2013 13th International Conference on Computational Science and Its Applications

Tools in Mining Software Repositories

K.K. Chaturvedi
Department of Computer Science
University of Delhi
Delhi, India
kkcchaturvedi@gmail.com

V.B. Singh
Delhi College of Arts & Commerce
University of Delhi
Delhi, India
vbsinghdcacdu@gmail.com

Prashast Singh
MSIT,
GGSIPU
New Delhi
prashastsingh@gmail.com

***Abstract*— Mining software repositories (MSR) is an important area of research. An international workshop on MSR has been established under the umbrella of international conference on software engineering (ICSE) in year 2004. The quality papers received and presented in the workshop has led to initiate full-fledged conference which purely focuses on issues related to mining software engineering data since 2007. This paper is the result of reviewing all the papers published in the proceedings of the conferences on Mining Software Repositories (MSR) and in other related conference/journals. We have analyzed the papers that contained experimental analysis of software projects related to data mining in software engineering. We have identified the data sets, techniques and tools used/ developed/ proposed in these papers. More than half of the papers are involved in the task accomplished by building or using the data mining tools to mine the software engineering data. It is apparent from the results obtained by analyzing these papers that MSR authors process the raw data which in general publicly available. We categorizes different tools used in MSR on the basis of newly developed, traditional data mining tools, prototype developed and scripts. We have shown the type of mining task that has been performed by using these tools along with the datasets used in these studies.**

Keywords— Software repositories; data mining tools; ICSE;

I. INTRODUCTION

Data mining is a burning topic of current day's research. Data miners are building the prediction models, pattern identification tools and techniques by mining the large repositories of data generated through the use of information technology in various domains. The accessibility of data is very limited. The source code of the software was not at all available until the birth of open source software and open access. The software development and maintenance process generates a very huge amount of data. This data varies from source code development to the bug reports data. The source code of the software is publicly available in open source software. The open source software provides source code of the software for further development and enhancement of the software. The data varies from versions to versions, change log data, web usage data, version archives, discussion forums on bug reports etc. The failure data is also maintained using different bug reporting and tracking system. The reported bugs contains many attributes such as severity, priority, components, operating system used, summary, description of the reports and status updates of the bug reports as time series. This data is very useful in conducting the research on software reliability, finding developer expertise, quality of software, resource utilization, effort, cost and time estimation, duplicate detection, dependency analysis, bug prediction, impact analysis, guiding co-change analysis, change prediction, and many more. To perform these analyses, we require the access to software repositories and analyze it which is called mining software repositories (MSR). We can use already build and matured data mining tools to perform these analytical task. Many data mining tools are available in the market but due to very high price, it is generally beyond the reach of the general researchers. Software practitioners and researchers are recognizing the benefits of mining this information to support the maintenance of software systems, improve software design/reuse, and empirically validate novel ideas and techniques. In this study, we have considered only those papers which have used the data mining tools. We have also considered the papers which are of special interest from the MSR Challenge which started in the year 2006. The MSR challenge is an opportunity to researchers to apply, compare and challenge methods and tools with a common repository. The accepted papers from MSR Challenge have also been published in the proceedings as short papers. Our main contribution of the paper is to identify the task of data mining, datasets and tools used in mining software repositories. Rest of the paper is divided into five sections. Section 2 describes the importance of mining software repositories. Section 3 provides the tools used by the researchers in MSR while section 4 describes the data sets used by them for empirical validation of the results. Section 5 specifies the results and discussions and finally paper is concluded in section 6 with a future research direction.

II. MSR AND ITS IMPORTANCE

The inception of mining software repositories (MSR) workshop started in 2004 as a workshop session of highly rated international conference on software engineering (ICSE). The popularity and research areas discusses in the workshop led to the full fledged working conference on MSR in 2007. The Mining Software Repositories (MSR) analyzes the rich data available in software repositories to uncover interesting and actionable information about software projects [1]. Software repositories such as source control systems, archived communications between project personnel, and defect tracking systems are used to manage the progress of software projects. Software engineering researchers are recognizing the benefits of mining these repositories to support the

978-0-7695-5045-9/13 $26.00 © 2013 IEEE
DOI 10.1109/ICCSA.2013.22

maintenance and evolution of software systems, improve software reusability, and empirically validation of novel ideas and techniques. Research in mining software repositories are mainly focuses on identifying and development of new tools and techniques to uncover the ways in which mining these repositories can help to understand software development and software evolution, to support predictions about software development, and to exploit this knowledge concretely in planning future development.

In the present study, we have considered the MSR conference proceedings since 2007 to 2012 along with other related papers. Full length papers from the proceedings have been downloaded and studied including the presentations for the key note talks from the MSR website. Number of papers considered under study has been shown in table I and subsequently shown in figure 1. The trends regarding the number of papers related to mining tasks are decreasing in initial year i.e., from 2007 to 2010 but it significantly increases in 2011. This shows the people start taking the interest in conducting the research in mining with software engineering resources.

TABLE I. RESEARCH PAPERS CONSIDERED UNDER STUDY

S.No.	Year	No. of papers
1	2007	33
2	2008	25
3	2009	19
4	2010	16
5	2011	31
6	2012	22

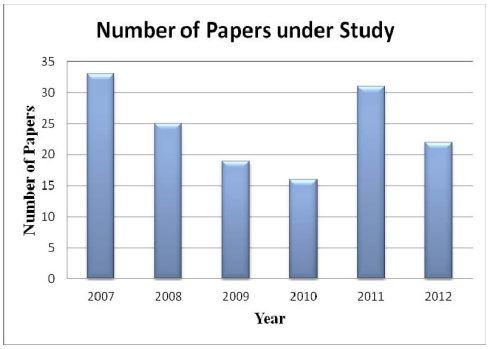

Figure 1 Trends of Number of papers considered under study

III. TOOLS USED IN MSR

Most of the researchers used already developed tools for data mining tasks in software engineering repositories. Few of the researchers developed new tools and scripts for their mining tasks. These tools are used mainly for data extraction from repositories, required data filtering, pattern finding, learning and prediction. Some researchers developed tools for creating the workflow for data retrieval to predictions while others are interested in writing their own customized scripts in various programming languages instead of using already developed tools. The basic comparisons of the data mining tools are categorized in four categories such as newly created tools, traditional data mining tools, prototype developed/implemented and scripts. In few papers, researchers have written the scripts in python, java languages which is used as plug-in for eclipse environment. The tools reported as newly developed or created are also integrated in eclipse or in some already established data mining tools. Some researchers initiated the tool development process as prototype development and its implementation as their preliminary results. There are researchers who are reluctant to mention the tools used for analysis. This analysis varies as regression, correlation, logistic regression and other exploratory analysis of the data. The tools are categorized into four classes namely, newly developed tools, traditional data mining tools, prototype development/implementation, and scripts. The categorized list of tools is shown in table II.

