Sampling Process Model for Coordination and Communication in Free/Open Source Software Projects

Preet Kanwal
Research Scholar
Department of Computer Science and Applications,
Panjab University, Chandigarh
India

Anu Gupta
Associate Professor
Department of Computer Science and Applications,
Panjab University, Chandigarh
India

Ravinder Kumar Singla
Professor
Department of Computer Science and Applications,
Panjab University, Chandigarh
India

ABSTRACT

The Free/Open Source Software (F/OSS) development environment consists of three components: The development process, the community of software development volunteers and the coordination and communication tools. The rise and expansion of Internet make this concept of cooperative software development model a dominant force in comparison to the traditional software development. Various project hosting sites for F/OSS projects across an entire range of applications have come up offering a multitude of services for developers as well as users. Various aspects of F/OSS are being studied by researchers. Coordination and Communication become key factors in F/OSS development for information dissemination among its decentralized and geographically spread-out teams of volunteers. It becomes important to choose the right sample to study this aspect. This paper proposes a four-phase sampling process model especially suited for studies related to coordination and communication aspects in F/OSS although it can be extended to all aspects of F/OSS development with suitable changes in the parameters.

Keywords

F/OSS; FLOSS; Coordination; Communication; CC Tools; SVN, SRDA; Source List ; Sampling Process Model.

1. INTRODUCTION

The global reach with 24x7 availability of Internet has made Free/Open Source Software (F/OSS), an innovative development methodology, a dominant force [1-2]. In general, the F/OSS development environment consists of three components: the development process, the community of volunteers [3] and the coordination and communication tools (CC Tools). The cooperative and group assisted methods of software creation, modification and distribution allowing code reuse by distributed self-learning and self-organizing teams form the backbone of F/OSS development in a decentralized and distributed software development environment. Software repositories and allied coordination and communication infrastructure provide a base for cooperative development. Thus, the tools which support cohesion among geographically spread-out volunteers for effective coordination and communication have become an important aspect in the F/OSS development process.

With F/OSS research becoming extensive on each aspect of its development environment, it is very important to understand the process of selection of case studies for effective and meaningful research.

2. LITERATURE REVIEW

Internet has been observed as a necessary condition for F/OSS to evolve along with cooperative customs which could allow co-developers to be attracted [4]. Crowston *et al*. have based their study of understanding Free/Libre Open Source Software (FLOSS) success on information system (IS) research in order to identify processes that enhance the performance of FLOSS development teams. They observed that success indicators in traditional closed software development were influenced by 'use environment' like system quality, use, user satisfaction and organizational impacts. They further highlighted the fact that FLOSS success tends to look more at the 'development environment' which is publicly available [5]. The studies on F/OSS development process have mainly focused on the participants and the processes involved in this form of development to understand the success of this model. The research areas have been categorized into three main perspectives: Developers, Metrics and Tools [6].

The researchers have highlighted the importance of formal and informal communication in coordination of software development in general [7]. CC Tools which can be used by F/OSS projects have been suggested [8]. A study, while acknowledging the fundamental relevance of communication in F/OSS development, proposed an Extended Information System Model comprising of eight interrelated dimensions which links quality, success, communication and contributions in F/OSS projects [9]. The majority of development activity in F/OSS utilizes asynchronous communication which has the advantage of maintaining total record of communications [10, 11, 12]. Researchers used random sampling to test whether the process maturity is related to the success of F/OSS projects, especially where processes related to coordination and communication are concerned [13]**.** Another study used two data sets of different sizes – successful projects and a random set from *SourceForge.net* to explore the possible benefits of CC Tools on efficiency of open source projects. The study focused on tools offered by Sourceforge.net only. Various parameters used for analysis included number of developers, project age, number of downloads, web hits, project size (in bytes), lines-of-code and development status. Negative influence of tool adoption on efficiency was found for data set comprising of successful projects whereas data set of random projects showed positive influence. Thus, mixed results so obtained were non-conclusive regarding effect of CC Tools on efficiency of Open Source projects. Many researchers have acknowledged the fact that the selection of Open Source projects to study the impact of CC Tools, must be made very carefully and more extensively [14].

Most of the research work on F/OSS has been focused on a few case studies of widely accepted successful projects. Majorly convenience sampling has been adopted across the literature for case selections. For an objective view of F/OSS concept, it is imperative that larger samples need to be considered and selected. This raises the question as to how to select the sample in accordance with the aspect to be studied.

The following sections present the methodology used to develop a Sampling Process Model to select case studies from a single hosting site with focus on research in CC Tools.

3. METHODOLOGY

3.1 Aim of the Study

The aim of the study is to develop a Sampling Process Model for selecting case studies for research on coordination and communication aspect in development of F/OSS projects.

