Software Defect Prediction Using Linear Svm

Gunjan Arora Krishna Gupta

Abstract: An efficient software product can be generated with the help of various techniques properties and procedures provided by software engineering. Therefore, it is necessary for a software developer to ensure that developed product is less in cost, time and budget. Careful planning is required before working on software projects because it is large in size and developer must have proper knowledge about the requirement of the user and all the systematic procedures for the development of software The abstract-present is the model of software engineering which is used to generate the source code from the sequence model. The code that is generated for the one phase will be given as input to generate code for the second phase. To generate reliable code, the improvement will be proposed in the abstract-present model. To do so, the SVM classifier will be used to classify required and non-required code to generate next phase of code. The proposed model is implemented in python and results are analyzed in terms of accuracy. Precision, recall and F-measure

keywords: Software defect prediction, SVM, LLE-SVM, LTSM-SVM

1 INTRODUCTION

Software is identified as the compilation of programs, processes, data and documentation. The software for a particular business is designed by considering the hardware and operating system. The operating system is also called platform. Engineering is a methodical technique which is used to develop software. The designing process of software is very complicated. Various guidelines are followed during the designing process [1]. The gaming techniques of reverse engineering can be categorized into a number of categories. These categories are known as static approaches, dynamic approaches and memory patching. The static approaches are used to analyze the binary data or application can be analyzed while dynamic approach communicates with the software during run time [2]. The memory patching is provided by compelling the required state through the manipulation of the memory of the game rather than focusing on the logic of game.

A. Static Approaches

In order to understand the binary files and the way of application distribution, the disassemblers can be used. With the help of this technique, the machine language is converted into assembly language. The availability of disassembly can detect any type of vulnerabilities present within the code. This may prove advantageous to the applications in which these systems are implemented. The utilized data structures can be recognize in an improved manner for detecting them within the memory to provide memory patching [3]. The pseudo-random algorithm (PRNG) employed within the application can be detected along with disassembly application. The predictability within PRNG can predict ingame procedures. For complementing the disassembly, the accessible binary can be matched with the custom code. in order to load a custom DLL, the DLL imports can be examined and the addresses can be changed. The custom third party code can be executed prior to or later than the actions of application.

- *Gunjan Arora Krishna Gupta*
- *M.Tech Scholar, Yagyavalkya Institute of Technology, Jaipur*
- *To Assistant Professor Krishna Gupta, YIT, Jaipur*

One more entry point for custom code insertion is the render loop. This loop can be called regularly.

B. Dynamic Approaches

Automation is the most common technique. This technique is used to interrelate with the games during their runtime. The bots and macros are utilized for automatic resources gathering. The spell-combos or skip repetitive tasks are performed for this purpose [4]. Within the simple case of macro, the fire key is returned to the mouse wheel. This creates massive fire during the scrolling. The reverse engineering is used to develop these tools. The memory locations having related data can be read directly rather than sampling of pixels from the display. This approach provides a rapid and more precise access to the preferred game. Instead of simulating mouse movement, the keystrokes can be inserted directly for providing communication with the game. This approach provides a good tool support for automation to imitate the keystrokes and mouse actions. Following are the dynamic approaches and mouse actions used within reverse engineering [5]:

- Debugger: The breakpoints can be set according to the essential events involved within an application. In addition, this gives knowledge about control flow or changing variables [6].
- Proxy: In the networked games, the network interaction of the application can be captured, scrutinized and replayed. The used protocol can be reverse engineered. The data on-the-fly can be utilized to adapt the interaction.
- Fuzzing: By sending the arbitrarily generated input to the game and by monitoring the behaviors, the susceptibilities can be discovered. This can give a improved perceptive of the utilized algorithms applications [7].

C. Memory Patching

With the help of memory patching, the most popular approach can be applied to improve the run time of an application. This method is just based on the screening of a program as a set of the memory which is not valid in other dynamic approaches. On the basis of need, the bits can be changed. After this, bits are recognized and understood [8] [9].

Benefits

- Generic: There memory patching does not depend on the programming language or employed structures. The memory patching performs according to the collected and interpreted programs.