TABLE II. MSR TOOLS

Types	Tools
Newly Developed	Exempler[8], CloneTracker[11], sqminer[13], CallExtractor[14], J-REX on MapReduce[15], REXML(Ruby for XML)[16], Ruby Porter Stammer and Ruby Classifier[16], FPLearner[17], FPClassifier[17], EvoOnt[18], iSPARQL[18], Deep Intellisense[19], Operation Recorder[20], SpotWeb[21], PopCon[22], CP-Miner[23,24], PatchMiner[25], Binary Analysis Tool (BAT)[26], Anchored signature matching and Rationalizer[27], SeCold[59], rodrigorgs2[61], checkstyle[97], Apache Pig [98], Marmoset[99], Evolizer[100], Swanson's Maintenance Classification[101]
Traditional data mining tools	CodeFinder[9], CodeBroker[10], R Tools[28], WEKA toolkit[29], GIT and GitMining Tools[30], KEEL[31], RapidMiner[32], SAS Text Miner[33], SPSS Clementine [34], GNUPlot[35], Matlab[36], CCFinder[12], SourceMonitor[37], MALLET[39], CTAGS[],CHANGEDISTILLER and EVOLIZER suite[41], UCINET[42], UNDERSTAND[43], Unix Diff utility, CVSAnalY tool[45], FLOSSmole[47], Simian[49, 50], Statistica[107]
Prototype	Bug report assignment on eclipse[51], Automatic Labeling of Software Components in Java[52], How developers work together in Tomcat and JUnit[53], defect correction effort using the defect data collected from a Japanese multi-vendor information system development project COSE [54], which methods are cloned and which methods are changed in each transaction in DNSJava[55], Renaming recommendation system[56]
Scripts	Eclipse, FreeBSD, GNOME, Apache and Apache Ant, Java Generics adoption[62]

SourcererDB [86] is a relational database containing entity/relationship models of the projects from the Sourcerer Java Repository. This is contained a repository of 18,000 Java projects downloaded from Apache, Java.net, Google Code and Sourceforge. Spotweb [21] is a Spotnet implementation in PHP. Spotnet only shows actual spots. Spots are manually created by humans which categorize them and provide an image and description for the spot. EvoOnt [18] is a set of software ontologies and data exchange format based on OWL. It provides the means to store all elements necessary for software analyses including the software design itself as well as its release and bug-tracking information. Tool callextractor [13] is used for extracting call sequences based on the srcML

format (www.sdml.info/projects/srcml) and the tool sqminer for mining frequent patterns. Tool sqminer [13] has been used for mining the ordered patterns, variants, and violations directly from the ordered patterns extracted by callextractor. In addition the tool sqminer was configured for itemset mining to mine the unordered patterns, variants, and violations from the unordered patterns formed from the ordered patterns extracted by callextractor.

IV. DATASETS USED IN MSR

Software repositories consists of different data sets namely, source control systems, archived communications between project personnel, source code repositories, bug repositories, mailing lists, concurrent versioning system (CVS) repositories, stack traces, web usage logs and other project repositories. Generally, people used various types of data from MSR Challenge and popularly known source code repositories as Linux (linux.org) Kernel, Eclipse (eclipse.org), GNOME (gnome.org), Mozilla (Mozilla.org), FreeBSD (freebsd.org), NASA's PROMISE (promisedata.org), MySQL (mysql.com), Postgres (postgresql.org), Python (python.net), AgroUML (agrouml.tigris.org), Apache (apache.org), Wikipedia (Wikipedia.org), OpenOffice (openoffice.org), JBOSS (jboss.org), JEDIT (jedit.org) and Linux distributions. The raw datasets of various versions used by the researchers are available for download from project repositories such as sourceforge (sourceforge.net), BugZilla (www.bugzilla.org/), Promise (promisedata.googlecode.com), CVS, etc. Very few researchers provided the processed datasets for replication of the study which can be used to run and build new experiment to further find some insights in the data. The summarized list of data sets, tools and their applications are shown in table III.

V. RESULTS AND DISCUSSIONS

We have studied around 150 papers from MSR conferences and other IEEE conferences as shown in table I. These tools are used to mine useful information from software repositories to enhance the quality of software product. We have categorized the tools on the basis of newly developed, traditional data mining tools, prototype developed/implemented and scripts. The names of different tools along with their types/categories have been shown in table II. We have also presented a comprehensive list of applications with respect to tasks of data mining, tools along with their and datasets used as mentioned in different MSR related research papers in table III.

TABLE III. TOOLS AND THEIR APPLICATIONS

S.No.	Application(s)	Tools Developed/Used or task performed	Dataset(s)
Data Processing and Exploration			
1.	Finding highly relevant software projects from a large archive of executable applications Exemplar (EXEcutable exaMPLes ARchive) for finding highly relevant software projects & information retrieval and program analysis[8]	Exemplar, CodeFinder, Codebroker	Sourceforge.net
2.	MapReduce to Support Research in MSR[15]	J-REX on MapReduce	Eclipse, BIRT and Datatools
3.	Summarizes and displays historical information about source code[19]	DeepIntellisense	Source Code repository
4.	Mechanism for recording all editing operations a developer has applied to source code on an integrated development environment[20]	OperationRecorder	-
5.	Exploration of promise and perils of mining OSS[30]	Git Mining Tools	Git Repository
6.	Information and Data retrieval and storage[48]	Flossmetrics, Sourcerer, Alitheia Core project, FLOSSmole	Github and GHTorrent, MongoDB
7.	Open and collaborative platform for sharing software datasets crawling, extracting, and representing facts from different resources as Linked Data[59]	SeCold (http://secold.org/)	-
8.	Amassing and indexing a large sample of version control systems[75]	Exploration in general	Sourceforge and googlecode
9.	Collecting and pre-processing data, calculating metrics, and synthesizing composite results[76]	Alitheia Core tool	CVS
10.	Rich dataset of source code to facilitate the sharing of extracted data[86]	SourcererDB	2,852 open source Java projects
11.	Data preparation and analyzing large Software Repositories [98]	Apache Pig	Eclipse project repositories
Classification			
1.	Mining Usage Data how software systems are actually used in practice	GNU R tool	Firefox and Chrome
2.	Severity prediction of bug reports [16]	Developed in Ruby (REXML, Porter Stammer and Classifier)	Mozilla, Eclipse and GNOME components
3.	Detect fault-prone modules [17]	FPLearner and FPClassifier	agroUML project and eclipse BIRT
4.	Software, bug, version ontology model (Som, Bom, Vom)[18]	EvoOnt, Isparql	Eclipse and Bugzilla
5.	Code-search-engine-based approach that tries to detect hotspots in a given framework by mining code examples[21]	SpotWeb	JUnit and Log4j
6.	Bug has diagnosed and fixed in some other branches. These branches have diverged over the years and many code are appear to be similar but are not exactly the same.[25]	CP-Miner, PatchMiner	Source Code
7.	System for code clone detection in binaries[26]	Binary Analysis Tool (BAT)	Linux kernel
8.	To identifying library version information within a given Java application[27]	Anchored signature matching	Maven2 repository