3.2 F/OSS Empirical Research – The Longitudinal Aspect

Literature review revealed two main approaches of F/OSS Development process. The first approach comprised of seven iterative phases: Problem Discovery, Finding Volunteers, Solution Identification, Code Development and Testing, Code Change Review, Code Commit and Documentation and Release Management [15]. The Second approach identified six stages: Planning, Pre-Alpha, Alpha, Beta, Stable and Mature [16]. In actual practice, both these approaches are integrated. This integration is depicted in Fig. 1 where the first approach representing the Developer's perspective of development works iteratively for each stage described in the second approach which is from a Project's Progress perspective. The volunteers may join the project at any phase/stage and the developer pool of a project may change over time. Trace data about volunteers, code development, time spans as well as communication is recorded in various CC Tools and is publicly available through repositories and project hosting sites.

Since the source code is available and can be modified by anyone who has skills and interest, F/OSS projects are developed incrementally over longitudinal time frames. The software requirements and the corresponding coordination and communication requirements of the volunteers may change over this incremental longitudinal development span. Longitudinal research allows researchers to study the same data set over an extended time span in order to observe the changes over time. Thus a longitudinal study of the F/OSS projects would provide significant insight into the various aspects of development process, coordination and communication requirements and the significance of various CC Tools.

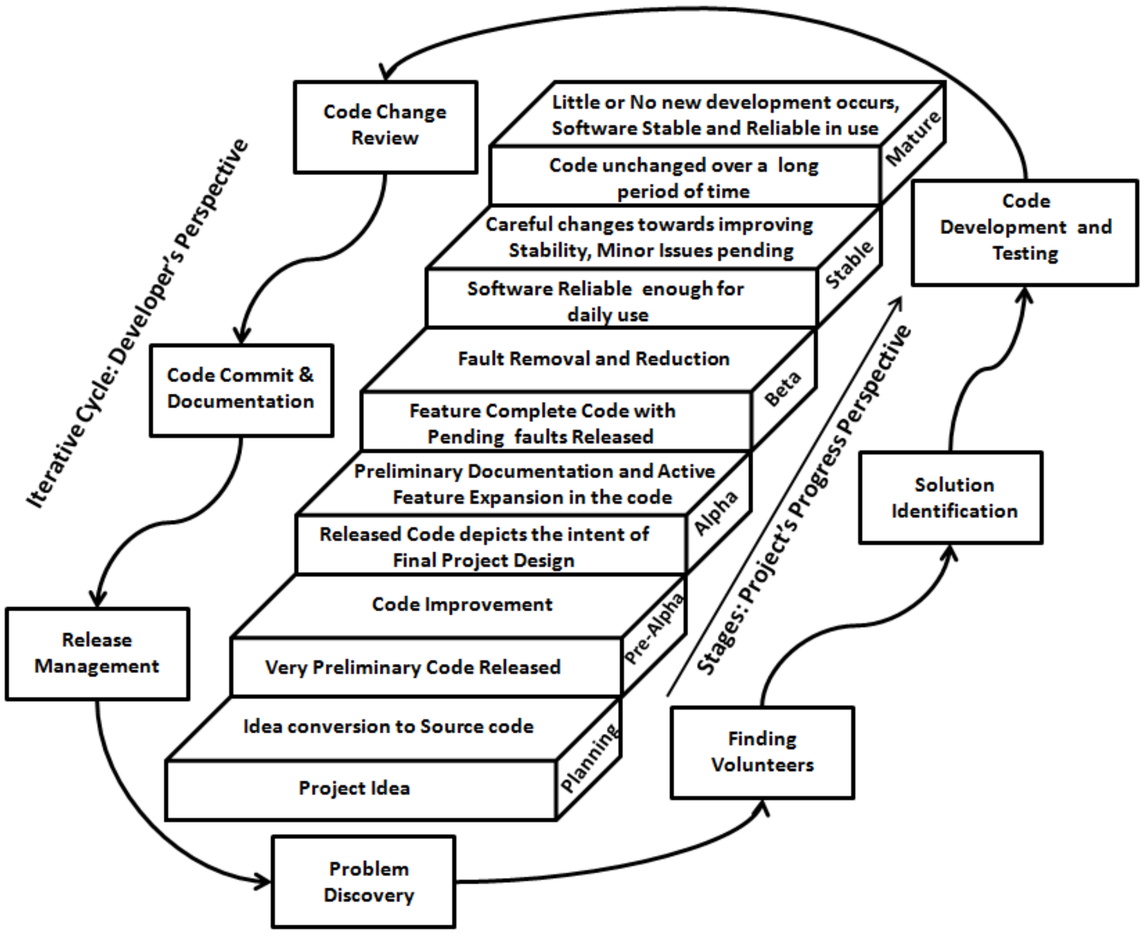
Code Change Review
Code Commit & Documentation
Release Management
Problem Discovery
Finding Volunteers
Solution Identification
Code Development and Testing
Iterative Cycle: Developer's Perspective
Stages: Project's Progress Perspective
Little or No new development occurs, Software Stable and Reliable in use
Code unchanged over a long period of time
Mature
Careful changes towards improving Stability, Minor Issues pending
Software Reliable enough for daily use
Stable
Fault Removal and Reduction
Feature Complete Code with Pending faults Released
Beta
Preliminary Documentation and Active Feature Expansion in the code
Released Code depicts the intent of Final Project Design
Alpha
Code Improvement
Very Preliminary Code Released
Pre-Alpha
Idea conversion to Source code
Project Idea
Planning