- Fast: The memory patching detects and manipulates the value of address quickly as compared to the binary analysis. This situation is identified normal using bigger and extremely difficult programs.
- Invisible: Any type of changes made to the memory remains invisible to the program that needs modifications. The eternally written value and the value being added within the program cannot be distinguished [10].

2 RELATED WORK

The requirement analysis was a very important process in the software development projects. The utilization of the particular client requirements and management had several effects in the software projects. Therefore, the improvement in this domain in terms of both educational and industrial fields was essential. In this study, a model of CMMI was proposed to reveal the development and management need. It also specified the different objectives and practical stages [11]. The need of the management and the main challenges and problems experienced by it were listed in this work using CMMI model and its normal behavior. An important role was played by the technical documentation to in determine the success and failure of any software. For this purpose, a Software requirement specification document had been used. This technical document contained all the needed data like features of the produce. In the earlier time, various developments were executed to enhance the quality of the SRS by utilizing different features of the good. The acceptable success rate for the product was not attained in outcomes due to which more improvement was needed. Therefore, an effectual approach was proposed in this work to solve this issue [12]. In this study, two main open research directions were provided. These directions were identified as the computerization of technological aspects and the requirement of the left over non-computerized sense making. The human analysts could only apply these directions. In this study, the issue regarding the research was reviewed surveyed along with the research based on the gaining of knowledge. This supported to provide quantification of the crucial human factors. This also helped to give upcoming research directions for capturing the difficulty of hardware reverse engineering. This could assist to reveal the different hardware requirements required in the reverse engineering along with the existing challenges [13]. In this work, a novel methodology was proposed for reverse engineer and to recognize the security faults. An analysis was presented in this work about different communications protocols being used in ISO/IEC 14443-B public transportation card by several clients. The application of method along with the hardware tool can give the access of confidential data. With the help of proposed approach, the tag-reader interactions could be captured. In this approach, the tags and readers were also imitated [14]. This phenomenon applied the idea of the interesting domain. The programmer behavior within Reoom encrypted two explanations. Reoom was a new light-weight static scrutiny system. This system was used in this type of methods. A number of comparisons were made amid Womble and the some other third-party open source applications. The different simulation outcomes revealed that the results showed improved performance in terms of some metrics. These metrics were known as overall precision, recall and accuracy. As compared to Womble, a better tradeoff was identified among full Reoom. The locals and parameters used in this study gave improved recall value than the return types [15]. A methodical survey was presented on this topic in this work. This review helped to provide suitable solutions to three research queries. The model-driven reverse engineering techniques were scrutinized on the basis of several features such as extensibility, automation and generosity in the reverse engineering procedure. In this work, a comparison amongst fifteen different approaches was performed. These approaches applied the model-driven reverse engineering techniques. The different problems that rose within the model-driven reverse engineering approaches were explained in this study as well [16]. The imaging metrics of a triumphant tomography are also presented in this work along with the mixture of superior 3-D image processing. A scrutiny to computerize RE was used to reduce the associated time and cost of these systems. In this paper, the two PCBs as a four-layered custom design board and the intricate commercial board were offered as well. The performances of proposed algorithms were analyzed on the basis of performed tests. The tested results showed enhancement in the proposed techniques as compared to the conventional techniques [17].

3 MOTIVATION

The method in which the code can be generated from the designed UML model is known as reverse engineering. For model generation, sequence diagram plays an important role such that the efficient code can be derived using this method. To implement the reverse engineering, the base paper research applies abstract-present model. For developing the source code of second stage, the source code of first state is analyzed within the abstract-based model. To generate the syntax code in this model, the abstract-syntax tree is generated. For deriving the new phase source code from sequence diagram, the base paper visualizes the sequence diagrams. This research aims to identify the required source code to perform classification such that the reliability of designed model can be increased. This research aims of cover the research gaps of previous researches among which two important are mentioned below:

1. It is important to classify the sequence diagram that is defined by the complete flow of project. Based on the sequence of diagrams, the identification of required and non-required functions is done in case of base paper. This research aims to use SVM classifier for classification such that the dataset of function can be collected.
2. It is important to minimize the execution time required by the previous algorithm for classifying the functional and non-functional requirement functions.

