



Empirical Studies in Software Engineering: A Survey

Suparna Ghanvatkar^{*}, Rohit Kumar BralaComputer Engineering & International Institute of Institute Information Technology,
Pune, Maharashtra, India

Abstract— *Software engineering and development is getting increasingly dependent on empirical studies performed on various datasets released by various open source and closed source software developed. These studies and analysis provide interesting results about the current development paradigms and aim to let the developers and researchers make informed decision regarding the approach of their software development and collaboration. In this paper, we try to present various studies performed in a comprehensive manner and try to realize the strategies adopted and their impact on the software development methodologies*

Keywords— *Collaborative software development, Empirical studies, Software engineering, Software life cycle, Empirical Software Engineering*

I. INTRODUCTION

Over the recent years, researchers and developers have been emphasizing the importance of empirical studies in software engineering and their results in gauging the effectiveness of the strategies adopted by developers and collaborators. A systematic literature review is required for analyzing, evaluating and interpreting the available research data which will allow fair evaluation and in-depth study of the available research data and material. The collaboration among developers or researchers is a very important factor for good project outcome. There is a lot of literature which tries to understand and evaluate various collaboration and development efforts undertaken in various projects. The empirical studies of project datasets like bug reports when studied on the background of literature which reviews collaboration strategies give us interesting approaches and results for the domain of software engineering.

The development, formulation and evaluation of software engineering principles is a tough and complicated task. As mentioned by Dybå et. al[11], empirical research methods are expected to enable the development of scientific knowledge about how useful different SE technologies are for different kinds of actors, performing different kinds of activities, on different kinds of systems. Over the years the empirical studies have gained more importance in the domain as evident by comparison of views of Dybå et. al[11] and Seaman[1].

The focus of the paper is to present a review of the momentum gained by empirical studies in the domain of software engineering and present it as an exciting area of research for students and researchers. As evident from the number of conferences solely dedicated to empirical studies like EASE (Evaluation and Assessment in Software Engineering), ESEM (Empirical Software Engineering and Measurement) and journals like Empirical Software Engineering by Springer, etc. We try to capture the gaining importance of empirical studies and present a study of how various datasets released in the open access by various foundations like Eclipse, Mozilla, Apache etc. actually have helped in growth and understanding of the software engineering principles and paradigms. The same set of data can be used for evaluation of various paradigms. We try to explore the literature present which helps us realize the interesting results available by performing empirical analysis considered on background of an understanding of collaboration between developers. The empirical studies pave the way for gauging the principles of software engineering and need to be conducted with more focus for synthesizing evidence and building theories.

II. WHAT ENCOMPASSES EMPIRICAL RESEARCH

Large software projects involve various developers and researchers. Though some companies have rigid policies and methods defined, it is usually flexible and in the hands of the project manager. There is no proof methodology to quantify and prove the decisions made for software project workflow in terms of the cost effectiveness and result quality of the project. This is usually where the empirical studies performed on various methodologies come to the rescue and can be used as a base for defining and forming the approaches and strategies for undertaking software projects. Thus, empirical studies focus on evaluation of new methods and help to perform low-payoff improvements.

Empirical studies, for the case of simplicity, can be considered as analogous to experimental validation of a theory but can also encompass case studies, prototyping exercises and various survey activities. In its basic essence, empirical studies aim to compare theory with the reality and give an analysis of this comparison to improve our strategies and methodologies. Roughly, we can say that usually empirical studies consist of following steps:

A. Formulating research question

The research question drives the methodology of performing the empirical studies and is an important step.

B. Acquiring necessary data

Based on the research question formulated, all the necessary data required for synthesizing and answering the question are acquired.

C. Performing Systematic Literature Review

As outlined by Kitchenham[26], systematic literature review is necessary for understanding and evaluating the current available literature.

D. Analysing data and drawing conclusions

The data acquired is analysed and evaluated. Drawing conclusions and proper interpretations is the most important step and drives the research into being used by practitioners.

The empirical studies undertaken and the conclusions drawn are used to identify inefficiencies or identify scope of improvements. As observed by Perry et. al[13], the quality of empirical research in domain of software engineering is rising. But it is important to realize that simply applying tests and algorithms to a set of data does not yield good usable results.

An important pillar of empirical research is the credibility of the study research. The conclusions drawn by the study should be valid and it is important that readers and practitioners have confidence in the conclusions. For achieving this it is important to follow defined strategies in undertaking such research, even if it involves exploring some obvious intuitions. Thus, empirical research is a very important methodology in the field of science which is increasing gaining importance in the domain of software engineering and in its essence follows the basic scientific approach which encompasses formulating hypothesis and performing series of research on it.

