International Journal of Computer Sciences and Engineering Open Access
Survey Paper Volume-4, Issue-3 E-ISSN: 2347-2693

Data Mining Techniques used in Software Engineering: A Survey

Nidhin Thomas[1], Atharva Joshi[2*], Rishikesh Misal[3] and Dr Manjula R[4]

[1,2*,3,4,]*Department of Computer Science and Engineering,*
VIT University, India

www.ijcseonline.org

Received: Feb/16/2016 Revised: Feb/26/2016 Accepted: Mar/19/2016 Published: Mar/31/ 2016

***Abstract*—** A typical software development process has several stages; each with its own significance and dependency on the other. Each stage is often complex and generates a wide variety of data. Using data mining techniques, we can uncover hidden patterns from this data, measure the impact of each stage on the other and gather useful information to improve the software development process. The insights gained from the extracted knowledge patterns can help software engineers to predict, plan and comprehend the various intricacies of the project, allowing them to optimize future software development activities. As every stage in the development process entails a certain outcome or goal, it becomes crucial to select the best data mining techniques to achieve these goals efficiently. In this paper, we survey the available data mining techniques and propose the most appropriate techniques for each stage of the development process. We also discuss how data mining improves the software development process in terms of time, cost, resources, reliability and maintainability.

***Keywords*—** Data Mining, Software Engineering, KDD methods, Software Development, Frequent Pattern Mining, Text Mining, Classification, Clustering

I. INTRODUCTION

In Computer Science, software engineering deals with the design and creation of programs for computers and other electronic devices. [1] In a typical software development methodology, the work is split into distinct stages with specific activities in each, with the intention of improving planning and management. [2]

The most commonly used methodologies include waterfall, prototyping, iterative and incremental development, spiral development, rapid application development, extreme programming and various types of agile methodology. While a life-cycle "model" is a more general term for a category of methodologies, a software development "process" is often synonymous to a specific process chosen by a specific organization. A variety of such frameworks have evolved over the years, each with its own recognized strengths and weaknesses [3]. One software development methodology framework is not necessarily suitable for use by all projects. Each of the available methodology frameworks are best suited to specific kinds of projects, based on various technical, organizational, project and team considerations. [2]

Such contrasting development paradigms and the intricate dependencies that they create increase the complexity of software systems. This slows down development and maintenance activities, causes faults and defects and eventually leads to an increase in cost of the software. Organizations often fail to understand how their process impacts the quality of the software that they produce. This is mainly due to the difficulty innate in discovery and measurement. Although software metrics have long been the de-facto standard for the assessment of software quality and development processes, their drawbacks are numerous. The over-reliance on metrics that can be easily obtained and understood, usage of metrics that seem interesting but remain irrelevant and uninformative and the difficulty in obtaining truly valuable metrics are but to name a few [4].

Data mining is defined as the process of discovering previously unknown and potentially useful information from data collections. Thus utilizing data mining in software engineering with the aim of software improvement has piqued the interest of researchers worldwide. There are several challenges that emerge in mining software repositories [5]. The major ones being, dealing with the inherent complexity and sheer volume of the software engineering data.

In this survey, we present an overview of data mining techniques and how they can be applied in the context of software engineering. More specifically, we categorize these techniques with respect to the software development stages that they assist in the most.

II. RELATED WORK

The advent of cutting edge technology has made it possible to develop software that are highly complex and versatile, especially with respect to the type of data that they deal with. With such increase in complexity, it is inevitable that such software end up dealing with a large number of problems.

Thayer et al. [6], in their earlier work laid down the major problems encountered in software engineering project

management. They highlighted problems related to planning, organizing, staffing and controlling as the major challenges in this area.

Ramamoorthy et al. [7] suggested that software developers would find it increasingly hard to deal with the increase in complexity of software which would result in poor quality software and higher maintenance costs. While Thayer stated problems that were related to software project management, Ramamoorthy introduced problems that were tied to the limitations of human beings.

