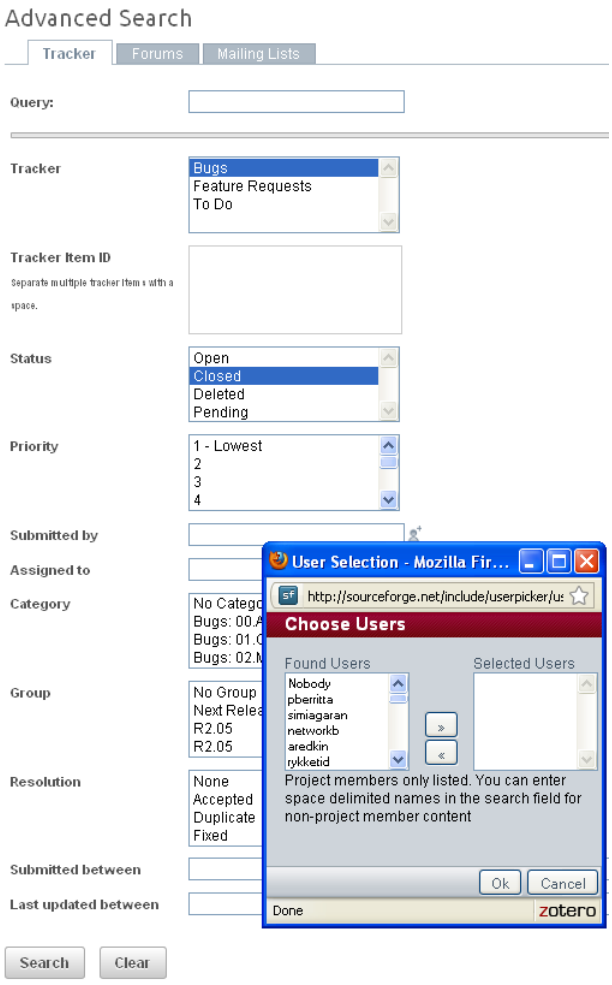

Figure 3. Defect tracking system on Sourceforge.net

IV. Data Analysis and Result

A. Validity and Reliability

PLS-Graph was used to undertake a regression analyses and also to examine reliability, convergent validity, and discriminant validity. The convergent validity tests how closely the items are associated with their underlying constructs [26]. According to Hair et al. [27] in order to have convergent validity all path loadings from construct to measures should be 0.5 or higher, ideally 0.6, or 0.7. As Table 2 shows, all factor loadings were higher than 0.7, and they are all significant since their t-value is higher than 1.96 (α=0.05). Another way to test convergent validity is to examine the average variance extracted (AVE) of constructs. AVE shows that the variance explained by the extracted factor is higher than the errors remained in the underlying indicators. As a rule of thumb, the value of AVE should be greater than 0.5 for each construct [27]. In this research AVE was 0.8872 which is much higher than 0.5.

TABLE II.
ITEM LOADINGS FOR MEASURES OF DEFECT-FIXING EFFECTIVENESS

Item	Loading	Sample Mean	Standard Deviation	T-Statistics
Sbmt	0.8292	0.8141	0.1323	6.2679
Asgn	0.9791	0.9708	0.029	33.7788
Fix	0.9696	0.961	0.0263	36.9248
Cmpl	0.981	0.9728	0.0283	34.6105

Discriminant validity tests whether the items that represent a latent construct differ from those that are not believed to constitute the latent construct. According to Gefen and Straub [28], in order to evaluate the presence of the discriminant validity, two conditions should be satisfied: firstly, each indicator should highly load on its associated construct. As Table 2 shows, this condition is met since all the loadings are higher than 0.7. Secondly, the square root of the AVE of each factor should be higher than any correlation amongst any pairs of constructs. This criterion was also met since the square root of the AVE, 0.941, is much higher than maximum correlation amongst any pairs of constructs. Table 3 exhibits the loading and cross-loading of all indicators of the model. As Table 3 shows item loadings in their corresponding columns are all higher than the loadings of the measures used to calculate the other variables.

Reliability is defined as the extent to which indicators of a latent construct correlate or move together [26]. There are different techniques to calculate the reliability, however the composite reliability is one of the most popular ones. Composite reliability has been employed in this research and is measured using PLS-Graph. Composite reliability is recommended to be higher than 0.7 to show a high level of reliability [27]. Composite reliability of 0.9691 advises that reliability is achieved in our research.

