Mining Software Engineering Repositories: Issues and Challenges

K.K. Chaturvedi
ICAR-IASRI, Pusa, New Delhi, India
kkcchaturvedi@gmail.com

Meera Shama
SSN College
University of Delhi, India
meerakaushik@gmail.com

Sujata Khatri
DDU College
University of Delhi, India
khatri.sujata@gmail.com

Kamlesh Kumar Raghuvanshi
Ramanujan College
University of Delhi, India
raghukamlesh@gmail.com

Anil Rajput
Chandra Shekhar Azad Govt P.G. Nodal College, Sehore, MP, India
dranilrajput@gmail.com

ABSTRACT
Software repositories such as source control repositories, bug repositories, archived communications, deployment logs, and code repositories are used to understand the progress of software development. Software engineering researchers are recognizing the benefits of mining these repositories to support the maintenance/ evolution to improve software reusability and empirically validate the novel ideas and techniques. Research in Mining Software Repositories mainly focuses on identification and development of new tools and techniques to understand software development and software evolution. The source code has been maintained in sourceforge.net, github.com, code.google.com and openhub.net source code repositories. The reported bugs has been reported in BugZilla, Jira, Trac, Mantis, BugTracker.Net, Gnats and Fossil bug tracking system. With the popularity of open source software, the changes in source code and reported bugs related to software development and maintenance are recorded in these software repositories. In this study, tasks and tools of mining software repositories have been identified and discussed the issues and challenges being faced by the researchers. Case study has also been presented to determine monthly commits, monthly contributors, monthly projects and contribution of lines of code changed (LOC) to help in understanding the usage of programming languages and its evolution.

Categories and Subject Descriptors
D.2.9 [**Software Engineering**]: Management – *software development, software maintenance, software metrics, software process, life cycle, software configuration management, software quality assurance.*

General Terms
Management, Measurement, Performance, Design, Reliability, Experimentation, Verification.

Keywords
Software repositories; data mining and machine learning; bug reporting tools; defect prediction; software evolution.

1. INTRODUCTION
With the evolution of open source software, the data of various phases of software development life cycle are being made available to researchers. The software development and maintenance process generates huge amount of data varying from source code development to reported bugs. The availability of source code of open source software projects provides the opportunities to the contributors or researchers for further development and enhancement in the software. Researchers are building the prediction models, pattern identification tools and techniques by applying the data mining techniques in software repositories. The data mining requires the access of the data to empirically validate the methods and techniques proposed by the researchers. The access of such data is limited in many domains.

The changes in the source code are committed in the repositories from versions to versions and available to the researchers as change log data, web usage data, version archives, bug reports and archived communications. The change log data is described as changes in the source code and maintained using concurrent versioning system. Web usage data is the historical access of the website/software/tools and provide its usage. Version archives contain the version specific data as well as source code. The process of analyzing these repositories is called mining software repositories.

The bug report consists of information related to reported bugs. The archived communication stores the communication taken place between developers during the development and maintenance of the software. The bug report data is maintained using bug reporting and tracking systems separately. The reported bugs require many attributes such as severity, priority, components, operating system used, description of the reports and status updates. This data is very useful in conducting the research on software reliability, severity and priority prediction, finding developer expertise, resource utilization, effort and time estimation, duplicate bug detection, dependency analysis of bugs and bug prediction. Mining source code repositories open the door to conduct research on guiding co-change analysis, measuring and monitoring source code quality and complexity, developers contributions and linkages by looking into commits history, identifying the team leaders and expert programmers, application of text mining to analyze commit messages with respect to bugs and changes in source code, understanding the software evolution and predicting the optimal changes required in source code without affecting the quality and complexity.

The mining tasks in software engineering are broadly categorized similar to knowledge discovery in data (KDD) as data preprocessing, classification, association, software evolution and visualization. There are many data mining tools available to perform these tasks. The data mining tools are available in the industry that can perform these mining tasks. These tools have been developed for generic purpose but not specialized for application in software repositories. Efforts have been made to compare the bug reporting and tracking tools and identify the mining tasks and tools for mining software repositories [1, 2]. The specialized tools involve very high cost of accessibility and are generally beyond the reach of researchers. There are open source initiatives to develop the open source tools for data mining such as WEKA (Waikato Environment for Knowledge Analysis, http://www.cs.waikato.ac.nz/ml/weka/), RapidMiner (http://rapidminer.com), KEEL (Knowledge Extraction based on Evolutionary Learning, http://keel.es), R project (http://r-project.org), Open Hub (http://openhub.net) and SAMOA (Software Analytics for Mobile Applications, http://samoa.inf.usi.ch).