Clarke [8], stated that the maintainability of software becomes difficult with the increase in its complexity. This might eventually result in problems regarding software integrity and bug detection. A bug is a flaw in a computer program that can ultimately cause glitches, program failure or software destruction.

Mohammed and Govardhan [3] compared the pros and cons of five commonly used development models. This provided the base of our study in which we shortlisted the common stages in these five models.

The earliest work on the use of data mining in software engineering is the state-of-the-art report by Data and Analysis Center for Software (DACS), published in 1999 [12]. It describes the various data mining techniques thoroughly and provides an extensive list of data mining products.

A highly detailed online bibliography on mining software engineering data has been maintained by Xie [13]. He has presented numerous tutorials at several international conferences on the same subject.

In 2004, The Mining Software Repositories (MSR) Workshop was established, with the goal of increasing the understanding of software development practices through data mining. Papers published in MSR focus on topics such as assessment of mining quality, models and meta-models, exchange formats, replicability and reusability, data integration and visualization techniques.

Kagdi et al. [14] have recently published a detailed taxonomy of software evolution data mining methodologies and identifies a number of related research issues that require further investigation.

Taylor et al. [4] have produced a comprehensive survey covering the most recent applications of data mining to software engineering. They also discuss the issues one might encounter in mining software data and the necessary conditions for success. Lovedeep et al. [24] have also described various ways in which data mining can be used to improve the software engineering process.

Halkidi et al. [20] have written one of the most thorough papers on the subject. Their work features an in depth look at the data mining techniques and how they can be effectively applied in software engineering.

Aouf et al. [10] described how clustering techniques could be used to discover hidden patterns in data to gather valuable information. Chang and Chu [15] showed how Association rule mining could be used to detect software defects. Gegick et al. [25] demonstrated the importance and usage of text mining in bug identification whereas Runeson et al. [26] showcased the capabilities of NLP in tackling duplicate defect reports.

Islam and Brankovic [9] proposed techniques to ensure privacy in data mining. This was mainly done by filling parts of the dataset with noisy data. On the other hand, Ma and Chan [11] in their work suggested iterative mining for mining overlapping patterns in noisy data. While, Islam and Brankovic [9] were concerned with preserving privacy with the help of noisy data, Ma and Chan [11] dealt with the elimination of noisy data to achieve the objective of extracting valuable information.

III. DATA MINING TECHNIQUES

A. Frequent Pattern Mining and Association Rules

Association rules are used to reveal interesting relations between variables in large datasets. These relations are presented in the following form: A→B, where A and B are variables in the given dataset. Thus they can be used to discover patterns that cause defects of a severe nature [15]. The discovery of such patterns can aid several decision making processes such as cross marketing, catalog design and loose-leader analysis [16]. Algorithms such as a-priory, FP-growth and OPUS search are widely used in the software engineering context for this purpose.

B. Classification: (Supervised Learning)

One of the most widely used data mining techniques, classification find applications in statistics, pattern recognition and machine learning [17, 18]. Classification is the process of constructing a model using which future data items can be grouped into a set of pre-defined classes [19]. On the contrary, the clustering process does not rely on predefined classes or examples [13]. Classification algorithms such as decision trees, Bayesian classifiers and K-Nearest Neighbor can be used to carry out the said task effectively. In case of numerical data, regression models such as linear regression, non-linear regression and logistic regression can be used.

C. Clustering: (Unsupervised Learning)

As opposed to classification, clustering does not rely on predefined classes. Hence it is referred to as an unsupervised learning process [21]. It partitions a given data set into groups or clusters such that the intra cluster similarity is maximized and inter cluster similarity is minimized [22, 23] . Entities to be clustered need to be identified and attributed need to be selected before applying the clustering algorithm. Clustering algorithms such as graph theoretical algorithms, construction algorithms, optimization algorithms and hierarchical algorithms can be used for software engineering data. Clustering high dimensional data is relatively difficult. To deal with this scenario, highly specialized algorithms like CLIQUE can be used [24].