TABLE III. CROSS LOADINGS

Item	Defects resolved by users	Defects submitted by users	Effectiveness of defect-fixing process
Sbmt	0.3159	0.1716	0.8292
Asgn	0.5406	0.3547	0.9791
Fix	0.5450	0.3764	0.9696
Cmpl	0.5077	0.3508	0.9810
UserDefSub	0.2786	1.0000	0.3474
UserDefRes	1.0000	0.2786	0.5221

B. Analysis and Results

The data analysis for this research was carried out through partial least squares (PLS). The PLS method has been employed by researchers in recent years because of its ability to model latent variables under conditions of non-normality [29]. PLS allows the researchers to examine the relationships among the conceptual variables. It also allows the researcher to analyze how well the measures relate to the associated variable. PLS-Graph version 3.00 [30] and Smart PLS were used for data analysis of this research. Bootstrap re-sampling procedure with 200 samples was also employed to test the significance of all paths. Figure 4 shows the results of

testing the research model. The model explains %38.6 of the variance in user contribution to defect-resolution (R^2=0.386), and %23.5 of the variance in user contribution to defect submission (R^2=0.235).

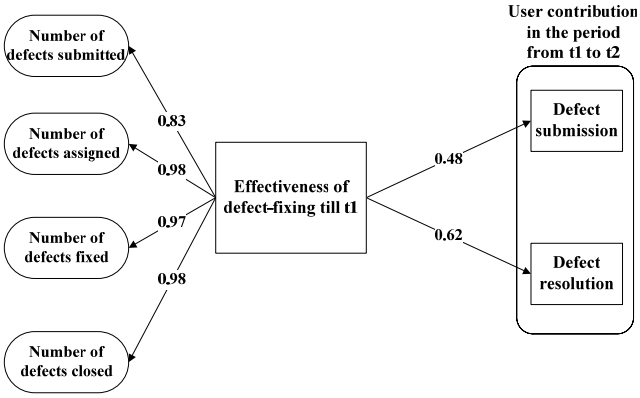

Figure 4. The results of base-line model

As indicated in Table 4, both hypotheses 1 and 2 were supported. A significant (at α=0.05) and positive direct effect (coefficient = 0.48) was found between the effectiveness of the defect-fixing process and user contribution to submit defects (H1). It demonstrates that OSS projects that are more committed and responsive to defect resolution process are more likely to receive user contribution in terms of reporting software defects. As predicted in Hypothesis 2, defect-fixing effectiveness was positively and significantly (at α=0.05) related to user contribution to resolving defects (coefficient = 0.62) indicating that the more responsive a project acts with respect to the defect fixing process, the more contribution the project can receive from the users in terms of completing the defects. These agree with expectations from theory of competency rallying.

To test the impacts of the control variables: project size, development status, project audience, and project age, a latent construct was made with all four control variables as its formative measures. The relationship between that latent variable and the dependent variable was insignificant at a 0.05 level meaning that they don't significantly influence user interest in contributing to the project.

TABLE IV.
PATH COEFFICIENTS FOR BASELINE MODEL

Relationship	Path coefficient	Sample Mean	Standard Deviation
Effectiveness of defect-fixing -> Defects submitted by users	0.4851	0.7362	0.1405
Effectiveness of defect-fixing -> Defects resolved by users	0.6214	0.4178	0.2469

V. DISCUSSIONS

In light of the insights from this study, we would like to pose two important implications for OSS scholars. First, prior research into OSS success has considered various success measures for OSS projects including the number of times an OSS project has been downloaded, the number of developers registered on the project website, and the amount of development activity undertaken in the project [15] [16] [23]. However, this paper introduced user interest in contributing to the project and process effectiveness (e.g. defect-fixing effectiveness) as other potential indicators of success in an OSS environment. This opens new avenues for future research to examine more innovative success measures for OSS projects. Second, our paper contributes to the OSS literature by augmenting our understanding of the implications of defect-fixing effectiveness to OSS success in terms of user interest in contributing to this process. Prior research on OSS projects has implied the significant role of effective defect-fixing in impacting OSS projects positive outcomes, but has not empirically investigated this relationship [6] [9] [10].

In addition to the research implications, our study has several implications for OSS projects' managers as well as corporations interested in adopting open source development style or adopting OSS software. Firstly, structural equation modeling of this research showed that OSS projects that operate their defect resolution process more responsibly and effectively till any period of time are more likely to receive defect reports from the community in the subsequent period. Such projects are also more likely to have a higher user participation in their defect-fixing process. Such contributions could result in higher product quality. This is of value for OSS project managers as well as for organizations attempting to adopt an OSS development style because it shows that OSS projects interacting with the community have a higher chance of receiving contributions from the community if they are perceived as being responsive and sympathetic to the user community's needs.