Software practitioners and researchers are recognizing the benefits of mining software repositories to support the maintenance of software systems, improve software design/reuse, and empirically validate novel ideas and techniques. The decision making are essentially in the need of big data analytics in software engineering as well. The availability of these repositories enabled and helped us in identifying the issues and challenges faced by researchers in mining these repositories.

Earlier, the analytics is limited to statistical analysts or analysts only. But with the opening of new avenues and huge amount of data availability and tools, the term analyst has been emerged as data miners, business analysts and data scientists.

In this study, programming language usage and evolution dataset set has been analyzed to show the usage pattern of programming languages in the industry. The study also describes the issues and challenges faced by the software engineering community.

The paper is organized into five sections. Section 2 describes the software repositories and tools. Section 3 discusses the mining tasks in software repositories and section 4 emphasizes the issues and challenges faced by researchers in mining software repositories. Finally, the paper is concluded in section 5.

2. SOFTWARE REPOSITORIES AND TOOLS

2.1 Source Code Repositories and Tools

The hosting environment of source code for the software project is made available for further development and deployment using popularly known tools such as Sourceforge (http://sourceforge.net), Git (http://github.com) and Google Code (http://code.google.com).

Sourceforge (http://sourceforge.net) is an open Source community resource and extend help to open source projects. The community collaboration helps to create resources for open source software development and distribution. The developers on Sourceforge creates powerful software using over 430,000 projects; over 3.7 million registered users, connects more than 41.8 million customers with all of these open source projects and serves more than 4,800,000 downloads a day. The developers/ researchers have been using Sourceforge to develop, download, review, and publish open source software. Sourceforge is the largest, most trusted destination for Open Source Software discovery and development on the web.

GitHub (http://github.com) is the kind of repositories used to share code with friends, co-workers, classmates, and even strangers. Over eight million people use GitHub to build tools with the collaborative features of GitHub.com, our desktop apps, and GitHub Enterprise. There are 8.2 million people collaborating in nineteen million repositories on GitHub. Developers are collaborating towards the development of project from all around the world.

Google Code (http://code.google.com) provides a free collaborative development environment for open source projects. The control of the project code is with its own members using Subversion/Mercurial/Git repository, issue tracker, wiki pages, and downloads section. The project hosting service is simple, fast, reliable, and scalable, so that the manager or owner can focus on the source code development.

2.2 Bug Repositories

These repositories maintain the status of a reported bug and its update towards its resolution. There are many bug reporting and tracking systems available for use in the industry. The popular bug reporting systems are BugZilla (http://Bugzilla.org), Mantis (https://www.mantisbt.org/), Jira (https://www.atlassian.com/software/jira) and Trac (trac.edgewall.org/). These tools help in triaging the bugs, assessing the bugs, severity, priority, managing the status and many other attributes.

2.3 Data Sets

Data from source code repositories namely Linux (linux.org), Eclipse (eclipse.org), GNOME (gnome.org), Mozilla (mozilla.org), FreeBSD (freebsd.org), NASA's PROMISE (promisedata.org), MySQL (mysql.com), PostgreSQL (postgresql.org), Python (python.net), AgroUML (agrouml.tigris.org), Apache (apache.org), Wikipedia (Wikipedia.org), Open Office (openoffice.org), JBOSS (jboss.org) and JEDIT (jedit.org) have been used to empirically validate the proposed methods and techniques by various researchers. The repositories and analytical platform also provides the access to these data. The data from open source projects will also provide the historical changes in source code and reported bugs. In addition to the above, the software repositories have been made available in sourceforge, Git and Google Code.