D. Text Mining

Text mining refers to the process of deriving information from textual data. Around 80% of all software engineering data is in text format [25]. Thus text mining can be effectively used to trace requirements, identify and predict software failures and code duplication. It can be combined with natural language processing to prevent the duplication of bug reports [26]. Text mining usually involves the transformation of raw textual data into a structured representation. It is imperative that the raw data undergo a preprocessing stage before the application of text mining techniques.

IV. SOFTWARE ENGINEERING DATA SOURCES AVAILABLE FOR MINING

This section describes the various software engineering data that can be utilized for data mining and analysis. The type of data generated by software engineering determines the choice of data mining techniques that can be applied on it to infer valuable knowledge. The following are the most common sources of software engineering data:

A. Documentation

Although software documentation data is of high importance, its complexity is relatively high as well. This includes application, system administration and source code documentation which mainly consists of free text in natural language. Among these text data, software description, start up and usage configuration, user guide, file management issues, logging, license and compatibility issues can be considered of great value for use in data mining [20] . An analytical reference of all possible types of software documentation data can be found in [27]. Software documentation might also contain multimedia data in the form of figures and audio/video instructions. Multimedia mining techniques can be used to mine such data efficiently.

B. Software Configuration Management Data

Data generated by software configuration management systems (SCMs) include software code, documents, design models, status accounting, defect tracking as well as revision control data [20]. The evolution of SCMs is discussed in [28] .Revision control software is utilized by software development organizations to manage the ongoing development of digital assets that may be worked on by a team of people. Such systems maintain a historical record of each revision and allow users to access and revert to previous versions. Thus we can analyze the historical data generated during software development. This includes details such as number of common software metrics, number of lines that have been written and authors who have written particular lines [29].

C. Source Code

Source code can be utilized by data mining applications to aid software maintenance, program comprehension and software components' analysis. The source code should initially be parsed. Once parsed, it becomes structured text. Central aspects of applying data mining techniques in source code among others include prediction of future changes through mining change history, predicting change propagation, faults from cached history, as well as predicting defect densities in source code files [20].

D. Issue tracking and bug databases

An issue-tracking database, usually consists of three types of information:

1. Structured data (database tuples) containing the description of an issue
2. The reporter's details
3. Date/time

Most software development organizations use a system for tracking software defects. Bug tracking software links bugs with meta-information (status, assignee, comments, dates and milestones, etc.) that can be mined to discover patterns in software development processes, including the time-to-fix, defect-prone components, problematic authors, etc. Some bug trackers are able to correlate defects with source code in a revision system [29]. Techniques that involve machine learning have been used to predict correct assignments of developers to bugs, cleaning the database from manifestations of the same error, or even predicting software modules that are affected at the same time from reported bugs [20].

E. Mailing Lists

Mailing lists are often offered by large software systems as a means of connecting users and developers in a collaborative environment. Mailing lists primarily contain a lot of free text. It is relatively easy to pull out message and author graphs from the data, but content analysis may be significantly harder since messages involving replies would require one to consider initial and/or previous discussions in

the mailing lists [20].The major data mining applications in mailing lists are text analysis, text clustering of subjects discussed, and linguistic analysis of messages to highlight the developer's personalities and profiles. Text mining techniques can be applied to archives of such communication to gain insight into development processes, bugs and design decisions.

V. STAGES OF SOFTWARE DEVELOPMENT

In this paper, we have evaluated the following development models, [3] short listed the stages that are common in them and demonstrated how data mining techniques can be applied to these.

Serial Number	Software Development Model
1	Waterfall Model
2	Iterative Model
3	V-Shaped Model
4	Spiral Model
5	Extreme Model

Table 1: The Software Development Models that have been considered for this paper

The following are the most common stages encountered in the development methodologies mentioned above in Table 1.