Secondly, project managers should be aware of the fact that simply attracting a high number of developers or download rate, or having a high level of development activity might not guarantee that their project will be successful in other respects. Process effectiveness (e.g. defect-fixing effectiveness) and user contribution (i.e. to the defect-fixing process) are two other dimensions introduced in this paper as indicators of success in an OSS environment.

Thirdly, defect-fixing effectiveness is an indicator that corresponds nicely to practitioners' concerns regarding open source software in that responsiveness to customer needs is one of the two most frequently cited concerns of IT practitioners adopting OSS [31]. By scale development for the construct of defect-fixing effectiveness based on the theory of competency rallying, this paper enables organizational users to assess a particular OSS with respect to defect removal and quality assurance. Moreover according to our study, another way for organizational users to evaluate a particular OSS would be measuring the amount of contribution that users make to a certain OSS (e.g. contributing to the defect-fixing process).

VI. LIMITATIONS

One limitation of this research is the generalizability of the results. First, we limited our sample to five project categories, potentially limiting the generalizability of the results across all project categories. Although we don't see any reason that the findings cannot be generalized across all project topics, this issue remains an empirical question. Another limitation arises from the fact that we selected the sample from projects registered on Sourceforge. Since these projects might differ from those OSS projects that have not been registered on Sourceforge, this could also limit the generalizability of the results.

VII. CONCLUSIONS

Since OSS development relies on volunteer developers and voluntary contributions from the user community, this paper studied the success of OSS projects in attracting user contributions. The focus of our study was to examine the relationship between the effectiveness of the defect-fixing process and user interest in contributing to this process in an OSS environment. The data collected from 1481 OSS projects hosted on Sourceforge confirmed that a more effective defect fixing process results in a higher user contribution to defect submission and defect resolution.

REFERENCES

[1] J. Colazo and Y. Fang. "The impact of license choice on open source software development activities," *Journal of the American Society for Information Science and Technology*, Vol. 60, No. 5, pp. 997-1011, 2009.

[2] S. Chengalur-Smith and A. Sidorova. "Survival of open-source projects: A population ecology perspective," *In Proceedings of 24th International Conference of Information Systems*, Atlanta, GA, 2003.

[3] S. Krishnamurthy. "Cave or community? An empirical examination of 100 mature open source projects," Working Paper, University of Washington, Bothell, Bothell, WA, 2002.

[4] K. J. Stewart and S. Gosain. "The impact of ideology on effectiveness in open source software development teams," *MIS Quarterly*, Vol. 30, No. 2, pp. 291-314, 2006a.

[5] Y. Fang and D. J. Neufeld. "Understanding sustained participation in open source software projects," *Journal of Management Information Systems*, Vol. 25, No. 4, pp. 9-50, 2009.

[6] K. Crowston. J. Howison and H. Annabi. "Information systems success in free and open source software development: Theory and measures," *Software Process: Improvement and Practice*, Vol. 11, No. 2, pp. 123-148, 2006.

[7] A. Mockus and D. Weiss. "Interval quality: Relating customer-perceived quality to process quality," *In Proceedings of International Conference on Software Engineering*, Leipzig, ACM Press, pp. 733–740, 2008.

[8] V. Venkatesh, M. G. Morris, G. B. Davis and F. Davis. "User acceptance of information technology: Toward a unified view," *MIS Quarterly*, Vol. 27, No. 3, pp. 425-478, 2003.

[9] K. Crowston. H. Annabi and J. Howison. "Defining open source software project success," *In Proceedings of the 24th International Conference on Information Systems*, Seattle, WA, 2003.

[10] V. Garousi. "Evidence-Based Insights about Issue Management Processes: An Exploratory Study," *In Proceedings of the International Conference on Software Process*, Vancouver, Canada, 2009.

[11] J. S. C. Hsu. C. L. Chan. J. Y. C. Liu and H. G. Chen. "The impact of user review on software responsiveness: Modeling requirements uncertainty," *Information & Management*, Vol. 45, No. 4, pp. 203-210, 2008.

[12] J. Bragge and H. Merisalo-Rantanen. "Engineering E-Collaboration Processes to Obtain Innovative End-User Feedback on Advanced Web-Based Information Systems," *Journal of the Association for Information Systems*, Vol. 10, Special Issue, pp. 196-220, 2009.

[13] J. Feller. P. Finnegan. J. Hayes and P. O'Reilly. "Institutionalising information asymmetry: governance structures for open innovation," *Information Technology & People*, Vol. 22, No. 4, pp. 297-316, 2009.

[14] N. Livari. "Constructing the users" in open source software development: An interpretive case study of user Participation," *Information Technology & People*, Vol. 22, No. 2, pp. 132-156, 2009.