Fig 1: F/OSS Development Process: An Integrated Cycle - Stage Approach

3.3 Sampling Process and Model Development

Sample design i.e. deciding the way of selecting a sample is the first requirement for model development. The Sampling process comprises of several stages [17]:

- Clearly defining the Population of concern
- Deciding the Sampling Unit for the study.
- Specifying a Source List or Sampling Frame, a set of items or events possible to measure
- Determining the Sample Size
- Determining the parameters of interest and Data which can be collected
- Specifying a Sampling Method for selecting items or events from the frame
- Implementing the Sampling Plan
- Sampling and Data Collecting

For sampling, an initial research setting for data collection is required to be fixed in accordance with the aim of the study. The foremost requirement of proposed sampling process is to choose F/OSS project hosting site for selecting candidate projects or case studies. The type and number of data sets as well as the sources of data collection have to be predetermined. The criteria may need to be refined during the actual selection depending upon the constraints of the chosen study. The following sections explain the various steps followed for sample selection to study the role of CC Tools in F/OSS development by way of proposing a project sampling process model and the challenges of making the appropriate selections.

4. RESEARCH SETTING FOR DATA COLLECTION

4.1 Choosing Project Hosting Site for Sample Selection

Various project hosting sites for open source software have come up offering a multitude of services for developers as well as users. Choosing a single source allows control for difference in available tools and project visibility [18]. For this particular study, *SourceForge.net* [19] has been chosen as it is one of the leading F/OSS project hosting sites. It offers user friendly interface and provides statistics for various aspects related to project development. The project support is provided through code repositories, forums, bug trackers, mailing lists etc.

4.2 Formulation of Data Sets

Formulation of data sets is based on the following parameters:

- **Project Category** - The interface provided by *SourceForge.net* allows search and listing criteria for projects based on filters which include category, translation, license, programming language, development status and operating system. The data sets would be formed from two broad categories: System Software Projects having significance for developers and system manufacturers and Allied Applications Projects having significance for those who integrate the F/OSS applications with their core area [20].

- **Code Repository** - Various source code management tools being used across projects on *SourceForge.net* include CVS, SVN, Git and Mercurial. Subversion (SVN) has been adopted on a large scale across the hosting site thus allowing flexibility to choose a large sample for the study.

- **Project Popularity** - The projects on *SourceForge.net* can be listed in sorted order by factors like most popular, relevance, date of last update, name and rating. User interest is an important point of consideration for the selection process for studies taking into account the volunteers' perspective. The number of downloads represents an objective measure of user interest in the project and its success. Thus, the data sets would be formed from the chosen categories in sorted order of popularity (in terms of weekly downloads).

4.3 Data Collection Sources

The various sources identified for collecting data for the study are as below:

- ***SourceForge.net*** - Manual inspection of the hosting site is required to create Source List or Sampling Frame from which sample projects would be chosen and then gather project information which is not available through public code repository and research data archives.

- ***Subversion* (SVN) Repositories** - F/OSS for SVN repository access would be used to access the project logs for extracting developer related data.

- ***SourceForge Research Data Archive* (SRDA)** implemented by University of Notre Dame, Notre Dame, USA. - The access to the data archive comprising of monthly dumps of over 100 relations/tables is provided via user-id and password to academic and scholarly researchers only after submitting a research proposal and signing the agreement/sublicense for data access. The data can be accessed through a web-based form for executing SQL queries against the relational database [21, 22].

5. THE SAMPLING PROCESS MODEL

A Four-phase model is proposed for the sampling process as depicted in Fig. 3 and is explained below:

5.1 Phase 1 - Data Collection Framework Design

It is important to design a framework to collect data in accordance with the criteria set in the research setting. The availability of appropriate data and in turn the data sources for F/OSS study influence the sampling process. The Fig. 2 shows the framework derived from Gupta and Singla [18] used for this study.

5.2 Phase 2 - Candidate Projects Identification

The goal of this phase is to find all possible candidate projects which meet the major criterion in accordance with the research setting for the data collection for the study. Only the high-level requirements are applied to initiate the selection process.