9.	Software Defects Using Topic Models (code quality) [40]	MALLET	Mozilla Firefox, Eclipse, and Mylyn
10.	Predict Types of Code Changes(Fine-grained source code changes) [44]	EVOLIZER suite, UCINET, UNDERSTAND, CHANGEDISTILLER, Rapid Miner	Eclipse platform and Azureus3
11.	Impact on the development team for transition from a company-backed to a more community-backed project [46]	CVSAnalY tool, GNUplot	Mozilla
12.	Bug report assignment [51]	Develect (Prototype)	Eclipse
13.	Number (amount), density (amount per time unit)of co-change, change in containing clones & Tool measures which methods are cloned and which methods are changed in each transaction [55]	Prototype tool, CloneTracker (Uses CCFinder to detect clones, CTAGS to detect where methods start, and CVS commands to extract information from the source code repository)	DnsJava (implementation of a domain name system)
14.	Verification of Bug Fixes, Software product and process quality. The tool is accessible at https://sites.google.com/site/rodrigorgs2/msr2012 [61]	Rodrigorgs2 (Developed)	Eclipse and NetBeans bug repositories
15.	Comparing fine-grained source code changes and code churn for bug prediction [63]	CHANGEDISTILLER	Eclipse platform
16.	Induction of external developers as code committers and find out his trustworthiness[64]	Automated approach based on mining code repositories and bug-tracking systems	Eclipse projects
17.	Automated topic naming to support cross-project analysis [65]	"Automated labelled" topic extraction using LDA in CVS and Bit-Keeper	MySQL and MaxDB
18.	To inform developers who are presenting historical information either directly from or mined from software repositories[67]	Deep Intellisense tool, Rationalizer	-
19.	Visualization techniques to user profiles and repository metadata for (a) follower relationships, (b) successive commits, or (c) contributions to the same project [68]	geo-scatter maps, small multiple displays, and matrix diagrams	GitHub source code hosting service
20.	Predicting fault-proneness of software modules [69]	Rational XDE (UML), Rational ClearCase (VCS), Rational ClearQuest (For Bug Tracking), WEKA and SPSS	Integrated healthcare system in the Netherlands
21.	Mining Security Changes [70]	Scripts	FreeBSD
22.	Finding copy and clones of the code of the system and then applies the classification based on the version information available from the system's subversion repository [50]	Simian	GNOME Desktop suite
23.	Predict bug lifetimes [71]	WEKA toolkit	FreeBSD bug repository
24.	Predicting security bug reports [72]	SAS Text Miner	Cisco software system
25.	Build defect prediction models and visualize the prediction quality along the time axis to identify concept drifts[78]	WEKA	Eclipse, OpenOffice, Netbeans and Mozilla (CVS and Bugzilla)
26.	Author entropy is used to characterize author contributions per file[80]	Python script implemented by Taylor et al. and custom Java program to aggregate the output of each Python script execution	GNOME suite of desk-top applications
27.	Mining Search Topics from Usage Log of Code Search Engine [81]	Dragon Toolkit (LDA topic modeling feature)	Usage log of Koders, one of the major commercial code search engines
28.	History of Synchronous Changes to Refine Code ownership[82]	Syde tool in Java with Eclipse	Speed http://www.cpmbraxis.com
29.	Jazz provides huge opportunities for software mining and defect prediction[83]	Jazz	Jazz repository
30.	Automatically mine repositories and link information across repositories & Bugzilla bug tracker and Launchpad bug tracker [84]	HTMLScraper used to implement HTML downloader and HTML parser	Fedora, Ubuntu, Suse, RedHat, and Firefox
31.	Determine earlier changes which may have introduced bugs[85]	FindBugs and Pylint	Groovy, CherryPy, Python
32.	Emergent Expertise Locator[88]	EEL is implemented as a Java plug-in for Eclipse	Eclipse project, Firefox and Bugzilla
33.	Mining refactoring decision and software change history[90]	IDE plug-in for the Squeak Smalltalk environment, under the moniker "SpyWare"	Created a software change-based software repository
34.	For each class of revision, does the frequency of those revisions increase (or decrease) softChange for CVS repositories and bt2csv for BitKeeper repositories[91]	Hiraldo-Grok- an OCaml based spin off of Grok used for answering statistical based queries, R- a plotting	MySQL database