2.4 Tools

The tool callextractor is used for extracting call sequences based on the srcML format (www.sdml.info/projects/srcml) and the tool sqminer for mining frequent patterns [3]. The sqminer has been used for mining the ordered patterns, variants, and violations directly from the ordered patterns that have been extracted by callextractor [3]. EvoOnt is a set of software ontologies and data exchange format based on web ontology language (OWL) [4]. It provides means to store all elements necessary for software analysis including the software design and bug-tracking information. Spotweb is a Spotnet implementation in PHP [5]. SourcererDB is a relational database containing entity/relationship models of the projects from the Sourcerer Java repository [6]. This repository contains 18,000 Java projects developed and available with Apache, Java.net, Google Code and Sourceforge. Open Hub (http://openhub.net) is an online community and public directory of free and open source software (FOSS), offering analytics and search services for discovering, evaluating, tracking, and comparing open source code and projects. This is formerly known as ohloh.net. Open Hub code search is free code search engine indexing over 21 billion lines of open source code from projects mapped into this.

Input, technique and goal of software engineering have been summarized in figure 1.

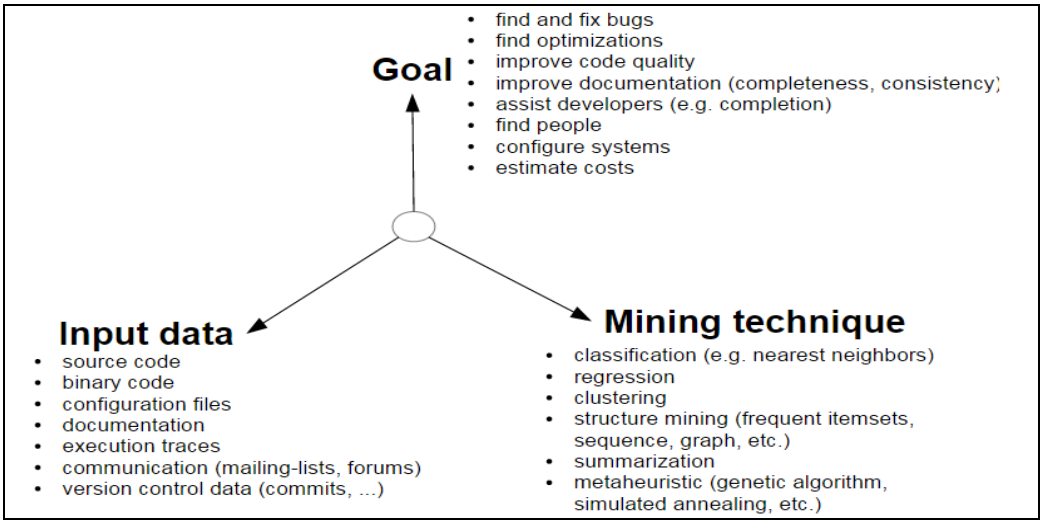

Figure 1. Input, technique and goal in software engineering [Source: http://www.monperrus.net/martin/data-mining-software-engineering]

3. MINING TASKS IN SOFTWARE REPOSITORIES

The mining tasks in software engineering are categorized as data pre-processing, classification, association, software evolution and visualization. The data processing mainly involves with data preparation, data visualizations, data summarization and visualization of results. The classification tasks include bug prediction, detection of defect prone modules, change prediction, severity & priority prediction of bug reports and bug report assignment. The clustering task mentions identification of developers teams, making the cluster of similar bugs, identify the code similarity and identification of duplicate efforts. The association rule mining involves dependency analysis, guiding co-change analysis, function call usage pattern finding, mining the changes made by prior developers and changes towards patch submission. The software evolution concerned with changes in a software system over versions or releases of the same system. The other categories include the tasks related to regression analysis, topic mining, social network analysis, topic categorization, cross referencing of source code and linkages with different repositories helps in understanding software evolution.

The summarized list of the projects from open hub analytical platform is shown in figure 2. The Subversion and Git are being mostly used source code control repositories.