The Identification Phase started with searching for candidate projects on *SourceForge.net*, the hosting site selected for studying the impact of CC Tools on Open Source Projects. The search interface provided by the site was used to filter projects. The projects were sorted based upon "Popularity". The filters used were: Category and Development Status. The decision to use the "Status" filter in addition to the "Category" filter was based on two points: to explore the role of CC Tools and the requirement of longitudinal data for the study. *SourceForge.net* categorizes projects according to development status from 1-7 i.e. Planning, Pre-Alpha, Alpha, Beta, Production/Stable, Mature and Inactive. The decision

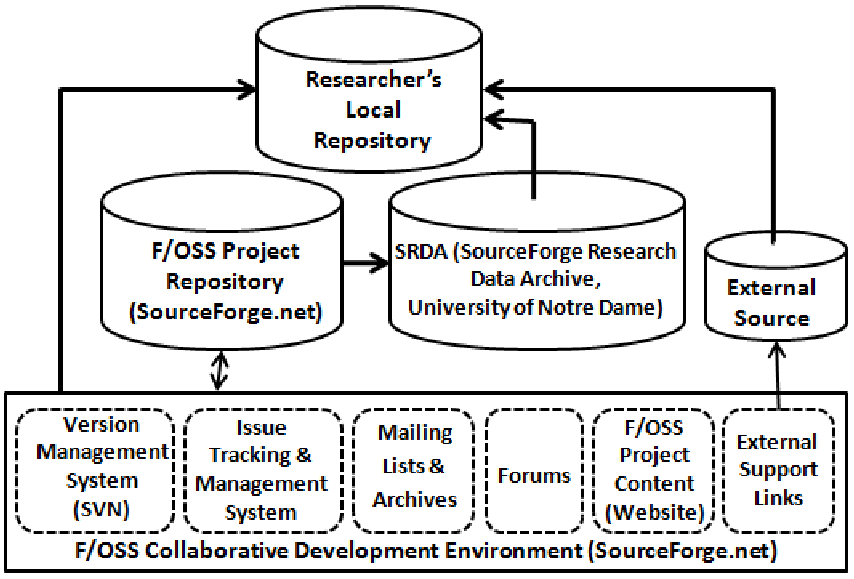
Researcher's Local Repository
F/OSS Project Repository (SourceForge.net)
SRDA (SourceForge Research Data Archive, University of Notre Dame)
External Source
Version Management System (SVN)
Issue Tracking & Management System
Mailing Lists & Archives
Forums
F/OSS Project Content (Website)
External Support Links
F/OSS Collaborative Development Environment (SourceForge.net)

Fig 2: Framework for Data Collection

for a cut-off of 5 i.e. production/stable development status was based on the fact that projects are at similar stage of development so that comparable archived longitudinal data for projects' progress would be available. Moreover, the selected projects are expected to remain active during the study period with scope for their movement to a higher development status giving opportunity to study the entire development process. On the day of fixing the finite population/universe from which the sample would be chosen, the projects on the hosting site were organized in seven categories – Software Development, System, Internet, Communications, Scientific/Engineering, Multimedia and Games/Entertainment. Out of the seven categories, the Software development and System category projects fall under the broad category of System Software projects and the remaining five categories fall under Allied Applications. The research decision taken with regards to category was to choose projects from all available N (i.e. 7) sub-categories to remove any selection bias and form two data sets as per research setting. It was further decided to fix an equal upper limit (M) for choosing an initial number of projects from each sub-category to form the Source List out of which the final sample would be drawn.

A query, using project names from *SourceForge.net,* was formulated to retrieve project details comprising of its unique identification number, date of registration with *SourceForge.net* and its SVN usage status from SRDA [23]. To formulate the query, the Schema and Tables details along with ER diagrams provided by SRDA were studied in detail [21, 22]. The query was executed, against the monthly dump of latest schema available, for all N (i.e. 7) lists comprising of M projects each. SRDA uses a binary parameter to depict usage of SVN where 1 implies that the project uses SVN and 0 otherwise. From the query output text files, the details were manually ported to a spreadsheet package. The projects which were not using SVN were struck off the list.

The identification phase provided N (i.e. 7) lists of projects corresponding to N categories as on *SourceForge.net* sorted according to "popularity", comprising of projects at production/stable status and using SVN at the time of selection process. The lists comprise of projects fulfilling the high level requirements fixed in the research setting of the data collection.

5.3 Phase 3 - Candidate Projects Screening

In this step, refined criteria are applied to find the best candidate projects which satisfy the longitudinal aspect of the study in respect of coordination and communication in F/OSS development process.