III. EMPIRICAL STUDIES ON COLLABORATION AMONG DEVELOPERS BASED ON AVAILABLE DATA FROM PROJECTS

There is a rich literature available which tries to analyze and understand the collaboration among developers and researchers in a project. Empirical studies which try to understand this collaboration give a direction to practitioners on how effective their collaborations are. Collaboration among developers has seen a changing paradigm by having collaborations across latitudes and longitudes. Collaborations also occur in the open source software projects and this is more diverse than any intra-company developer interaction. Having empirical analysis on such data also opens new avenues to the project managers for applying interesting results into their project domain.

Many projects have their SVN data or bug report or feature request data available which can be studied to obtain interesting results. A lot of studies are conducted to analyze the severity of bugs or the resolution times and similar studies which try to evaluate how issues are resolved. A typical software project undergoes the traditional steps of planning, designing, developing, testing, deployment. Documented data for all these steps is being increasingly available in the public domain and opens new avenues for researchers to undertake empirical research for these data.

In this paper, though we try to explore and understand empirical studies in software engineering as a growing field, we try to present developer collaboration study as an exciting field of research which is bound to be of use to the practitioners. As observed by Saha et. al[9], bug tracking systems are among the most frequently used resources for research in mining software repositories. There has been an increasing importance to formulation of dataset suitable for research as well and few tracks in notable conferences reserve tracks for researchers who compile and present such verified and valid data for analysis. Various such datasets have been compiled by various researchers which form the basis of further analysis and empirical studies [6], [10], [16]. There is a lot of literature based on analysis of Jazz [3],[23],[15],[21].

There have been interesting results obtained regarding the bug fixing in software projects by analysis of the bug report datasets. There has been study on the effect of severity on resolution time [4] which yields a positive confirmation for the hypothesis. A very interesting study on whether the classification of bugs is really valid and how much of data do we lose by eliminating out “normal” level bugs[9] tries to remove the bias from the usual strategy of eliminating normal level bugs from the empirical studies performed. Lot of studies are conducted on the long lived bugs and try to understand the reasons for these bugs to remain unresolved. Another approach of research also tries to provide solutions to various problems [7] and by performing empirical analysis tries to prove the worth and usability of the solution.

An interesting aspect which can be examined from such datasets is the developer interactions. As observed from the study of MacLean and Knutson [6], the development of a software project involves a social network among the developers. Developer interactions and its study give useful insights into which developers actually resolve the bugs and can be used for easier allotment of developers to a bug triage[8]. A study of Android bug dataset to understand whether developer interaction and involvement actually yields better results [18] gives an interesting approach to combine the studies for domain of developer interaction and bug report datasets available.

Collaboration among developers can be better studied and formulated by learning the developer interactions. This kind of research is expected to help in taking decisions for team formations in software projects. The collaboration among developers has many other factors as well like the work and technical dependencies of the developers [17],[21], time coordination amongst developers due to geographical and temporal distance [21],[25]. Thus, interesting results can be expected by combination of above two approaches namely the collaboration among developers evaluated from available data for various software projects.

IV. GENERAL METHODOLOGY

As there is no define strategy for conducting empirical studies, researchers follow methods according to requirement of the research problem. Still, by broadly studying the various empirical researches conducted we can get a hint of some useful strategies followed by various researches. Various Scientific methods are defined that helps in empirical research. Empirical methods are concerned with acquisition of empirical science. However, assumptions regarding ontology (i.e. what we believe to exist) and epistemology (i.e. how beliefs are acquired and what justifies them) is what is needed to acquire knowledge. Scientific disciplines use a no of scientific methods which constitutes a no of concepts, rules, techniques, and approaches. Two main theories of scientific method are inductive and hypothetic-deductive methods which are based on abduction.

Both qualitative and quantitative methods are used for collecting and analysing data. Quantitative methods basically collect numerical data and then analyse it using statistical methods, whereas qualitative methods draws text, images or sounds by observing, interviewing and analysing it using methods that do not rely on precise measurement to produce conclusions.

Although different steps are suggested to acquire knowledge by various approaches but mostly all empirical methods require the researcher to formulate a research question, design study and gather evidence and data, analyse and interpret it. Design elements can be placed into 4 groups which are assignment, measurement, comparison and treatment groups.

Apart from scientific methods most commonly used methods are experimentation, surveys, case studies, and action research.

Shaowei Wang et. al[24] have done a study on how developers use question and answer sites with Stackoverflow in particular. In this research they have analysed the questions using text and graph analysis, where links are decided upon the kind of questions a developer asks and answers. Text analysis is applied on normal texts but for code snippets, LDA is applied which would help us to come up with different topics that exists in questions and answers.

From the study of many such empirical studies conducted, we can formulate a general methodology followed which is described as follows

A. Problem definition and research context

Problem definition is formulated and research review is conducted to decide which problems are yet to be answered.