		and statistics package, GNUPlot - a graph plotting package.	
35.	To assess the personality of core OSS developers Personality type of developer new, current, departing developer[92]	WEKA	Apache httpd server
36.	Probabilistic Author-Topic model for mining developer contributions and similarities[96]	Matlab implementation of the LDA-based AT algorithm from	Eclipse 3.0 source code
37.	Classifying the maintenance commits as per their types[101]	Swanson's Maintenance Classification	Nine open source projects (Boost, Egroupware, Enlightenment, Evolution, Firebird, MySQL, PostgreSQL, Samba, Spring framework) commits repositories
38.	Severity prediction of reported bug based on bug report data[104]	RapidMiner	PITS projects, selected components of Eclipse, Mozilla and GNOME
39.	Priority prediction of reported bugs based on bug repositories[106]	RapidMiner	Eclipse Ver2 and Ver3, Open Office (Database, Spreadsheet and Power point packages)
Association			
1.	Software co-change prediction using association (Frequent pattern mining)[13]	Sqminer	KDE, Apache, jEdit, and GCC
2.	Item set and sequential-pattern mining for function-call usage patterns from source code with association[14]	Tool callextractor for extracting call sequences and sqminer for mining frequent patterns	Linux kernel v2.6.14
3.	How Developers Work Together? [53]	Prototype tool for visualizations	JUNIT and TOMCAT
4.	Studying how terms (identifier atomic components) change in source code identifiers[56]	Renaming recommendation system	Eclipse-JDT and Tomcat
5.	Mapping bug reports to the changes for fixing the bugs. These fixes identified buggy code lines Association (bug introducing changes) [57]	Git and Diff	Android
6.	How Java developers use generics by mining the changes made by prior researchers.[62]	Java Generics adoption	History of 20 popular open source Java programs
7.	Automated Dependency Resolution[74]	Sourcerer (Modified in Eclipse)	Maven2 Central Repository
8.	Rule violations to fault fixing changes[77]	Qmore and CMSynergy	Component of the NXP TV platform (referred TVC for TV component)
9.	Defect correction effort using the defect data collected from a Japanese multi-vendor information system development project using association rule mining [54]	Prototype implementation	Probe information system carried out by members of the COSE
10.	Patch submission and acceptance into the codebase[93]	Built-in tools	Apache, Python, Postgres SQL, and MySQL
Software Evolution			
1.	Evolution of the Linux kernel using code-clone analysis and the code-clone coverage metrics [12]	Used clone detection tool CCFinder	136 versions of the stable Linux kernel
2.	Faster Releases Improve Software Quality [38]	SourceMonitor	source code repository (i.e., Mercurial), crash repository (i.e., Socorro), and bug repository (i.e., Bugzilla)
3.	Analyzing samples of the committed code (evolution of data races) [58]	Race detectors with RoadRunner dynamic analysis framework	JEdit and Columba
4.	To detect more distinct topics as well as more sensitive and accurate topic evolutions[66]	Diff model	-
5.	Automatic Labeling of Software Components and their Evolution[52]	Prototype implementation	Java 6.0 API (packages, classes (interfaces, annotations, and nums), fields, methods and type parameters) and JUnit
6.	Social Interactions to Release History During Software Evolution mapping discussion archives to the source code changes[89]	Implementation as a set of scripts	LSEdit and Apache Ant
7.	Learning about the program and software evolution [99]	Marmoset	CVS repositories over

			51,502 snapshots of participating students
8.	Discusses about meta-models for evolution data, data analysis and history mining, software quality attributes, as well as visualization of analysis results [100]	Evolizer	AgroUML
9.	Supporting Architecture Evolution by Mining Software Repositories [102]	Custom Made Software (ClearCase Scripts)	Software system of Philips Healthcare MRI (Versioning system)
Others (regression analysis, topic mining, social network analysis, topic categorization, cross referencing of source code, linkages with different repositories, etc.)			
1.	Provides a bridge between high-level design documentation and low-level API documentation by statically analyzing a framework and several of its clients[22]	PopCon	Eclipse
2.	Improving Defect and Effort Prediction Models[60]	R Code (Developed)	Xalan, Lucene, CHINA and NASACoC, PROMISE2007
3.	Stability of network metrics in the presence of inadequate and missing data using social network analysis metrics[73]	Constructing information flow network	Apache, MySQL, and Perl
4.	Fault prediction models[79]	Regression (Exponential)	Business maintenance system that has had 35 releases
5.	Predict the fixing effort[87]	Lucene framework	Effort data from the JBoss project
6.	Prioritize warning categories by analyzing the software change history and Warning prioritization through bug finding tools[94]	FindBugs	FindBugs, JLint, PMD, Columba and jEdit
7.	Modelling and predicting bug lifetimes[95]	WEKA toolkit	Eclipse Bugzilla database
8.	Normative behaviour in open source software development from the data available in various software repositories [97]	Checkstyle	Checkin the coding convention of Java Code
9.	Predicting the effort aware models using three datasets by preparing a code in R tool [103]	R Tools	Module (i.e., file and package) datasets from v.3.0, v.3.1 and v.3.2 in each subproject (Platform, JDT and PDE) from the Eclipse CVS repository
10.	Bug prediction using Support Vector Regression[105]	Statistica	Layout sub-components of Mozilla projects (BASE, TABLES, and XUL)

We have studied and explored the research papers presented in MSR conferences related to data mining tools involvement. We found that major task mentioned in the papers are classification, association, data processing & data retrieval, software evolution and others as shown in figure 2. We have also found that data retrieval, pre-processing and post processing of the data are most important and mentioned in most of the papers.

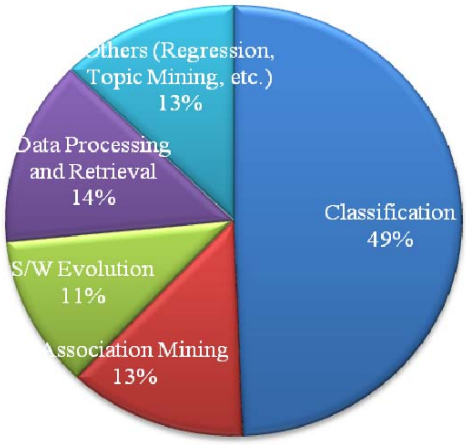

Figure 2 Mining tasks in MSR

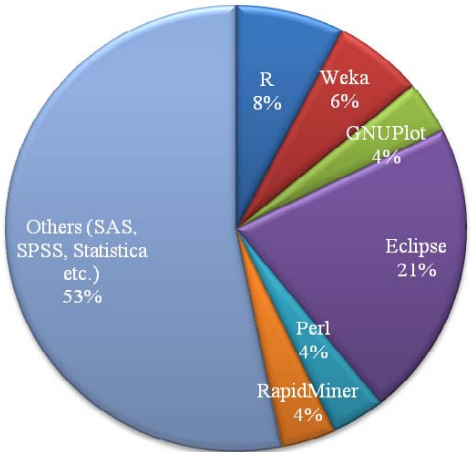

Figure 3 Mining Tools in MSR

This is a time consuming task as data is not available in the required format of the available and developed data mining tools. The data processing mainly involves with data preparation, data visualizations, data summarization, and visualization of results. The classification tasks mainly include bug prediction, detection of defect prone modules, change prediction, severity prediction of bug reports, bug report assignment etc. The association rule mining involves dependency analysis, guiding co-change analysis, function call

usage pattern finding, mining the change made by prior developers, changes towards patch submission, etc. The others category includes the task related to regression analysis, topic mining, social network analysis, topic categorization, cross referencing of source code, linkages with different repositories, etc. The techniques mainly used are Naïve Bayes, Support Vector Machine, Frequent item set mining etc.