Though *Phase 2* identifies candidate projects according to the criteria fixed in the research setting, yet further screening was required to fulfil the requirements of the study under consideration. The criteria for refinement include following parameters:

- **Number of Developers** - A cut-off of 5 developers was taken. SRDA query was formulated and executed to find the number of developers from the monthly dump of the schema used for queries in *Phase 2* to ensure data consistency. The projects with number of developers less than 5 were rejected because very few developers would result in lesser use of a variety of CC Tools resulting in insufficient data for proper study of Communication and Coordination aspect.
- **Date of Registration** - A cut-off on date of registration on *SourceForge.net* had to be implemented to ensure that projects falling outside the scope of longitudinal span chosen for the study are not included in the sample data sets.
- **Multiple Category Resolution** - For projects listed simultaneously in multiple categories, the category was resolved after studying the usage, application and intended audience as listed at *SourceForge.net* page of the project as well as project's own website.
- **Current Code Repository** - For each project in the output list of identification phase, *SourceForge.net* page of the project was checked manually to find out its current SVN usage. Of the selected projects, a few had moved from SVN to other code repositories. Such projects were rejected due

to partial or zero availability of data for the time span chosen for study. Moreover, a few projects were using more than one code repository apart from SVN. The dates and count of SVN commits of such projects were observed during the time span chosen for the study and the decision regarding keeping or discarding a project were taken in accordingly. The Screening phase results in selection of projects which best fit the parameters of the study i.e. the Source List for the study. From this the actual sample data sets have to be formulated.

5.4 Phase 4 - Data Sets Sampling

Finally, in *Phase 4*, the projects are sampled into two data sets based on categories as per the criterion fixed in the research setting. Since popularity measure or downloads has been chosen to sort the candidate projects, both the sets of equal size comprise of the most downloaded projects in each category.

The decision to choose equal sized data sets was based on the following reason: As described in [17], in stratified sampling "if the purpose happens to compare differences among strata, then equal sample selection from each stratum would be more efficient even if strata differ in size". In this study, the sample comprising of two data sets have to be formulated from a heterogeneous population comprising of N individually homogeneous categories. These N categories represent N strata which are individually more homogeneous than the total population due to their classification by the hosting site based on parameters of usage and intended audience.

The proposed Four Phase Sampling Process Model depicted in Fig. 3 shows the step-by-step procedure followed for the Sample selection process.

6. THE CHALLENGES OF PROJECT SELECTION

The selection process has been a tedious process due to certain constraints which arose during the course of the selection process.

- **Dynamic Nature of the Hosting Site** - The sorting order of the projects keep changing according to the number of downloads i.e. popularity on the hosting site. It was observed from manual inspection of the site that some projects occurred in multiple categories. If the lists corresponding to each of the N categories on the hosting site were fixed at a larger time differential then this could have led to sampling error. Consider the following scenario: Category wise lists have been fixed on separate days and a project which occurs in more than one category gets fixed in one list on the first day. Before the fixing of the next category in which it occurs, it gets downloaded thus changing its ordering in '*sort by popularity'*. If both the categories had been selected with a larger time differential between them, then there is a possibility of the project being low in priority in first category with chances of rejection being high. The same project may occur on higher order in the next list with a chance of being selected in final list. The problem would arise if the project had more relevance in the first category than the other thus leading to sampling error. Thus, the projects to form the Source Lists from which the final sample had to be drawn needed to be fixed within the shortest possible time frame. To ensure minimum sampling error on this account, the initial list of N×M projects had to be prepared within the smallest time frame possible within the same day.
- **Retrieving Unique Project Identification Number**- From the detailed study of SRDA database tables, schemas and ER Diagrams, it was found that a project's identification number is the most important parameter to be used in queries to be executed against the relational database. Manual inspection of *SourceForge.net* showed that this number was not listed at the site for all projects. This meant that project identification number had to be retrieved from SRDA. For this, care had to be taken to put the exact names of projects as listed on *SourceForge.net* site in the first query to retrieve the corresponding unique project identification number, date of registration and SVN usage for the entire lists of N×M projects in the initial list.
- **Manual Porting of SRDA Query Results -** The results of the query executed against the SRDA were in the form of a text file with entries for each project on a separate line. The result of query is not sorted according to "popularity" of a project. Moreover, only entries of projects which used SVN had to be retained. This partial set of entries could not be directly ported in order of popularity to the spreadsheet package. So, this had to be done manually at this stage.
- **Multiple Code Repositories on *SourceForge.net*** - In projects where other code repositories are being used in addition to SVN, the code commit activity was checked from code repository statistics and history through *SourceForge.net*. Comparisons were made amongst commit activity in all repositories of a project through manual observation. Projects which were maintaining simultaneous partial data on more than one repository had to be rejected. Moreover, in certain cases statistics from other repositories were not available for comparison.
- **Multiple Category Resolution -** Some projects were found to be listed under more than one of the seven categories on *SourceForge.net.* This verification was done manually as well as filter option of spreadsheet package was used. Category wise coding was done in order of popularity in each list. For projects having multiple codes, category had to be resolved by manually studying the project page on *SourceForge.net*, its website and other related sites for checking its application usage and intended audience. The coding process was tedious manual work and verification process took considerable time.
- **Identifying Criteria to be used in Candidate Screening Phase -** Proper care had to be taken to identify the criteria to be used for screening phase keeping in view the focus of the study and to minimise the sampling bias.
- **Manual Inspection of the Hosting Site -** Traditional way of data collection through spidering using Perl scripts has been reported as time and resource consuming process with significant challenges in cleaning, screening and interpreting the data. Moreover, *SourceForge.net* has introduced defence mechanism against spidering [24]. Therefore, the gaps in the data accessed from research collaboratory SRDA at different stages had to be filled through manual inspection. The constraints of the study did not allow automated application to be developed at this stage.