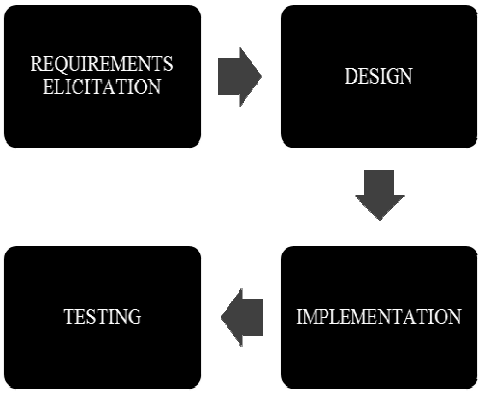

Figure 1: Common stages in Software Development

VI. ROLE OF DATA MINING IN IMPROVING THE DEVELOPMENT PROCESS

We have mentioned below the ways in which data mining helps each stage of software development in terms of time, cost and resources.

In the Requirements Elicitation phase, the requirement document provides a full description of all the software and hardware requirements for the projects. Since this document is highly detailed and descriptive in nature, it increases the time required to summarize the requirements in such a way that they are available at the appropriate time. Time management in resource availability is critical for the functioning of all the subsequent phases. Data mining techniques such as classification could be used on such data, which will classify and prioritize the requirements in such a way that all the resources which are required at each stage shall be present on time. Text mining can be used to summarize the huge amount of given data. This will decrease the amount man hours put into summarizing and prioritizing the requirements, thereby saving time, cost and human resources.

In the design phase, while designing the layout of the architecture and planning out the database structure it becomes critical to know which data would be required where and when. Data mining techniques such as Clustering can gather similar data from time to time so that extraction of data will become easier. Data gathering becomes a tedious job especially when it has to be pre-processed over and over again. By using Clustering on data elements, the data can be differentiated based on its similarity or dissimilarity. Labelling data from any incoming site would also be much easier using clustering.

During implementation, independent parts of codes or modules are implemented first, after which they are integrated with each other. This integration phase can prove to be more challenging than actually coding these modules. The functionalities of each module has to be understood so that they can be integrated efficiently. Data mining techniques such as classification and text mining will allow the developer to understand the possible bugs that might occur during integration. Here the input would be the source code of these independent modules and the output would be whether or not there would be bugs after integration. Frequent pattern mining will also help in correcting those defects that are discovered while performing classification. Clustering can help group together the software processes that are similar. The reliability of a software system is inversely proportional to the number of failures and bugs encountered in the software. Using the data mining techniques mentioned above, these bugs and failures can be detected easily and rectified. This saves valuable time, money and the additional resources that might have been required for their detection and resolution along with increasing the reliability and maintainability of the software.

While testing the software, unit testing will usually be performed at the implementation level. However, the other testing techniques will most likely be performed by a tester who isn't from the development team. The job of finding bugs in a code is time consuming and the possible test cases

are infinite. Classification techniques can be used for I/O variables of the system which will produce a network with sets for functional testing. This reduces the time for testing phase and the product can be released as early as possible. As a result, it also prevents the development time from extending which in turns saves cost and resources. The tester has to also look out for behavioral/state changes in the execution of the program. Clustering and classification techniques can be used to look out for such changes in the system behavior for a particular set of data of testing.

Software Development Stage	Data Mining Techniques	Input Data	Data Analysis Result
Requirement Elicitation	Classification	Documentation	Classification of requirements
	Text mining	Mailing lists	Data Summarization
Design	Clustering	Design document	Data gathering, labelling
Implementation	Clustering	Source code	Software processes
	Classification	SCM	Bug tracking
	Text Mining	Source code	Bug tracking
	Frequent Pattern Mining & Association rules	Defect	Defect correction
		Program dependence graph	Neglected conditions
Testing	Classification	I/O variables of software system	A network producing sets for function testing
		Program executions	Software behavior classifiers
	Clustering	Execution Profiles	Clusters of execution profiles

Table 2: Overview of data mining techniques used in different stages of software development

Given above, in Table 2 is a summary of all the information mentioned in this section.