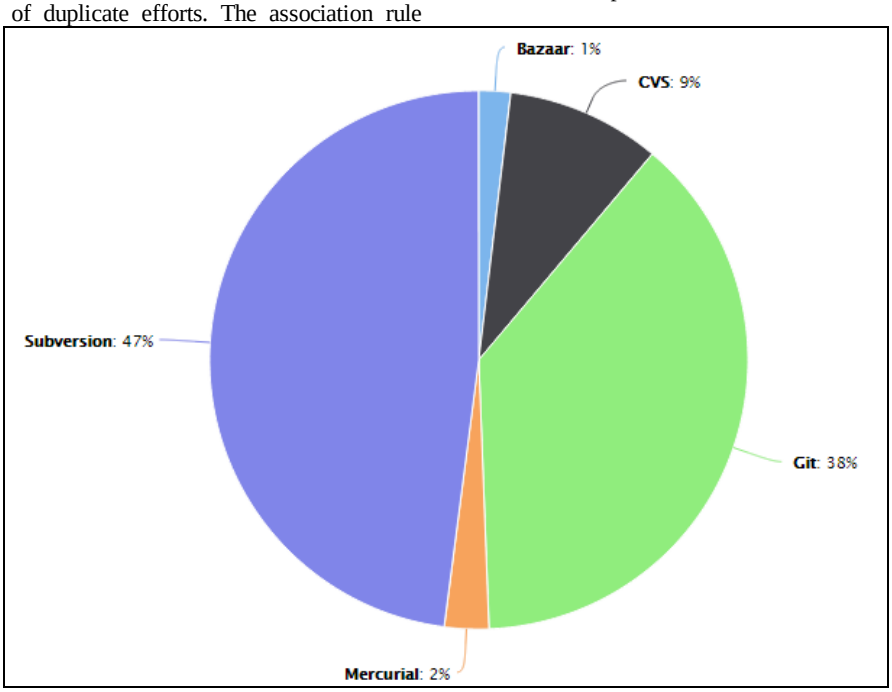

Figure 2. The source code repositories on openhub.net

The open hub provides the platform for analyzing the software repositories. The researchers can extract and draw patterns using various graphic utility such as monthly commits, monthly contributors, monthly projects and contribution of lines of code (LOC). The summarized results can be downloaded from the openhub.net. The statistics have been downloaded from openhub.net on 25th May 2015 and briefly summarized in table 1.

Table 1. Basic statistics with respect to programming language and its usage

Language	Total Lines	Code	Comments	Projects	Contributors	Commits	Earliest Usage tracked
C	8,432,106,251	5,931,462,311	1,381,047,265	74,419	173,367	42,687,448	Apr-95
C#	871,129,246	600,865,100	164,491,015	45,165	66,662	3,161,025	Oct-96
C++	3,039,601,703	2,018,161,734	570,078,161	60,350	151,477	21,352,934	Apr-95
HTML	2,050,551,726	1,717,884,174	84,768,589	176,802	276,886	13,585,608	Apr-95
Java	3,040,822,151	1,808,470,876	831,326,370	80,323	171,797	16,936,350	Jan-96
Perl	270,926,610	164,496,557	58,200,204	32,784	57,755	3,278,671	Apr-95
PHP	1,774,129,764	1,081,180,458	494,428,681	51,527	102,171	8,026,701	Nov-97

From the table 1, it is clear that the projects are currently developed using HTML environment and Java programming language. The tools are developed in web based platform which is one of the reason to extensive use of HTML and Java. In terms of number of projects, the popularity of HTML, C, Java and C++ programming languages are more used as compared to other programming environments in open

source projects. Similar trends have been observed in terms of number of contributors in open source projects with respect to programming language. The number of commits in HTML projects is less as compared to projects commits in C, C++ and Java projects. The code is significantly less in Perl projects because Perl is a modular programming language and it is also not a typed language. After looking into the trends of commits (figure 3), contributors (figure 4), contributions of projects (figure 5) and monthly changed LOC (figure 6) with respect to programming languages, the industry is focusing towards the use of HTML, PHP, JAVA and C++ instead of traditional C programming language in development of applications and software.

The trend is declining for use of C programming language over the last few years and increasing for other languages as clearly shown in figure 3. The contributors for HTML are significantly large as compared to other projects as shown in figure 4. There are many free lancers are working in web enabled programs and it is also easy to use. HTML is a basic for developing a static website. Similar trend has been observed in terms of number of projects as depicted in figure 5.