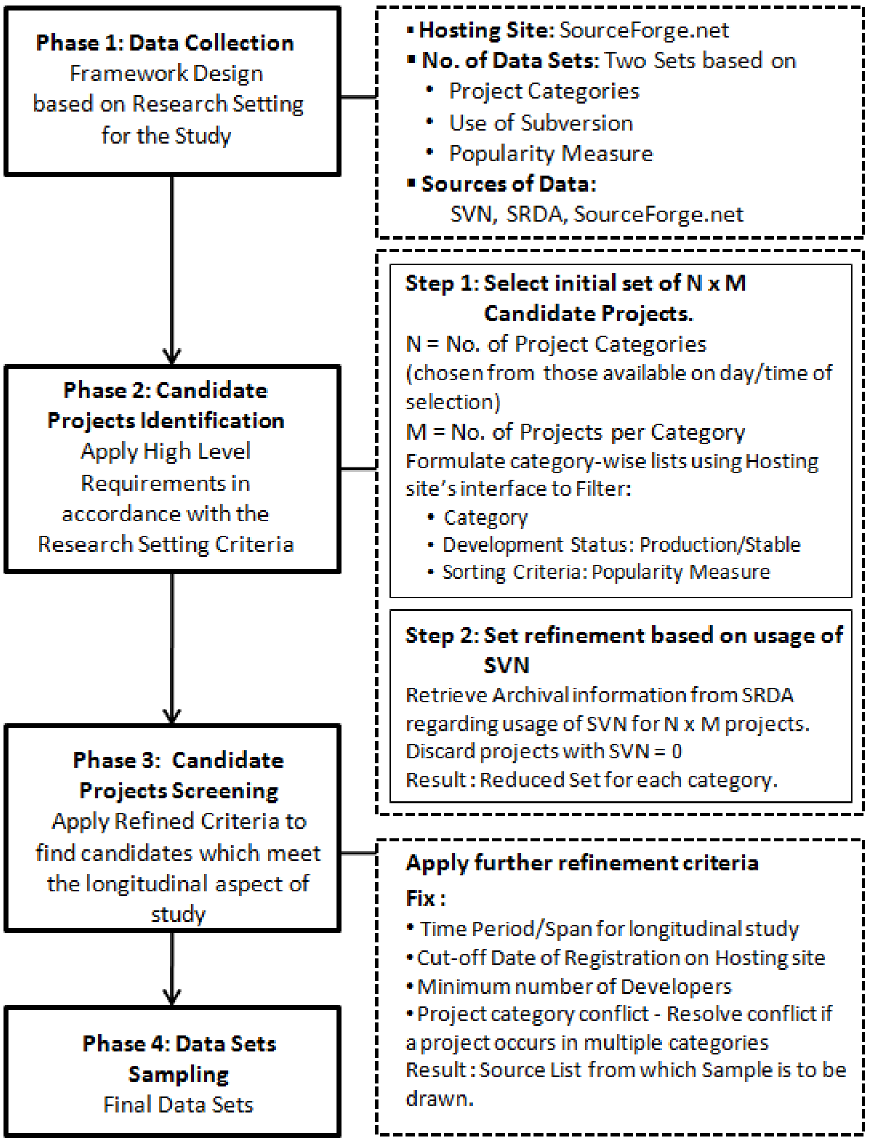
Phase 1: Data Collection
Framework Design based on Research Setting for the Study
▪ Hosting Site: SourceForge.net
▪ No. of Data Sets: Two Sets based on
• Project Categories
• Use of Subversion
• Popularity Measure
▪ Sources of Data: SVN, SRDA, SourceForge.net
Phase 2: Candidate Projects Identification
Apply High Level Requirements in accordance with the Research Setting Criteria
Step 1: Select initial set of N x M Candidate Projects.
N = No. of Project Categories (chosen from those available on day/time of selection)
M = No. of Projects per Category
Formulate category-wise lists using Hosting site's interface to Filter:
• Category
• Development Status: Production/Stable
• Sorting Criteria: Popularity Measure
Step 2: Set refinement based on usage of SVN
Retrieve Archival information from SRDA regarding usage of SVN for N x M projects.
Discard projects with SVN = 0
Result : Reduced Set for each category.
Phase 3: Candidate Projects Screening
Apply Refined Criteria to find candidates which meet the longitudinal aspect of study
Apply further refinement criteria
Fix :
• Time Period/Span for longitudinal study
• Cut-off Date of Registration on Hosting site
• Minimum number of Developers
• Project category conflict - Resolve conflict if a project occurs in multiple categories
Result : Source List from which Sample is to be drawn.
Phase 4: Data Sets Sampling
Final Data Sets