Parameter	Software Engineering with Data Mining	Software Engineering without Data Mining
Time	Using data mining techniques on the software development process will most certainly increase the development time. This is time well utilized as it helps eliminate time spent on handling bugs and failures. Important patterns from the data can be obtained easily, thereby saving large amount of time.	Although the development process will be significantly faster in the absence of the overhead resulting from data mining, the delay caused due to bugs and failures and the time spent on discovering patterns manually will overshadow the significant gains made in time.
Cost	Since data mining does not involve any physical components, the cost incurred by it is negligible. At the same time, it helps in saving money by automating jobs such as bug detection and discovering useful patterns from the data.	Without data mining, chances are that the developmental costs will increase with the extra resources needed for debugging and manual data analysis.
Resources	Using data mining techniques will not require any extra resources since it will run independently on any workstation and can be reused as and when required. It also cuts down on the human and software resources required for debugging and testing.	Just like with cost, the resources needed, especially human resources required to compensate for the lack of automated debugging and testing features will increase significantly.
Reliability	The reliability of the data mining technique depends on the quality of data that is available. As data mining helps in dramatically reducing potential failures that a system might encounter, the reliability of the developmental process increases in direct proportion to the reliability of the data mining technique.	With the absence of data mining, the chances of the system failing due to bugs and faults increases tremendously. This results in a decrease in the reliability of the system
Maintainability	Data mining techniques help to improve the productivity of the development process by discovering useful patterns, maintains	Without data mining, the risk of failure increases. Discovering the cause of the failure can be a time consuming and tedious process involving human

	consistency of the system after every stage of the software development process and helps in fault detection and prevention. Thus the maintainability of the system improves as a result.	resources. Thus the speed with which a system can recover from a failure decreases, thereby causing a decrease in the maintainability

Table 3: Comparison of the software development process with and without the use of data mining

As we have mentioned in our study, the selection of a data mining depends upon the type of data that has to be mined. There is no one technique that can be applied to all data. Every stage generates or works with a unique type of data. Frequent Pattern Mining is best suited for solving sequence data problems. Classification is ideal for large and complex data that can be modelled into graphs. Due to the increased complexity of such data, it is extremely difficult for the human eye to uncover patterns in it and classify it. Classification techniques can effectively automate the discovery of patterns and sub-graphs, thereby making it easy to deal with this type of data. Nothing beats text mining and natural language processing in uncovering useful patterns from pure text data. With the majority of software engineering data being in this form, it can be deemed as the most important and widely used technique in the mining of software engineering data from a quantitative perspective. Clustering is best used to group similar type of data from data that can be quite abstract and vague. Combining clustering with other techniques yields extremely efficient methods of mining software data.

CONCLUSION

In this survey, we have established the need and importance of using data mining techniques to aid software engineering, especially to tackle problems such as the occurrence of bugs, rise in the cost of software maintenance; unclear requirements, etc. that can affect software productivity and quality. Our study has outlined the major research works that have taken place in this field. We have also listed the sources of software engineering data that can be mined, most common stages in the development process as well as the data mining techniques that can be applied in these stages. However, the major contribution of our work lies in the specification of the data mining technique most suited for a particular stage in the development process. We have observed the advantages of using such powerful data mining techniques, especially in terms of time, cost, resources, reliability and maintainability. Finally, we have listed these observations in a tabular form, comparing the performance of software engineering with and without the involvement of data mining.

REFERENCES

[1] Laplante, Phillip (2007). "What Every Engineer Should Know about Software Engineering", Boca Raton: CRC. ISBN 9780849372285.