The tools used to carry out these tasks are mainly custom developed and implement in eclipse environment, R library and custom build self made code. While some of the researchers are used already developed tools such as R, Weka, GNUPlot, RapidMiner, etc. The contribution of various tools used in MSR papers are shown in figure 3. The use of tools depend on the data mining task being performed on the extracted data and availability of techniques used to accomplish the required task available in the data mining tools. The major contribution of tools is in others category. The others category includes the tools from open as well as commercial software. The commercial tools used for mining software repositories are mainly SAS, SPSS, Statistica, Matlab and many other proprietary tools. This category also includes the native tools developed by the established research group. The second most popular category of tools is by using the Eclipse development platform. The researchers also focus on development of the tools based on their specific requirement using Eclipse platform. As we know that Eclipse is an integrated development environment (IDE) provides the integration of Java, JSP, C, C++, Perl, etc. The contribution of the open source software tools such as WEKA, R, RapidMiner are gaining popularity and proven to be promising tools in the literature as these tools provide the flexibility of development of new tools as well as plug-in. The tools mainly gaining popularity which are having the functionality of creating a workflow/knowledge flow. Workflow consists of accumulation of more than one task as a single package. In this workflow, users need to change the paths for data and techniques required for their task.

Most commonly used applications in MSR are modeling and predicting changes, detection of defect prone modules, finding bugs and their lifetimes, dependency and co-change prediction, developer expertise, predicting type of code changes, severity, assignment, effort estimation, automated task of mining repositories and link information across these repositories etc.

Recent mining software repositories conferences are focusing towards the following topics

i. Discovering relationships in various software repositories

ii. Bug prediction

iii. Detection of fault prone modules/files

iv. Co-change prediction and association with other classes/files/modules

v. Severity, priority and assignment of bug reports for fix

vi. Duplicate bug reports detection

vii. Analysis of cross referencing of bug fix and code change

viii. Software evolution and prediction of time of next release

ix. Text mining for topic mining, social network analysis, code co-occurrence, methods recommenders etc.

VI. CONCLUSIONS

The repositories related to software development and evolution helps in improving the quality of software product. The most challenging part is how to extract the useful data/information from the repositories which can help in developing the software as well as ease the software development process. In this paper, we have surveyed MSR and other related papers to understand the usage of different tools. We have studied the purpose of tools along with the dataset used. We have also classified the list of tools based on newly developed, traditional data mining, prototype development and scripts to achieve the desired tasks of data mining. We have studied different task/ activities of data mining which performed on software repositories to improve the quality, reliability, development, maintenance and evolution. We have also found that major mining task in software repositories is classification. The tools used in mining software repositories are varied based on the availability of tools and most of the people used combination of various tools for their desired task.

We have not completely studied the functions of all tools used in MSR conference papers but we have studied their applications in mining software repositories. This study will help the researchers in getting tools for their application in initiating the research on mining software repositories.

REFERENCES

[1] Mining Software Repositories. Available at http://www.msrconf.org

[2] Proceedings of 29th International Conference on Software Engineering Workshops (ICSEW'07). 2007

[3] Proceedings of the 2008 international working conference on Mining Software Repositories (MSR '08). ACM New York, NY, USA ©2008. ISBN: 978-1-60558-024-1. 2008

[4] Proceedings of 6th IEEE Working Conference on Mining Software Repositories (MSR'2009). Eds. by Michael W. Godfrey and Jim Whitehead. ISBN 978-1-4244-3493-0. 2009

[5] Proceedings of 7th IEEE Working Conference on Mining Software Repositories (MSR'2010). Eds. Edited by Jim Whitehead and Thomas Zimmermann. ISBN 978-1-4244-6803-4. 2010

[6] Proceedings of International Conference on Software Engineering (ICSE '11). Waikiki, Honolulu, HI, USA — May 21 - 28, 2011 ACM New York, NY, USA. 2011.

[7] Proceedings of 9th IEEE Working Conference on Mining Software Repositories (MSR'2012). Eds. by Michele Lanza, Massimiliano Di Penta, and Tao Xie. ISBN: 978-1-4673-1761-0. 2012.

[8] M. Grechanik, K.M. Conroy, K.A. Probst. Finding Relevant Applications for Prototyping. *Mining Software Repositories, 2007. ICSE Workshops MSR '07. Fourth International Workshop on* , vol., no., pp.12, 20-26 May 2007.

[9] S. Henninger. Supporting the construction and evolution of component repositories. In ICSE, pages 279–288, 1996.

[10] Y. Ye and G. Fischer. Supporting reuse by delivering task-relevant and personalized information. In ICSE, pages 513–523, 2002.

[11] A. Lozano, M.Wermelinger, B.Nuseibeh. Evaluating the Harmfulness of Cloning: A Change Based Experiment. *Mining Software Repositories,*

2007. ICSE Workshops MSR '07. Fourth International Workshop on , vol., no., pp.18, 20-26 May 2007.

[12] T. Kamiya, S. Kusumoto, and K. Inoue. CCFinder: a multilinguistic token-based code clone detection system for large scale source code. IEEE Trans. Softw. Eng., 28(7):654-670, 2002.

[13] H. Kagdi, S. Yusuf, J. I. Maletic. Mining Sequences of Changed-files from Version Histories. In Proceedings 3rd International Workshop on Mining Software Repositories (MSR'06) Shanghai, China, May 22-23, 2006, pp. 47-53.

[14] H. Kagdi, M. L. Collard, J. I. Maletic. Comparing Approaches to Mining Source Code for Call-Usage Patterns in Proceedings of 4th International Workshop on Mining Software Repositories (MSR'07) (Minneapolis, MN, May 19-20, 2007).

[15] W. Shang, Z.M. Jiang, B. Adams, A.E. Hassan. Mapreduce as a general framework to support research in mining software repositories (MSR). In Mining Software Repositories, 2009. MSR'09. 6th IEEE International Working Conference on (pp. 21-30). IEEE.

[16] A. Lamkanfi, S. Demeyer, E. Giger, B. Goethals. Predicting the severity of a reported bug. In Mining Software Repositories (MSR), 2010 7th IEEE Working Conference on (pp. 1-10). IEEE.

[17] O. Mizuno, S. Ikami, S. Nakaichi, T. Kikuno. Spam filter based approach for finding fault-prone software modules. In Mining Software Repositories, 2007. ICSE Workshops MSR'07. Fourth International Workshop on (pp. 4-4). IEEE. 2007.

[18] C. Kiefer, A. Bernstein, J. Tappolet. Mining software repositories with isparol and a software evolution ontology. In Proceedings of the Fourth International Workshop on Mining Software Repositories (p. 10). IEEE Computer Society. 2007.

[19] R. Holmes, A. Begel. Deep intellisense: a tool for rehydrating evaporated information. In Proceedings of the 2008 international working conference on Mining software repositories (pp. 23-26). ACM. 2008.

[20] T. Omori, K. Maruyama. A change-aware development environment by recording editing operations of source code. In Proceedings of the 2008 international working conference on Mining software repositories (pp. 31-34). ACM. 2008.