Fig 3: Framework for Proposed Four Phase Sampling Process Model

Table 1. Sampling Process Model - Formulation of Data Sets

Phase	**Description**	**Data Set 1 (System Software Projects)**		**Data Set 2 (Allied Applications)**				
Phase 2 : Step 1	Categories[a] (N)	SD 1	S 2	I 3	C 4	SE 5	GE 6	M 7
	Total Projects	6852	5174	6296	3602	3548	2845	3271
	Initial Set (M)	200	200	200	200	200	200	200
Phase 2 : Step 2	Projects using SVN	105 (52.5%)	96 (48.0%)	111 (55.5%)	109 (54.5%)	108 (54.0%)	118 (59.0%)	107 (53.5%)
Phase 3	Source List	41 (39.0%)	30 (31.3%)	31 (27.9%)	30 (27.5%)	49 (45.4%)	49 (41.5%)	30 (28%)
Phase 4	Stratified Sampling	25	25	10	10	10	10	10
Sample Size = 100		50		50				

[a]Categories: SD - Software Development, S - Systems,
I - Internet, C - Communications, SE - Scientific/Engineering, GE - Games/Entertainment, M - Multimedia

7. MODEL APPLICATION

Table 1 summarizes the data sets formulation using the proposed Four Phase Sampling Process Model for research problem to study the longitudinal impact of CC Tools on F/OSS projects. The framework described in *Phase 1* has been utilized for preparing the Source List from which final sample has been formulated. As per the research setting, sample comprises of topmost projects sorted as per "popularity" measure on the hosting site on the day of selection, which satisfy the criteria of *Phase 2* and *Phase 3* respectively. Of the topmost 200 most popular projects in each available category, overall 53.86% projects used SVN. Of these 53.86%, 34.5% formed the Source List. The N=7 categories, divided into two broad categories as described in *Phase 2*, form the two data sets as listed in Table 1. Applying stratified sampling with equal sized sample selection from each stratum (category), two data sets of 50 projects each have been formulated in *Phase 4*. The topmost popular projects in each category in the Source List were taken to formulate the final data sets depending upon the broad category area.

The above data sets have been formulated for research problem to study the longitudinal impact of CC Tools on F/OSS projects. Depending upon the type of research problem, the proposed Sampling Process Model may be applied to related or similar longitudinal studies after requisite modifications in the criteria in each of the Phases. The number and size of data sets may vary depending upon the research questions to be addressed.

8. CONCLUSION

F/OSS developers rely on the coordination and communication infrastructure since they work in a decentralized and distributed development environment. Thus communication and the appropriate communication infrastructure, which provides flexibility to the volunteers to contribute as per their own convenience, is the key to progress of any F/OSS project over a period of time. Studying the effect of CC Tools on the development process is an important aspect of research to gain insight into this form of incremental collaborative development. The Four Phase Sampling Process Model proposed in this paper may find application in related or similar studies. It would find application for all researches which utilize multiple case-studies to gain insight into various aspects of F/OSS and software development in general.

9. ACKNOWLEDGMENTS

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

10. REFERENCES

[1] The Free Software Definition.

URL: http://www.gnu.org/philosophy/free-sw.html last accessed on March 06, 2012.

[2] Open Source Software Definition. URL: http://www.opensource.org/docs/osd last accessed on March 06, 2012.

[3] Crowston, K., Annabi, H., Howison, J. and Masango, C. 2005. Effective Work Practices for FLOSS development: A model and propositions. In Proceedings of the 38th Hawaii International Conference on System Sciences – 2005.URL: http://www.computer.org/portal/web/csdl/doi/10.1109/HICSS.2005.222 last accessed on August 27, 2011.

[4] Raymond, E.S. 1999. The Cathedral and the Bazaar. Cambridge, Massachusetts: O'Reilly & Associates, 1999.J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp. 68-73.

[5] Crowston, K., Howison, J., and Annabi, H. 2006. Information Systems Success in Free and Open Source Software Development: Theory and Measures. Software Process: Improvement and Practice, 11(2), pp. 123–148.