[2] "Selecting a development approach", Centers for Medicare & Medicaid Services (CMS) Office of Information Service (2008). Re-validated: March 27, 2008. Retrieved 27 Oct 2015

[3] Nabil Mohammed Ali Munassar1 and A. Govardhan, "A Comparison Between Five Models Of Software Engineering", IJCSI International Journal of Computer Science Issues, Volume 07, Issue 05, Page No (94-101), September 2010.

[4] Taylor, Q.and Giraud-Carrier, C. "Applications of data mining in software engineering", International Journal of Data Analysis Techniques and Strategies, Volume 02, Issue 03, Page No (243-257), July 2010.

[5] T. Xie, S. Thummalapenta, D. Lo and C. Liu, "Data mining for software engineering", IEEE Computer Society, Volume 42, Issue 08, Page No (55-62), August 2009.

[6] R. H. Thayer, A. Pyster, and R. C. Wood, "Validating solutions to major problems in software engineering project management," IEEE Computer Society, Page No (65-77), 1982.

[7] C. V. Ramamoorthy, A. Prakash, W. T. Tsai, and Y. Usuda, "Software engineering: problems and perspectives," IEEE Computer Society, Page No (191-209), October 1984.

[8] J. Clarke et al., "Refomulating software engineer as a search problem," IEEE Proceeding Software., Volume 150, Issue 03, Page No (161-175), June 2003.

[9] M. Z. Islam and L. Brankovic, "Detective: a decision tree based categorical value clustering and perturbation technique for preserving privacy in data mining," Third IEEE Conference on Industrial Informatics (INDIN), Page No (701-708), 2005.

[10] M. Aouf, L. Lyanage, and S. Hansen, "Critical review of data mining techniques for gene expression analysis," International Conference on Information and Automation for Sustainability (ICIAFS) 2008, Page No (367-371), 2008.

[11] P. C. H. Ma and K. C. C. Chan, "An iterative data mining approach for mining overlapping coexpression patterns in noisy gene expression data," IEEE Trans. NanoBioscience, Volume 08, Issue 03, Page No (252-258), September 2009.

[12] Mendonca, M. and Sunderhaft, N. "Mining software engineering data: a survey", Data & Analysis Center for Software (DACS) State-of-the-Art Report, No. DACS-SOAR-99-3.

[13] Xie, T., Pei, J. and Hassan, A.E. "Mining software engineering data", Software Engineering - Companion, 2007. ICSE 2007 Companion. 29th International Conference, Page No (172–173).

[14] Kagdi, H., Collard, M.L. and Maletic, J.I. "A survey and taxonomy of approaches for mining software repositories in the context of software evolution", Journal of Software Maintenance and Evolution: Research and Practice, Volume 19, Issue 02, Page No (77–131).

[15] C. CHANG and C. CHU, "Software Defect Prediction Using Inter transaction Association Rule Mining", International Journal of Software Engineering and Knowledge Engineering, Volume 19, Issue 06, Page No (747-764), September 2009.