B. Research questions and hypothesis formulation

The problem definition is decomposed to formulate research questions. On the basis of these questions hypothesis is formulated at any of the two levels: Abstract and Concrete. The concrete hypothesis are stated in terms of the experimental design, while abstract hypothesis are formulated at a higher level using natural language.

C. Data Collection

The required data for the empirical studies can be collected via participant observation or interviewing [1].In the participant observation method, there exists an interaction between researchers and informants. This method can be understood by considering an example of study of development pattern. In this situation if participant observation method is applied it will imply that the researcher has to observe the developer and have some interactions with parameters that govern the respective environment.

Another commonly used technique is conducting an interview to obtain historical data. This method can also be used to clarify the observations. In unstructured interviews the interviewee is the source of the question as well as the answers.

A third method which is now gaining popularity is using datasets compiled by some other researchers or organizations. Since the researcher himself doesn't collect data situations may arise where he may not get certain parameters which he is interested in.

D. Coding and Experimental setup

The analysis performed on the qualitative data obtained is regarded as coding. An experimental design is formulated to test the hypothesis. Variables and attributes of interest have to be analysed for cause and effect relationship. This stage also involves the integration of the data obtained in previous stage to formulate a consistent datasets for the further stages. The operational context of the study has to be formulated and described for physical, intellectual and cultural surroundings. The system thus designed should be capable of providing reliable and accurate results for the research questions.

E. Data Analysis

The data can be analysed by statistical methods, graphical models (like dependency networks, Markov chains etc.) or data mining approaches. Some theories or propositions are available "grounded in the data" [1] and hence used "grounded theory methods". One such method is the constant comparison which helps to keep track of the discoveries made as the data is analysed. A cross case analysis is useful for perceiving the data from many different ways. The data can also be divided based on the data source it is obtained from to identify useful group based attributes.

F. Theory Formulation and Result

To aid a researcher in the theory formulation, the "weight of evidence" approach can be followed. This would help to understand the validity and importance of the propositions and hypothesis confirmations. Propositions can be made to better fit the data by using qualitative methods.

Validation analysis plays an important role and requires that the variables accurately model the hypothesis. Anomalies in the data and the negative cases need to be handled according to the design and hypothesis to obtain a reproducible result.

The most important stage in the empirical study is to formulate useful results which can be reproduced and verified by standard techniques or those mentioned by the researchers.

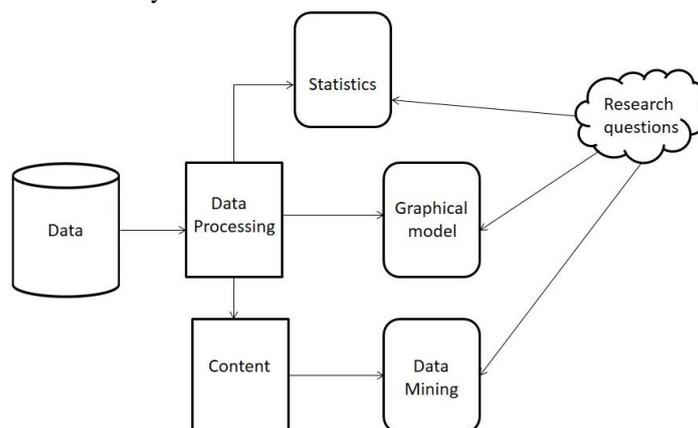


Fig. 1 Generic System Framework

V. CONCLUSIONS

Empirical Studies can be seen as a discipline for software engineering which require rigorous, verifiable and realistic result formulation. It tries to gauge the theories formulated in the real world and understand any changing patterns or updates required. We require proper, relevant and focused research synthesis and theory building for a successful and usable empirical research

VI. FUTURE SCOPE

Though empirical studies are gaining importance in the field of software engineering, we can see that a lot of such research does not formulate proper conclusions and inferences and hence does not remain directly usable. Also a lot of empirical research is non-scalable and also remains questionable regarding its validity and credibility. A defined methodology and standard if defined for empirical studies for software engineering would help the researchers and practitioners in adopting the research. Though empirical studies are being conducted in domain of software engineering for over a decade, there is still a lot of scope and need for quality research in the field.

ACKNOWLEDGMENT

The authors acknowledge the help and support of Dr. Subhajit Datta of SUTD, Singapore and Prof. Sandeep Patil of IIT, Pune. However, the views in this paper are solely those of the authors.