4 RESEARCH METHODOLOGY

The documents behaving similarly exist in similar clusters based on the relevancy of required information. This behavior is in relevance to the individual words of information content available. Therefore, there is a specific grouping of words in similar cluster. Certain assumptions are applied based on this method. The important aim of clustering the keywords extracted from functional requirements of software system is to generate the semantically coherent groups of natural language words. A set of disjoint groups that includes semantically similar words are clustered here. A set of disjoint groups that include semantically similar words are generated here. Using the clusters in C, the conceptual themes which can pass the original test are described ideally. Such themes are useful in representing the quality constraints of a system. Identifying the optimal cluster configurations that are best fitted is known as the NP-hard issue that is seen here. However, near-optimal solutions can be provided using the certain experiments. The Average Linkage clustering algorithm is denoted here by AL and test semantic similarity by TSS. To perform classification, this research applies linear SVM. Post-pruning techniques are applied to evaluate the linear SVM. The validation set is used to prune them. From the IVR dataset, the null and non null function, are categorized by this research. To perform data classification, NFR matrix is applied along with the linear SVM classifier. With respect to execution time and accuracy, the performances of proposed and existing techniques are compared to evaluate the improvements achieved. To perform data classification in existing method, only NFR matrix was applied. However, the proposed method aims to perform classification by combining the NFR matrix with linear SVM algorithm. The NFR is the Non-functional requirement matrix. “NGD_{wiki} Matrix” is the representation of Non-functional matrix. “Clusters of SRS Words” is the segmentation of the Non-functional matrix. NFR’s label means the target set in which we want to classify the data which is functional or non –functional attributes. The java is the programming language file of which we have taken the dataset for the functional or non-functional requirements analysis. Initially, to perform classifications within the systems, linear SVMs were introduced which were then further also implemented in regression and rank learning mechanisms. A binary classifier was the initial form of linear SVM from which either positive or negative output of the learned function was achieved. The classifiers that discriminate data points amongst two categories are known to be the binary SVMs. An n-dimensional vector is used to represent each data object. Every data point belongs to only one of the two classes. A hyperplane is present that separates each of the linear classifier. The hyperplane that has the largest margin is picked by SVM such that maximum separation amongst the two classes can be achieved. The sum of the least distance from the separation of hyperplane to the closest data point available within two categories is known as margin. The unseen of testing data points can be classified correctly by the hyperplane. For supporting various issues of nonlinear classification, the mapping from input space to the feature space is supported by SVM. The mapping function that would destruct the dimensionality is eliminated with the help of kernel trick. Thus, the linear classification in the new space and nonlinear classification in the original space are made equal here. The input vectors are mapped to a higher dimensional space in which there is generation of a maximal separating hyperplane for equalizing the linear and non-linear classification.

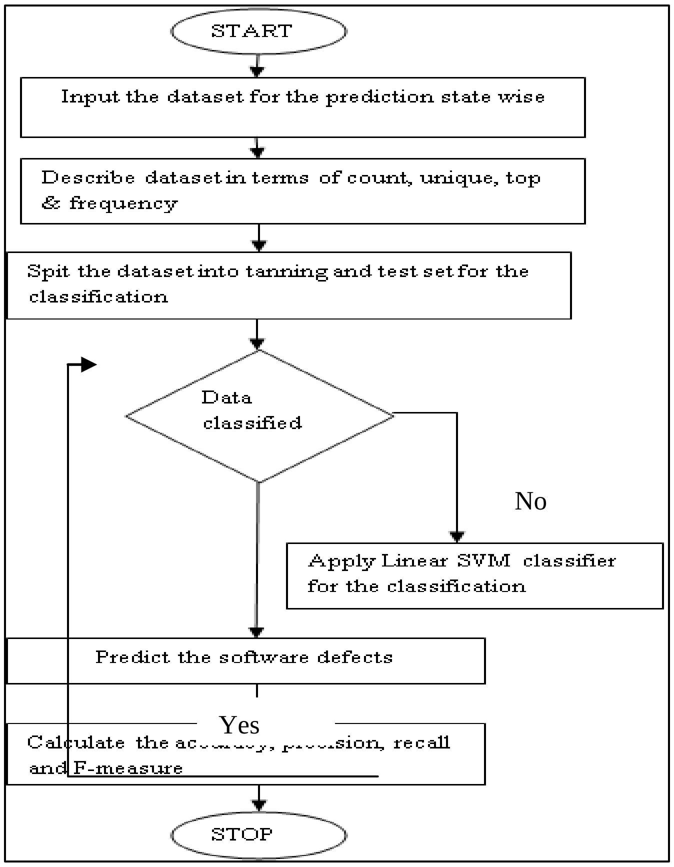
START
Input the dataset for the prediction state wise
Describe dataset in terms of count, unique, top & frequency
Spit the dataset into tanning and test set for the classification
Data classified
No
Apply Linear SVM classifier for the classification
Predict the software defects
Yes
Calculate the accuracy, precision, recall and F-measure
STOP