[16] N. Pannurat, N. Kerdprasop and K. Kerdprasop "Database Reverse Engineering based on Association Rule Mining" , International Journal of Computer Science Issues, Volume 7, Issue 2, Page No (10-15), March 2010
[17] Caiyan Dai and Ling Chen, "An Algorithm for Mining Frequent Closed Itemsets with Density from Data Streams", International Journal of Computer Sciences and Engineering, Volume-04, Issue-02, Page No (40-48), Feb -2016,
[18] S.M.Weiss and C. Kulikowski, "Computer Systems that Learn: Classification and Prediction Methods from Statistics, Neural Nets, Machine Learning and Expert Systems, Morgan Kauffman", Morgan Kaufmann Publishers Inc, ISBN:1-55860-065-5.
[19] U. M. Fayyad, G. PiateskyShapiro, P. Smuth and R. Uthurusamy, "Advances in Knowledge Discovery and Data Mining", AAAI Press, ISBN:0-262-56097-6.
[20] M. Halkidia, D. Spinellisb, G. Tsatsaronisc and M. Vazirgiannis, "Data mining in software engineering", Intelligent Data Analysis 15, Page No (413–441), 2011
[21] M. Berry and G. Linoff, "Data Mining Techniques For marketing, Sales and Customer Support", John Willey and Sons Inc., ISBN: 978-0-471-17980-1.
[22] K.Selvi, "Identify Heart Diseases Using Data Mining Techniques: an Overview", International Journal of Computer Sciences and Engineering, Volume-03, Issue-11, Page No (180-187), Nov -2015,
[23] L. Kauffman and P.J. Rousseeuw, "Finding Groups in Data: An Introduction to Cluster Analysis", John Wiley and Sons, ISBN - 9780470317488.
[24] Lovedeep, Varinder Kaur Atri, "Applications of Data Mining Techniques in Software Engineering", International Journal of Electrical, Electronics and Computer Systems (IJEECS), Volume 02, Issue 05, Page No (70-74), June 2014.
[25] M. Gegick, P. Rotella and T. Xie, "Identifying security bug reports via text mining: an industrial case study", Mining Software Repositories (MSR), 7th IEEE Working Conference, Page No (11 – 20), 2010.
[26] P. Runeson, and O. Nyholm, "Detection of duplicate defect reports using natural language processing", Software Engineering, 2007. ICSE 2007. 29th International Conference, Page No (499 – 510), 2007.
[27] Ian Somerville, "Software Engineering", AddisonWesley, Chapter 30, 4th edition, ISBN - 9783827370013.
[28] J. Estublier, D. Leblang, A. Van Der Hoek, R. Conradi, G. Clemm, W. Tichy and D. WilborgWeber, "Impact of software engineering research on the practice of software configuration management", ACM Transactions on Software Engineering and Methodology, Volume 14, Issue 04, Page No (383-430), October 2005 .
[29] H.A. Basit and S. Jarzabek, "Data mining approach for detecting higher level clones in software", IEEE Transactions on Software Engineering, Volume 35, Issue 04, Page No (497 – 514)
[30] Iam Sommerville, "Requirements Engineering A good practice guide", Ramos Rowel and Kurts Alfeche, John Wiley and Sons, 1997, ISBN – 9780470359396.

AUTHORS PROFILE

Nidhin Thomas received his B.Tech degree in Information Technology from Vishwakarma Institute of Technology under the University of Pune. He is currently pursuing his M.Tech degree in Computer Science and Engineering from VIT University, Vellore. His areas of interest include Machine Learning, Data mining and analysis, Human Computer Interaction and Natural Language Processing

Atharva Joshi received his B.Tech degree in Computer Science from Sinhgad Institute of Technology under the University of Pune. He is currently pursuing his M.Tech degree in Computer Science and Engineering from VIT University, Vellore. His areas of interest include Parallel computing, Graphic Designing, Programming and Databases.

Rishikesh Misal is currently a first year post graduate Computer Science and Engineering Student in Vellore, India at the Vellore Institute Of Technology. He will complete his post graduation in 2016 with a Master of Computer Science and Engineering. He has completed his graduation from Vidyalankar Institute Of Technology, Mumbai, India. He worked as a free lancer in several computer networking and data mining related projects. After post graduation he plans to pursue a career in research and development field of software systems.

Manjula R received her B.E in Computer Science & Engineering from University of Vishwesvaraya and Engineering, Bangalore, Karnataka State, India in 1992 and M.E in Software Engineering from Anna University, Tamil Nadu, India in 2001 and Ph.D. in Computer Science and Engineering from VIT University, Vellore, India. Presently she is working as Associate Professor at the School of Computing Science and Engineering at VIT University, Vellore, India. Her area of specialization includes Software Process modeling, Software Testing and Metrics, Service Oriented Architecture, Data Mining and Social Network Analysis.