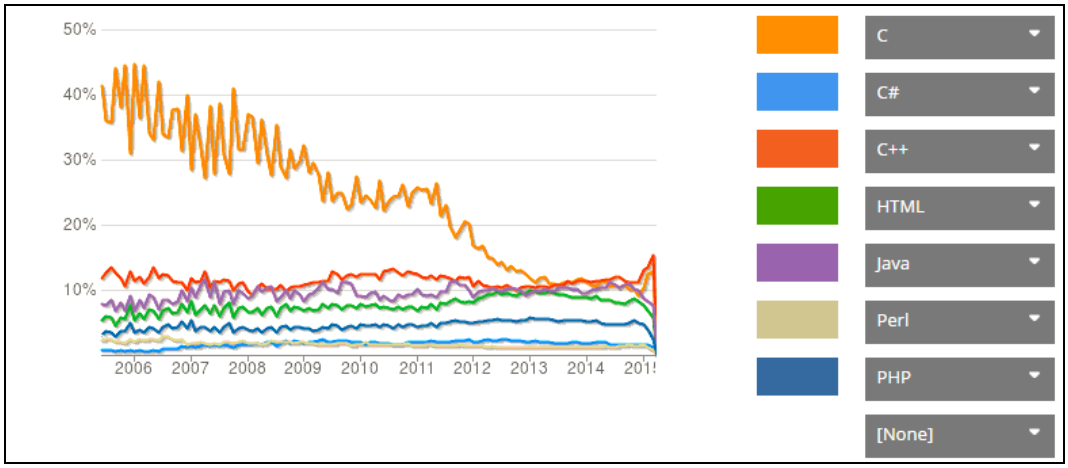

Figure 3. Monthly Commits (Percent of Total) [Source: openhub.net]

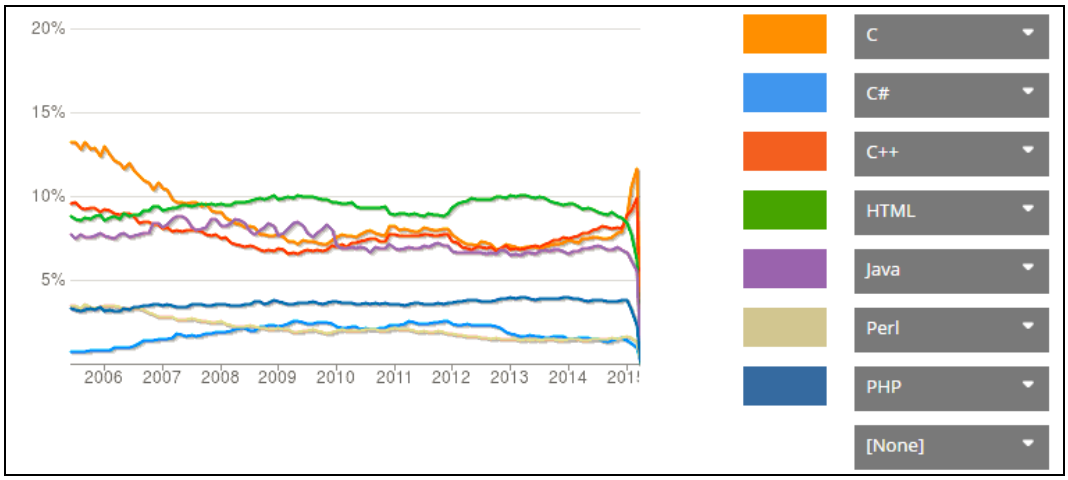

Figure 4. Monthly Contributors (Percent of Total) [Source: openhub.net]