[15] K. J. Stewart, A. P. Ammeter and L. M. Maruping. "Impact of license choice and organizational sponsorship on success in open source software development projects," *Information System Research*, Vol. 17, No. 2, pp. 126-144, 2006.

[16] C. Subramaniam, R. Sen and M. L. Nelson. "Determinants of open source software project success: A longitudinal study", *Decision Support Systems*, Vol. 46, No. 2, pp. 576-585, 2009.

[17] J. Colazo. *Innovation success: an empirical study of software development projects in the context of the open source paradigm*, PhD dissertation, University of Western Ontario, 2007.

[18] J. Long. *Understanding the Creation and Adoption of Information Technology Innovations: the Case of Open Source Software Development and the Diffusion of Mobile Commerce*, PhD dissertation, The University of Texas at Austin, 2004.

[19] C. A. Conley. *Design for quality: the case of open source software development*, PhD dissertation, New York University, 2008.

[20] J. D. Herbsleb and A. Mockus. "An Empirical Study of Speed and Communication in Globally Distributed Software Development," *IEEE Transactions on Software Engineering*, Vol. 29, No. 3, pp. 1-14, 2003.

[21] K. J. Stewart and S. Gosain. "The Moderating Role of Development Stage in Free/Open Source Software Project Performance," *Software process improvement and practice*, Vol. 11, pp. 177-191, 2006b.

[22] B. Katzy and K. Crowston. "A process theory of competency rallying in engineering projects. Centre for Technology and Innovation Management," working paper, Munich, Germany, available: http://cyowston.syr.edu/papers, 2000.

[23] K. Crowston and B. Scozzi. "Open source software projects as virtual organizations: Competency rallying for software development," *IEE Proceedings on Software*, Vol. 149, No. 1, pp. 3–17, 2002.

[24] B. R. Katzy and K. Crowston. "Competency rallying for technical innovation - the case of the Virtuelle Fabrik," *Technovation*, Vol. 28, pp. 679-692, 2008.

[25] A. H. Ghapanchi and A. Aurum. "Measuring the Effectiveness of the Defect-Fixing Process in Open Source Software Projects," *In The 44th Hawaii International Conference on System Sciences*. Hawaii, 2011.
[26] D. Straub, M. C. Boudreau and D. Gefen. "Validation Guidlines For IS Positivist Research," *Communications of the Association for Information Systems*, Vol. 13, pp. 380-427, 2004.
[27] J. F. Hair, W. C. Black, B. Babin, R. E. Anderson and R. Tatham. *Multivariate Data Analysis*, New Jersey, Pearson Education Inc, 2006.
[28] D. Gefen and D. Straub. "A Practical Guide To Factorial Validity Using PLS Graph: Tutorial And Annotated Example," *Communications of the Association for Information Systems*, Vol. 6, pp. 91-109, 2005.
[29] W. W. Chin. *The Partial Least Squares Approach for Structurational Equation Modeling, in Modern Methods for Business Research*, G. A. Marcoulides (ed.), Mahwah, NJ: Lawrence Erlbaum Associates, pp. 295-336, 1998.
[30] W. W. Chin. *PLS-Graph Manual Version 3*, 2001.
[31] B. Golden. *Succeeding with Open Source*, Addison-Wesley, Boston, 2004.

Amir Hossein Ghapanchi is a lecturer at the School of Information and Communication Technology, Griffith University, Australia. Amir has served in several national information system projects. His main research interests include open source software project management, IT human resource management, e-government planning and implementation, and multi-criteria decision making (MCDM). Amir has published in several prestigious management and information system journals such as Journal of Systems and Software, International Journal of Information Management, and International Journal of Project Management.

Aybuke Aurum is an associate professor at the School of Information Systems, Technology and Management, University of New South Wales, Australia. She received her Ph.D. in computer science. She has over 80 publications including three edited books, namely "Managing Software Engineering Knowledge", "Engineering and Managing Software Requirements", and "Value-Based Software Engineering". Her research interests include value-based approach in software development, management of software development process and requirements engineering. She is on the editorial boards of Requirements Engineering Journal and Information and Software Technology Journal.

Farhad Daneshgar is a Senior Lecturer at the School of Information Systems, Technology and Management, University of New South Wales, Australia. Farhad is a member of the editorial board in three IS/KM Journals, and the leader of Knowledge Management Research Group at SISTM. He is the creator of the awareness net modeling language and his research interests include e-collaboration, analysis and design of collaboration-support systems, collaborative learning, and mass customisation and has published extensively in these areas.