[6] Preet Kanwal, Gupta, A. and Singla, R.K. 2013. Open Source Software Development: Exploring Research Perspectives. Emerging Trends in Computing, Informatics, Systems Sciences, and Engineering, Lecture Notes in Electrical Engineering 151, Springer Science+Business Media New York 2013. Pp. 607-617.

[7] Kraut, R.E. and Streeter, L.A. 1995. Coordination in Software Development. Communications of the ACM, 38(3), 1995, pp. 69-81.

[8] Gardler, R. 2011. Essential tools for running a community-led project. OSS Watch open source software advisory service website http://www.oss-watch.ac.uk/ URL: www.oss-watch.ac.uk/resources/communitytools.xml last accessed on January 4, 2012.

[9] Fernandes, S. 2011. Quality, success, communication and contribution in Open Source Software. In Proceedings of 5th International Workshop on Foundations and Techniques for Open Source Software Certification (OpenCert 2011), pp. 33-42.URL: http://opencert.iist.unu.edu/Papers/2011-paper-S2-A.pdf last accessed on January 4, 2012.

[10] Koch, S. and Gonzalez-Barahona, J. M. 2005. Open Source Software Engineering – The State of Research. First Monday, 10(SI-2), 2005.URL: http://firstmonday.org/htbin/cgiwrap/bin/ojs/index.php/fm/article/view/1466/1381 last accessed on August 09, 2010.

[11] Lundell, B., Lings, B., Ågerfalk, P. J., and Fitzgerald, B. 2006. The Distributed Open Source Software Development Model: Observations on Communication, Coordination and Control.URL: http://is2.lse.ac.uk/asp/aspecis/20060058.pdf last accessed on 27 August 2011.

[12] Canfora, G., Lanubile, F. and Mallardo, T. 2003. Can Collaborative Software Development Benefit from Synchronous Groupware Functions? URL: http://cdg.di.uniba.it/cdg/mallardo/papers/FA2003.pdf last accessed on January 16, 2012.

[13] Michlmayr, M. 2005. Software Process Maturity and the Success of Free Software Projects. In: Zieliński, K., Szmuc, T. (Eds.), Software Engineering: Evolution and Emerging Technologies. pp. 3–14. URL:http://www.cyrius.com/publications/michlmayr-process_maturity_success.pdf last accessed on January 31, 2012.

[14] Koch, S. 2009. Exploring the effects of Sourceforge.net coordination and communication tools on the efficiency of open source projects using data envelopment analysis. Empirical Software Engineering, 14, 2009, pp. 397-417.

[15] Sharma, S., Sugumaran, V. and Rajagopalan, B. 2002. A framework for creating hybrid-open source software communities. Information Systems Journal, 12, 2002, pp. 7–25.URL: http://in953.kelon.org/archives/in953/2004/papers/ISJAFrameworkForCreatingHybrid-OpenSourceSoftwareCommunities.pdf last accessed on January 12, 2012.

[16] Tiwari, V. 2010. Some Observations on Open Source Software Development on Software engineering Perspectives. International Journal of Computer Science & Information Technology (IJCSIT), 2(6), December 2010.M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.

[17] Kothari, C.R. 2010. Research Methodology Methods and Techniques. New Age International Publishers. Second Revised Edition, pp. 62-63.

[18] Gupta, A. and Singla, R.K. 2009. Evaluating User Participation in Defect Reporting among Free/Open Source Software Projects. Journal of Computer Science, 4(1), 2009, pp. 1387-1403.

[19] SourceForge.net; URL: http://sourceforge.net.

[20] Preet Kanwal, Gupta, A. and Singla, R.K. 2011. Open Source Software – Spectrum of Applications. In Proceedings of 5th Chandigarh Science Congress (CHASCON 2011), Panjab University, Chandigarh, 2011.

[21] SRDA; http://zerlot.cse.nd.edu/

[22] Antwerp, M. V. and Madey, G. 2008. Advances in the SourceForge Research Data Archive. In proc. of 4th International Conference on Open Source Systems - (WOPDASD 2008), Milan, Italy, September 2008. pp. 21-27.URL: http://zerlot.cse.nd.edu/mediawiki/images/f/fd/Srda_final.pdf last accessed on December 26, 2011.

[23] Gao, Y., Antwerp, M.V., Christley, S. and Madey, G. 2007. A Research Collaboratory for the Open Source Software Research. In Proceedings of First International Workshop on Emerging Trends in FLOSS Research and Development (FLOSS 2007) Minneapolis, MN, May 2007. URL: http://www.nd.edu/~oss/Papers/FLOSS07.pdf last accessed on January 24, 2012.

[24] Howison, J. and Crowston, K. 2004. The Perils and Pitfalls of Mining SourceForge. International Workshop on Mining Software Repositories (MSR 2004), Scotland, United Kingdom, May 23-28 2004.