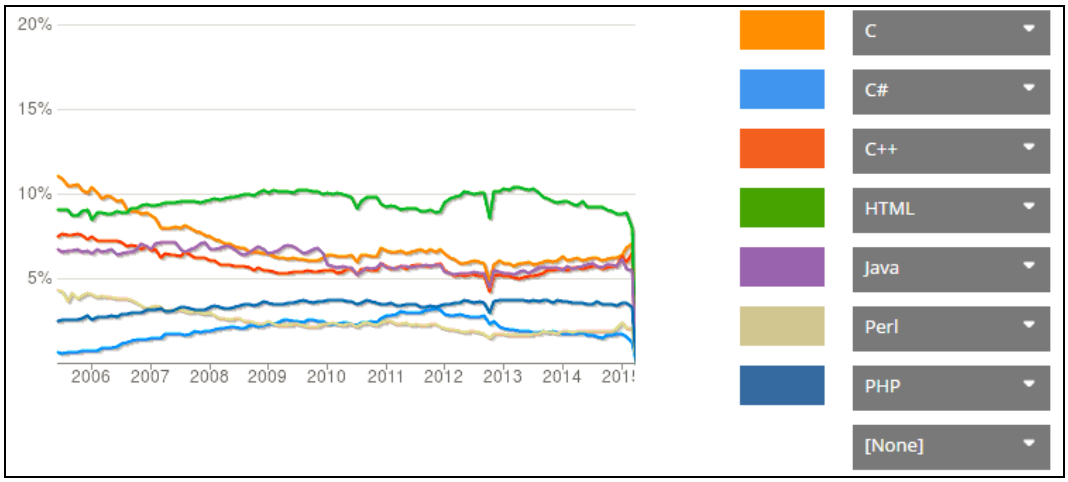

Figure 5. Monthly Projects (Percent of Total) [Source: openhub.net]

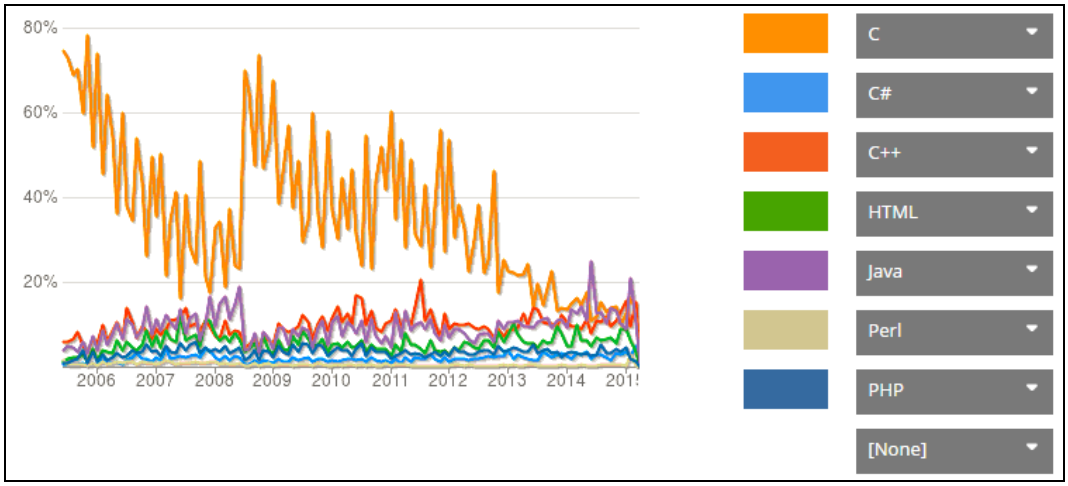

Figure 6. Monthly Lines of Code Changed (Percent of Total) [Source: openhub.net]

Monthly lines of code changed of C projects have been committed during 2009 to 2012 as mentioned in figure 6. The lines of code changed in Java, C++ and HTML are significantly increasing over the time since its tracking starts. There are very less number of lines have been changed in Perl projects because Perl is a modular language. The code base of the C project is very large as compared to other projects. App store mining has been introduced in 2012 with the focus towards the source code analytics, reusability and other historical data. To analyze these apps (applications for mobile platform), a web based software analytics platform has also been developed and called Software Analytics for Mobile Applications (SAMOA). The platform has been tested using public catalogue of FOSS (Free and Open Source Software) apps for the Android platform, called F-Droid (http://f-droid.org). SAMOA helps in mining software repositories of apps and uses visualizations for data presentation in graphical formats. This web based analytics provide the interactive platform to analyze the apps (fig. 7).