[21] S. Thummalapenta, T. Xie. SpotWeb: Detecting framework hotspots and coldspots via mining open source code on the web. In Automated Software Engineering, 2008. ASE 2008. 23rd IEEE/ACM International Conference on (pp. 327-336). IEEE. 2008

[22] R. Holmes, R.J. Walker. A newbie's guide to eclipse APIs. In Proceedings of the 2008 international working conference on Mining software repositories (pp. 149-152). ACM. 2008.

[23] Z. Li, S. Lu, S. Myagmar, Y. Zhou.. CP-Miner: Finding copy-paste and related bugs in large-scale software code. Software Engineering, IEEE Transactions on, 32(3), 176-192. 2006.

[24] Z. Li, S. Lu, S. Myagmar, Y. Zhou. CP-Miner: A tool for finding copy-paste and related bugs in operating system code. In Proceedings of the 6th conference on Symposium on Opearting Systems Design & Implementation (Vol. 6, pp. 20-20). 2004

[25] Y. Zhou. Connecting technology with real-world problems-from copy-paste detection to detecting known bugs (keynote abstract). In Proceedings of the 8th Working Conference on Mining Software Repositories (pp. 2-2). ACM. 2011.

[26] A. Hemel, K.T. Kalleberg, R. Vermaas, E. Dolstra. Finding software license violations through binary code clone detection. Proceedings of the 8th Working Conference on Mining Software Repositories, ser. MSR '11. New York, NY, USA: ACM, pp. 63–72. 2011.

[27] J. Davies, D.M. German, M.W. Godfrey, A. Hindle. Software bertillonage: finding the provenance of an entity. In Proceeding of the 8th working conference on Mining software repositories 2011.

[28] B. D. Ripley. The R project in statistical computing. MSOR Connections. The newsletter of the LTSN Maths, Stats & OR Network, 1(1), 23-25. 2011.

[29] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I.H. Witten. The WEKA Data Mining Software: An Update; SIGKDD Explorations, Volume 11, Issue 1. 2009.

[30] C. Bird, P.C. Rigby, E.T. Barr, D.J. Hamilton, D. M. German, P. Devanbu. The promises and perils of mining git. In Mining Software Repositories, 2009. MSR'09. 6th IEEE International Working Conference on(pp. 1-10). IEEE.2009.

[31] J. Alcalá, A. Fernández, J. Luengo, J. Derrac, S. García, L. Sánchez, F. Herrera. Keel data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework. Journal of Multiple-Valued Logic and Soft Computing. Vol. 17, pp. 255–287.2011.

[32] I. Mierswa, M. Wurst, R. Klinkenberg, M. Scholz, T. Euler. YALE: Rapid Prototyping for Complex Data Mining Tasks, in Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-06), 2006.

[33] SAS Institute Inc. Getting Started with SAS ® Text Miner 3.1. Cary, NC: SAS Institute Inc. 2007.

[34] http://www.spss.co.in

[35] T. Williams, C. Kelley. Gnuplot 4.5: an interactive plotting program. URL http://gnuplot.info.2011

[36] Guide, MATLAB User'S. "The mathworks." Inc., Natick, MA 5. 1998.

[37] Sourcemonitor. Available http://www.campwoodsw.com

[38] F. Khomh, T. Dhaliwal, Y. Zou, B. Adams. Do faster releases improve software quality? An empirical case study of Mozilla Firefox. In Mining Software Repositories (MSR), 2012 9th IEEE Working Conference on (pp. 179-188). IEEE.2012.

[39] A.K. McCallum. Mallet: A machine learning for language toolkit. [Online]. Available: http://mallet.cs.umass.edu. 2002.

[40] T. H. Chen, S. W. Thomas, M.Nagappan, A. E.Hassan. Explaining software defects using topic models. In Mining Software Repositories (MSR), 2012 9th IEEE Working Conference on (pp. 189-198). IEEE.2012.

[41] H. C. Gall, B. Fluri, M. Pinzger. Change analysis with evolizer and changedistiller. IEEE Software, vol. 26, no. 1,pp. 26–33, 2009.

[42] S. Borgatti, M. Everett, L. Freeman. UCINET 5.0 Version 1.00, Natick: Analytic Technologies., 1999.

[43] UNDERSTAND (http://www.scitools.com/).

[44] E. Giger, M. Pinzger, H.C. Gall. Can we predict types of code changes? An empirical analysis. In Mining Software Repositories (MSR), 2012 9th IEEE Working Conference on (pp. 217-226). IEEE. 2012.

[45] G. Robles, S. Koch, J. M. Gonz´alez-Barahona. Remote analysis and measurement of libre software systems by means of the CVSAnalY tool. In Proceedings of the 2nd ICSE Workshop on Remote Analysis and Measurement of Software Systems (RAMSS), pages 51–56, Edinburgh, Scotland, UK, 2004.

[46] J.M. Gonzalez-Barahona, G. Robles, I. Herraiz, I. Impact of the Creation of the Mozilla Foundation in the Activity of Developers. In Proceedings of the Fourth International Workshop on Mining Software Repositories (p. 28). IEEE Computer Society. 2007.

[47] J. Howison, M. Conklin, K. Crowston. Flossmole: A collaborative repository for FLOSS research data and analyses. International Journal of Information Technology and Web Engineering, vol. 1, no. 3, pp. 17–26, 2006.

[48] G. Gousios, D. Spinellis. GHTorrent: Github's data from a firehose. In Mining Software Repositories (MSR), 2012 9th IEEE Working Conference on (pp. 12-21). IEEE. 2012.

[49] http://www.redhillconsulting.com.au/products/simian

[50] J. Krinke, N. Gold, Y. Jia, D. Binkley. Cloning and copying between gnome projects. In Mining Software Repositories (MSR), 2010 7th IEEE Working Conference on (pp. 98-101). IEEE. 2010.

[51] D. Matter, A. Kuhn, O. Nierstrasz. Assigning bug reports using a vocabulary-based expertise model of developers. In Mining Software Repositories, 2009. MSR'09. 6th IEEE International Working Conference on(pp. 131-140). IEEE. 2009.

[52] A. Kuhn. Automatic labeling of software components and their evolution using log-likelihood ratio of word frequencies in source code. In Mining Software Repositories, 2009. MSR'09. 6th IEEE International Working Conference on (pp. 175-178). IEEE. 2009.

[53] P. Weissgerber, M. Pohl, M. Burch. Visual data mining in software archives to detect how developers work together. In Mining Software

Repositories, 2007. ICSE Workshops MSR'07. Fourth International Workshop on (pp. 9-9). IEEE. 2007.