Figure 1: *Proposed Methodology*

5 EVALUATION

In this section, the dataset is described on which classification methods are applied for the defect prediction. The various performance analysis parameters which are used for the analysis are also described in detail with the final results

I. Dataset

M1 is a NASA spacecraft instrument written in "C".Data comes from McCabe and Halstead features extractors of source code. These features were defined in the 70s in an attempt to objectively characterize code features that are associated with software quality. The nature of association is under dispute. The McCabe and Halstead measures are "module"-based where a "module" is the smallest unit of functionality. In C or Smalltalk, "modules" would be called "function" or "method" respectively. An alternative interpretation of Fenton's example is that static measures can never be a definite and certain indicator of the presence of a fault. Rather, defect detectors based on static measures are best viewed as probabilistic statements that the frequency of faults tends to increase in code modules that trigger the detector. By definition, such probabilistic statements will are not categorical claims with some a non-zero false alarm rate. The research challenge for data miners is to ensure that these false alarms do not cripple their learned theories. The McCabe metrics are a collection of four software metrics: essential complexity, cyclomatic complexity, design complexity and LOC, Lines of Code.

6 PERFORMANCE ANALYSIS PARAMETERS

Accuracy: Accuracy is defined as the number of points correctly classified divided by total number of points multiplied by 100,

$$\text{Accuracy} = \frac{\text{Number of points correctly classified}}{\text{Total Number of points}} * 100$$

Precision: In pattern recognition, information retrieval and binary classification, precision (also called positive predictive value) is the fraction of relevant instances among the retrieved instances.

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

Recall: Recall is the fraction of relevant instances that have been retrieved over the total amount of relevant instances.

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

7 RESULT AND DISCUSSION

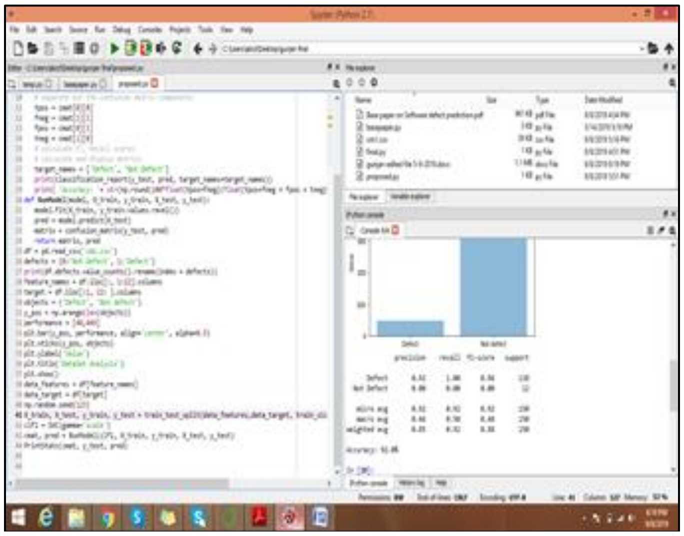

Figure 2: *Implementation*

As shown in figure 4.5, the implementation of Linear classifier is shown for the classification of dataset into defect and non-defect. When the linear SVM is applied on the input dataset the accuracy upto 92 percent is achieved with the CM1 dataset