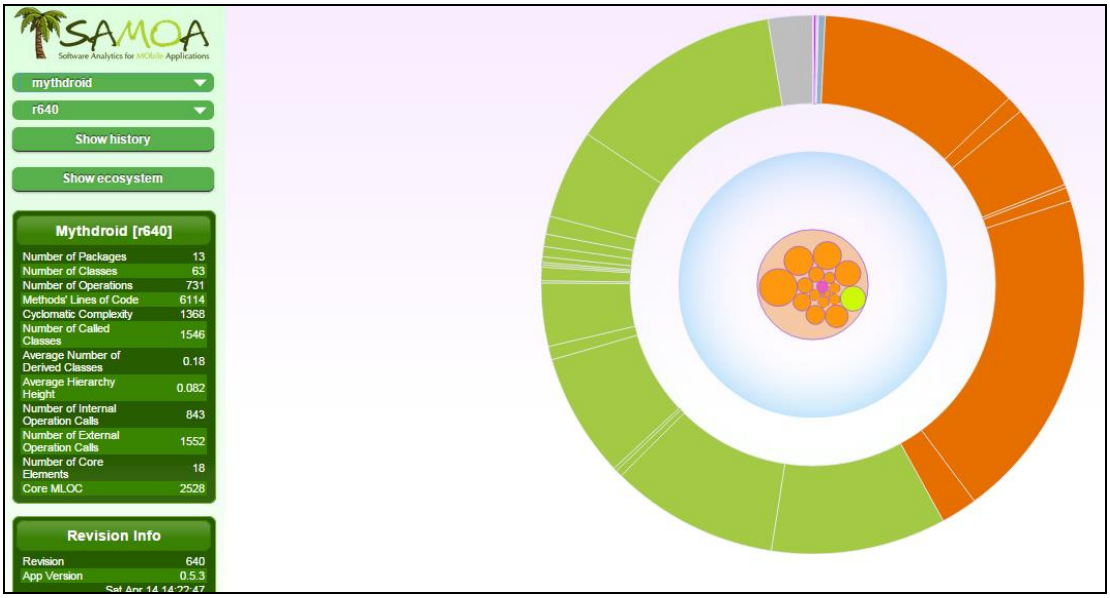

Figure 7. Visualization of apps using SAMOA

4. ISSUES AND CHALLENGES

There are many issues and challenges faced by researchers in accessing and analyzing these repositories. The study of code changes pattern by using these repositories [7] to study the popularity of programming language, identification of code cloning [8] and prediction of co-change within/across projects [9], priority prediction[10], severity prediction[11], complexity of code change based bug prediction[12, 13, 14], software complexity, quality, reliability and software evolution.

The changes in one project can propagate in many other projects and can impact on number of developers. Currently, there is no way for the developers of a library to find the affected projects due to particular changes made by a specific contributor in the selected project. This presents an opportunity to assist the developer in keeping track of all dependencies between different projects which will further open a new challenging area of research to integrate the migration of code changes across these projects.

Reusability and cloning of code helps to study the occurrence of code cloning as the open hub repository. The open hub repository contains the project wise commits, contributions and changes. Current code cloning research is focused on cloning management and awareness, such as by linking cloned source code snippets to the original site and informing both ends in case of a change. This opens exciting opportunities to track software clones and try to help developers to cope with it.

To cope up with the introduction of new programming languages and advancement of new features in the existing programming languages are being announced to increase the productivity, to survive and remain active in the industry. An evidence-based analysis of the various claims regarding these programming languages allows understanding whether there is any significant difference in how people use structured and object-oriented design in static and dynamic programming/languages. This can be studied by comparing the similar projects and its programming features with reference to their strong code base, commits and contributors.

Potential and relevant research directions in software engineering are revolving around run-time monitoring and adaptation of software systems for real-time planning and decision making, software quality assurance and diagnosis by developing software analytics to further extract the hidden fact and identify the paradigm shift towards smart digital device based apps instead of software. There are numerous issues and challenges involves in developing these apps wither from scratch or utilize the existing code with minor changes. Few of them are

- Availability of centralized repositories and extraction of desired information.
- Interlinking of software project repositories to discover the code originality or identification of duplicate code.
- Standardization of analytical tools for mining software repositories.
- Paradigm shift from software system to mobile apps (Evolution of software) in terms of platform and languages.
- High accessibility and availability in case of apps development.
- Graphical visualizations of the information.
- Ensuring quality data availability in originality of apps/software.
- Accuracy in defect predictions for quality perspective before and after testing/deployment of the software.
- Real time monitoring and decision making towards the improvement and corresponding code changes.
- Understanding the team structure and its collaborations.
- The development of parallel algorithm and partitioning of data with the evolution of multi-core and multi-processor based hardware architecture.
- Applicability of open source research to evaluate non open source projects.
- Helping the planners in effective decision making towards effective usage of available source code.
- Understanding the need of the developers/practitioners.