[54] S.Morisaki, A. Monden, T. Matsumura, H. Tamada, K.I. Matsumoto. Defect data analysis based on extended association rule mining. In Mining Software Repositories, 2007. ICSE Workshops MSR'07. Fourth International Workshop on (pp. 3-3). IEEE. 2007.

[55] A. Lozano, M. Wermelinger, B. Nuseibeh. Evaluating the harmfulness of cloning: A change based experiment. In Proceedings of the Fourth International Workshop on Mining Software Repositories (p. 18). IEEE Computer Society. 2007.

[56] L. M. Eshkevari, V. Arnaoudova, M. Di Penta, R. Oliveto, Y. G. Guéhéneuc, G.Antoniol. An exploratory study of identifier renamings. InProceedings of the 8th Working Conference on Mining Software Repositories(pp. 33-42). ACM. 2011.

[57] M.Asaduzzaman, M.C. Bullock, C. K. Roy, K.A. Schneider. Bug introducing changes: A case study with Android. In Mining Software Repositories (MSR), 2012 9th IEEE Working Conference on (pp. 116-119). IEEE. 2012.

[58] C. Sadowski, J. Yi, S. Kim. The evolution of data races. InMining Software Repositories (MSR), 2012 9th IEEE Working Conference on(pp. 171-174). IEEE. 2012.

[59] I. Keivanloo, C. Forbes, A. Hmood, M. Erfani, C. Neal, G. Peristerakis, J. Rilling. A Linked Data platform for mining software repositories. In Mining Software Repositories (MSR), 2012 9th IEEE Working Conference on(pp. 32-35). IEEE. 2012.

[60] N. Bettenburg, M. Nagappan, A.E. Hassan. Think locally, act globally: Improving defect and effort prediction models. In Mining Software Repositories (MSR), 2012 9th IEEE Working Conference on (pp. 60-69). IEEE. 2012.

[61] R. Souza, C. Chavez. Characterizing verification of bug fixes in two open source IDEs. In Mining Software Repositories (MSR), 2012 9th IEEE Working Conference on (pp. 70-73). IEEE. 2012.

[62] M. Grechanik, C. McMillan, L. DeFerrari, M. Comi, S. Crespi, D. Poshyvanyk, C. Fu, Q. Xie, C. Ghezzi. An empirical investigation into a large-scale java open source code repository. In Proceedings of the 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement (p. 11). ACM. 2010.

[63] E. Giger, M. Pinzger, H.C. Gall. Comparing fine-grained source code changes and code churn for bug prediction. In Proceedings of the 8th Working Conference on Mining Software Repositories (pp. 83-92). ACM. 2011.

[64] V.S. Sinha, S. Mani, S. Sinha. Entering the circle of trust: developer initiation as committers in open-source projects. In Proceedings of the 8th Working Conference on Mining Software Repositories (pp. 133-142). ACM. 2011.

[65] A. Hindle, N.A. Ernst, M.W. Godfrey, J. Mylopoulos. Automated topic naming to support cross-project analysis of software maintenance activities. In Proceedings of the 8th Working Conference on Mining Software Repositories (pp. 163-172). ACM. 2011.

[66] S. W. Thomas, B. Adams, A. E. Hassan, D. Blostein. Modeling the evolution of topics in source code histories. In Proceedings of the 8th working conference on mining software repositories (pp. 173-182). ACM. 2011.

[67] A. Bradley, G. Murphy. Supporting software history exploration. In Proceedings of the 8th working conference on mining software repositories. pp. 193–202. 2011.

[68] B.Heller, E. Marschner, E. Rosenfeld, J. Heer. Visualizing collaboration and influence in the open-source software community. InProceedings of the 8th Working Conference on Mining Software Repositories(pp. 223-226). ACM. 2011.

[69] A. Nugroho, M.R. Chaudron, E. Arisholm. Assessing uml design metrics for predicting fault-prone classes in a java system. In Mining Software Repositories (MSR), 2010 7th IEEE Working Conference on (pp. 21-30). IEEE. 2010.

[70] A. Mauczka, C. Schanes, F. Fankhauser, M. Bernhart, T. Grechenig. Mining security changes in freebsd. In Mining Software Repositories (MSR), 2010 7th IEEE Working Conference on (pp. 90-93). IEEE. 2010.

[71] G. Bougie, C. Treude, D.M. German, M.A. Storey. A comparative exploration of freebsd bug lifetimes. In Mining Software Repositories (MSR), 2010 7th IEEE Working Conference on (pp. 106-109). IEEE. 2010.

[72] M. Gegick, P. Rotella, T. Xie. Identifying security bug reports via text mining: An industrial case study. In Mining Software Repositories (MSR), 2010 7th IEEE Working Conference on (pp. 11-20). IEEE. 2010.

[73] R. Nia, C. Bird, P. Devanbu, V. Filkov. Validity of network analyses in open source projects. In Mining Software Repositories (MSR), 2010 7th IEEE Working Conference on (pp. 201-209). IEEE. 2010.

[74] J. Ossher, S. Bajracharya, C. Lopes. Automated dependency resolution for open source software. In Mining Software Repositories (MSR), 2010 7th IEEE Working Conference on (pp. 130-140). IEEE. 2010.

[75] A. Mockus. Amassing and indexing a large sample of version control systems: Towards the census of public source code history. In Mining Software Repositories, 2009. MSR'09. 6th IEEE International Working Conference on (pp. 11-20). IEEE. 2009.

[76] C. Gousios, D. Spinellis. A platform for software engineering research. In Mining Software Repositories, 2009. MSR'09. 6th IEEE International Working Conference on (pp. 31-40). IEEE. 2009

[77] C. Boogerd, L. Moonen. Evaluating the relation between coding standard violations and faultswithin and across software versions. In Mining Software Repositories, 2009. MSR'09. 6th IEEE International Working Conference on (pp. 41-50). IEEE. 2009.

[78] J. Ekanayake, J. Tappolet, H.C. Gall, A. Bernstein. Tracking concept drift of software projects using defect prediction quality. In Mining Software Repositories, 2009. MSR'09. 6th IEEE International Working Conference on (pp. 51-60). IEEE. 2009.

[79] Y. Shin, R. Bell, T. Ostrand, E.Weyuker. Does calling structure information improve the accuracy of fault prediction?. In Mining Software Repositories, 2009. MSR'09. 6th IEEE International Working Conference on (pp. 61-70). IEEE. 2009.