REFERENCES

- [1] Carolyn B. Seaman, "Qualitative Methods in Empirical Studies of Software Engineering," IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. 25, NO. 4, JULY/AUGUST 1999.
- [2] Liguoyu, Srinivas Ramaswamy, "Mining CVS Repositories to Understand Open-Source Project Developer Roles" Fourth International Workshop on Mining Software Repositories (MSR'07).
- [3] Adrian Schröter, "Predicting Build Outcome with Developer Interaction in Jazz,"
- [4] Gerardo Canfora, Michele Ceccarelli, Luigi Cerulo, Massimiliano Di Penta, "How Long does a Bug Survive? An Empirical Study," 18th Working Conference on Reverse Engineering, 2011
- [5] Yi Yang, Ping Xiang, Mike Mantor, Huiyang Zhou, "Fixing Performance Bugs: An Empirical Study of Open-Source GPGPU Programs," 41st International Conference on Parallel Processing, 2012
- [6] Alexander C. MacLean, Charles D. Knutson, "Apache Commits: Social Network Dataset," MSR 2013
- [7] Ripon K. Saha Sarfraz Khurshid Dewayne E. Perry, "An Empirical Study of Long Lived Bugs," CSMR-WCRE 2014
- [8] Motahareh Bahrami Zanjani, Huzefa Kagdi, Christian Bird, "Using Developer-Interaction Trails to Triage Change Requests," 12th Working Conference on Mining Software Repositories, 2015
- [9] Ripon K. Saha*, Julia Lawall, Sarfraz Khurshid, Dewayne E. Perry, "Are These Bugs Really "Normal"?", 12th Working Conference on Mining Software Repositories, 2015
- [10] Mayy Habayeb, Andriy Miranskyy, Syed Shariyar Murtaza, Leotis Buchanan, Ayse Bener, "The Firefox Temporal Defect Dataset," 12th Working Conference on Mining Software Repositories, 2015
- [11] Dag I. K. Sjøberg, Tore Dybå and Magne Jørgensen, "The Future of Empirical Methods in Software Engineering Research," Future of Software Engineering (FOSE'07)
- [12] Alex Borges, Waldemar Ferreira, Emanuel Barreiros, Adauto Almeida, Liliame Fonseca, Eudis Teixeira, Diogo Silva, Aline Alencar, Sergio Soares, "Support Mechanisms to Conduct Empirical Studies in Software Engineering,"

- [13] Dewayne E. Perry Adam A. Porter Lawrence G. Votta, "Empirical Studies of Software Engineering: A Roadmap,"
- [14] G. Steven McMillan and Robert D. Hamilton, "The Impact of Publicly Funded Basic Research: An Integrative Extension of Martin and Salter," IEEE TRANSACTIONS ON ENGINEERING MANAGEMENT, VOL. 50, NO. 2, MAY 2003
- [15] Subhajit Datta, Prasanth Lade "Will this be Quick? A Case Study of Bug Resolution Times across Industrial Projects," ISEC '15
- [16] Ahmed Lamkanfi, Javier Perez, Serge Demeyer, "The Eclipse and Mozilla Defect Tracking Dataset: A Genuine Dataset for Mining Bug Information," MSR 2013
- [17] Marcelo Cataldo, James D. Herbsleb, Kathleen M. Carley, "Socio-Technical Congruence: A Framework for Assessing the Impact of Technical and Work Dependencies on Software Development Productivity," ESEM'08
- [18] Subhajit Datta, Proshanta Sarkar, Subhashis Majumder, "Developer Involvement Considered Harmful? An Empirical Examination of Android Bug Resolution Times," SSE'14
- [19] Christian Bird¹, Adrian Bachmann², Eirik Aune¹, John Duffy¹ Abraham Bernstein², Vladimir Filkov¹, Premkumar Devanbu¹, "Fair and Balanced? Bias in Bug-Fix Datasets," ESEC-FSE'09
- [20] Cleidson R. B. de Souza¹, Stephen Quirk², Erik Trainer², David F. Redmiles², "Supporting Collaborative Software Development through the Visualization of Socio-Technical Dependencies," GROUP'07
- [21] Patrick Wagstrom, Shubhajit Datta, "Does Latitude Hurt while Longitude Kills? Geographical and Temporal Separation in a Large Scale Software Development Project," ISEC'14
- [22] Yuriy Tymchuk, Andrea Mocchi, Michele Lanza, "Collaboration in Open-Source Projects: Myth or Reality?," MSR'14
- [23] Subhajit Datta, Renuka Sindhgatta, Bikram Sengupta, "Talk versus Work: Characteristics of Developer Collaboration on the Jazz Platform,"
- [24] Shaowei Wang, David Lo, Lingxiao Jiang, "An Empirical Study on Developer Interactions in StackOverflow," SAC'13
- [25] Kelly Blincoe, "Timely Detection of Coordination Requirements to Support Collaboration among Software Developers," ICSE 2012
- [26] Barbara Kitchenham, and Stuart Charters, "Guidelines for performing Systematic Literature Reviews in Software Engineering", EBSE '07-001, Keele University and Durham University joint report, (2007).