5. CONCLUSION

Currently, the indexing of 6,69,127 open source projects are available in open hub analytical platform by connecting 38,03,027 open source contributors and tracking 6,80,127 source control repositories containing 31,15,83,35,990 lines of code as per availability on 25th May 2015. These repositories and analytical platform enable researchers to look into the possibility of mining various aspects of software engineering. The availability of open source tools and repositories help researchers in conducting the empirical research and motivate the developers to develop required software tools and apps in collaborative manner. The industry is focusing towards the use of web/mobile based application development, new programming paradigms in place of

traditional programming languages. Increased data availability and variability in data format opens up new avenues for research in software engineering using cloud computing and Big Data Analytics. The evolvement of software will generate huge amount of diverse data that requires the robust model development to handle such explosion of data.

6. ACKNOWLEDGMENTS

Our sincere thanks are due to ICAR-IASRI, New Delhi and University of Delhi to carry out this study.

7. REFERENCES

[1] Chaturvedi, K.K., Singh, V.B. and Singh, P. 2013. Tools in Mining Software Engineering. In *13th International Conference on Computational Science and Its Applications (ICCSA)*, 2013, pp. 89-98.

[2] Singh, V.B. and Chaturvedi, K.K. 2012. Bug tracking and reliability assessment system (BTRAS). *International Journal of Software Engineering and Its Applications*, 5(4), pp. 1-14.

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

[4] Kiefer, C., Bernstein, A. and Tappolet, J. 2007. Mining software repositories with iSPAROL and a software evolution ontology. In *proceedings of the 4th International Workshop on Mining Software Repositories*, 2007, pp. 10.

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

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

[7] Ying, A.T.T., Murphy, G.C., Ng, R. and Chu-Carroll, M.C. 2004. Predicting source code changes by mining change history. *IEEE Transactions on Software Engineering*, 30(9), pp.574–586.

[8] Roy, C.K., Zibran, M.F. and Koschke, R. 2014. The vision of software clone management: Past, present, and future. In *Proceedings of European Conference on Software Maintenance and Reengineering and Working Conference on Reverse Engineering (CSMRWCRE)*, 2014, pp. 18-33.

[9] Zimmermann, T., Zeller, A., Weißgerber, P., and Diehl, S. 2005. Mining version histories to guide software changes. *IEEE Transactions on Software Engineering*, 31(6), pp. 429–445.

[10] Sharma, M., Bedi, P., Chaturvedi, K.K. and Singh, V.B. 2012. 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 27-29th Nov. 2012 at CUSAT, Kochi (India). ISBN: 978-1-4673-5118-8_c 2012 IEEE Explore. pp. 539-545.

[11] Chaturvedi, K.K. and Singh, V.B. 2012. An empirical comparison of machine learning techniques in predicting the bug severity of open and closed source projects. *International Journal of Open Source Software and Processes*, **4(2)**, pp. 32-59.

[12] Chaturvedi, K.K. and Singh, V.B. 2013. Bug prediction using entropy based measures. *International Journal of Knowledge Engineering and Data Mining*, 2013, **2(4), pp.** 266-291.

[13] Chaturvedi, K.K., Kapur, P.K., Anand, S. and Singh, V.B. 2014. Predicting the complexity of code changes using entropy based measures. *International Journal of System Assurance Engineering and Management*, **5(2)**, pp. 155-164.

[14] Singh, V.B., Chaturvedi, K.K., Khatri, S.K. and Kumar, V. 2015. Bug prediction modeling using complexity of code changes. *International Journal of System Assurance Engineering and Management*, 6(1), pp-44-60.