[80] J.R. Casebolt, J.L. Krein, A.C. MacLean, C.D. Knutson, D.P. Delorey. Author entropy vs. file size in the gnome suite of applications. InMining Software Repositories, 2009. MSR'09. 6th IEEE International Working Conference on (pp. 91-94). IEEE. 2009.

[81] S. Bajracharya, C. Lopes. Mining search topics from a code search engine usage log. In Mining Software Repositories, 2009. MSR'09. 6th IEEE International Working Conference on (pp. 111-120). IEEE. 2009.

[82] L. Hattori, M. Lanza. Mining the history of synchronous changes to refine code ownership. In Mining Software Repositories, 2009. MSR'09. 6th IEEE International Working Conference on (pp. 141-150). IEEE. 2009.

[83] K. Herzig, A. Zeller. Mining the Jazz repository: Challenges and opportunities. In Mining Software Repositories, 2009. MSR'09. 6th IEEE International Working Conference on (pp. 159-162). IEEE. 2009.

[84] P. Anbalagan, M. Vouk. On mining data across software repositories. In Mining Software Repositories, 2009. MSR'09. 6th IEEE International Working Conference on (pp. 171-174). IEEE. 2009.

[85] N. Ayewah, W. Pugh. (2009, May). Learning from defect removals. In Mining Software Repositories, 2009. MSR'09. 6th IEEE International Working Conference on (pp. 179-182). IEEE. 2009.

[86] J. Ossher, S. Bajracharya, E. Linstead, P. Baldi, C. Lopes. Sourcererdb: An aggregated repository of statically analyzed and cross-linked open source java projects. In Mining Software Repositories, 2009. MSR'09. 6th IEEE International Working Conference on (pp. 183-186). IEEE. 2009.

[87] C. Weiss, R. Premraj, T. Zimmermann, A. Zeller. How long will it take to fix this bug?. In Mining Software Repositories, 2007. ICSE Workshops MSR'07. Fourth International Workshop on (pp. 1-1). IEEE. 2007.

[88] S. Minto, G.C. Murphy. Recommending emergent teams. InMining Software Repositories, 2007. ICSE Workshops MSR'07. Fourth International Workshop on (pp. 5-5). IEEE. 2007.

[89] O. Baysal, A.J. Malton. Correlating social interactions to release history during software evolution. In Mining Software Repositories, 2007. ICSE Workshops MSR'07. Fourth International Workshop on (pp. 7-7). IEEE. 2007.

[90] R. Robbes. Mining a change-based software repository. InProceedings of the Fourth International Workshop on Mining Software Repositories (p. 15). IEEE Computer Society. 2007.

[91] A. Hindle, M.W. Godfrey, R.C. Holt. Release pattern discovery via partitioning: Methodology and case study. In Mining Software Repositories, 2007. ICSE Workshops MSR'07. Fourth International Workshop on (pp. 19-19). IEEE. 2007.

[92] P.C. Rigby, A.E. Hassan. What can OSS mailing lists tell us? A preliminary psychometric text analysis of the apache developer mailing list. In Proceedings of the Fourth International Workshop on Mining Software Repositories (p. 23). IEEE Computer Society. 2007.

[93] C. Bird, A. Gourley, P. Devanbu. Detecting patch submission and acceptance in oss projects. In Proceedings of the Fourth International Workshop on Mining Software Repositories (p. 26). IEEE Computer Society. 2007.

[94] S. Kim, M.D. Ernst. Prioritizing warning categories by analyzing software history. In Mining Software Repositories, 2007. ICSE Workshops MSR'07. Fourth International Workshop on (pp. 27-27). IEEE. 2007.

[95] L.D. Panjer. Predicting eclipse bug lifetimes. In Mining Software Repositories, 2007. ICSE Workshops MSR'07. Fourth International Workshop on (pp. 29-29). IEEE. 2007.

[96] E. Linstead, P. Rigor, S. Bajracharya, C. Lopes, P. Baldi. Mining eclipse developer contributions via author-topic models. In Mining Software Repositories, 2007. ICSE Workshops MSR'07. Fourth International Workshop on (pp. 30-30). IEEE. 2007.

[97] B. T. R. Savarimuthu, H. K. Dam. Towards mining norms in open source software repositories (Working Paper). Retrieved from http://hdl.handle.net/10523/2101. 2012.

[98] W. Shang, B. Adams, A.E. Hassan. Using Pig as a data preparation language for large-scale mining software repositories studies: An experience report. Journal of Systems and Software, 85(10), 2195-2204. 2012.

[99] J. Spacco, J. Strecker, D. Hovemeyer, W. Pugh. Software repository mining with Marmoset: an automated programming project snapshot and testing system. In Mining Software Repositories, 2005. MSR'05. 2nd IEEE International Workingshop on MSR. 2005.

[100] H. C. Gall, M. Lanza. Software evolution: analysis and visualization. In Proceedings of the 28th international conference on Software engineering (pp. 1055-1056). ACM. 2006.

[101] A. Hindle, D.M. German, R. Holt. What do large commits tell us? a taxonomical study of large commits. In Proceedings of the 2008 international working conference on Mining software repositories (pp. 99-108). ACM. 2008.

[102] Vanya, A. Supporting Architecture Evolution by Mining Software Repositories. A PhD Thesis(http://hdl.handle.net/1871/32992). 2012.

[103] Y. Kamei, S. Matsumoto, A. Monden, K.I. Matsumoto, B. Adams, A.E. Hassan. Revisiting common bug prediction findings using effort-aware models. In Software Maintenance (ICSM), 2010 IEEE International Conference on (pp. 1-10). IEEE. 2010.

[104] K.K. Chaturvedi, V.B. Singh. Determining Bug Severity Using Machine Learning Techniques. In Proceedings of CSI-IEEE International Conference on Software Engineering (CONSEG-2012). IEEE Explore. pg. 378-387.

[105] V.B. Singh, K.K. Chaturvedi. Entropy Based Bug Prediction using Support Vector Regression. In Proceedings of 12th International Conference on Intelligent Systems Design and Applications during 27-29 Nov. 2012 at CUSAT, Kochi (India). ISBN: 978-1-4673-5118-8_c 2012 IEEE Explore. pg. 746-751.

[106] M. Sharma, P. Bedi, K.K. Chaturvedi, V.B. Singh. Predicting the Priority of a Reported Bug using Machine Learning Techniques and Cross Project Validation. In Proceedings of 12th International Conference on Intelligent Systems Design and Applications during 27-29 Nov. 2012 at CUSAT, Kochi (India). ISBN:978-1-4673-5118-8_c 2012 IEEE Explore. Pp. 539-545.

[107] StatSoft, Inc. (2006). STATISTICA (data analysis software system), version 7.1. www.statsoft.com.