Table 1: *Accuracy Analysis*

Algorithm	*Accuracy*
SVM	*81.56 percent*
LLE-SVM	*82.22 percent*
LTSA-SVM	*88 percent*
Linear SVM	*92.5 percent*

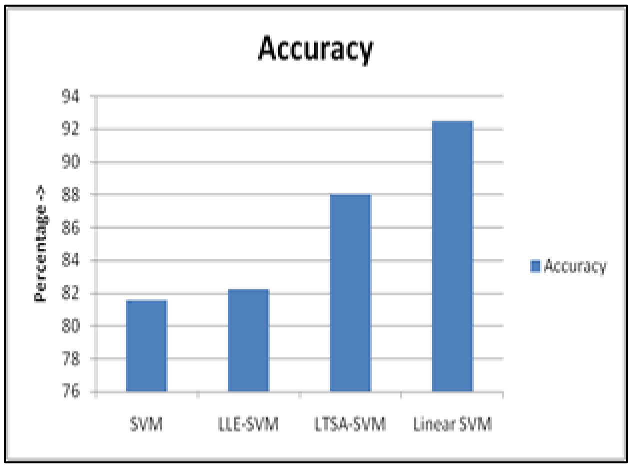

Figure 3: *Accuracy Comparison*

As shown in figure 3, the accuracy of for the software defect prediction is analyzed. The accuracy of 92 percent is achieved using Linear SVM classifier as compared to LTSA Algorithm with has maximum accuracy 88 percent on CM1 dataset

Table 2: *Precision Analysis*

Algorithm	*Precision*
SVM	*84.42 percent*
LLE-SVM	*86.21 percent*
LTSA-SVM	*90 percent*
Linear SVM	*92 percent*

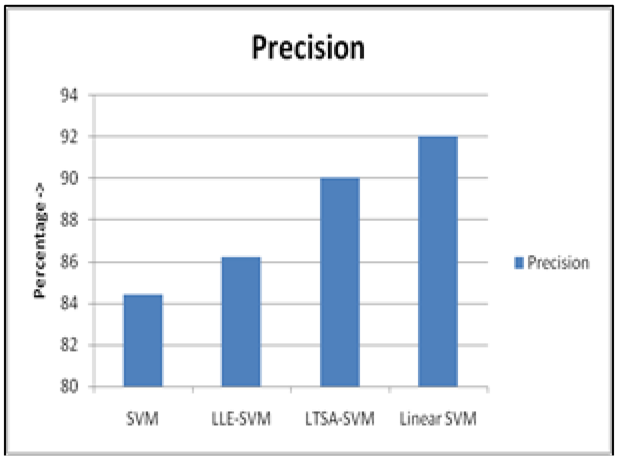

Figure 4: *Precision Analysis*

As shown in figure 4, the precision of for the software defect prediction is analyzed. The precision of 92 percent is achieved using Linear SVM classifier

Table 3: *Recall Analysis*

Algorithm	*Recall*
SVM	*91.22 percent*
LLE-SVM	*94.12 percent*
LTSA-SVM	*95 percent*
Linear SVM	*97 percent*

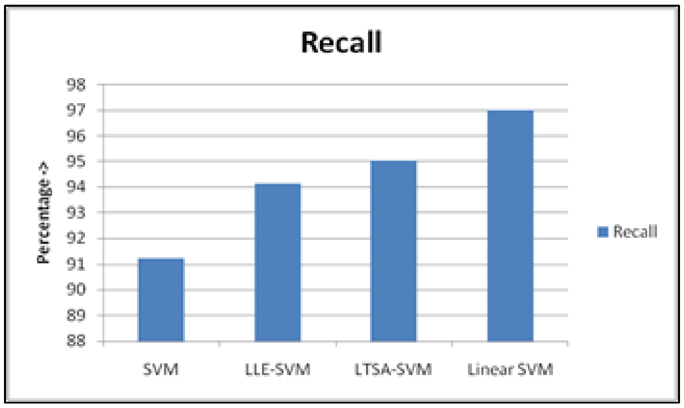

Figure 5 *Recall Analysis*

As shown in figure 5, the recall of for the software defect prediction is analyzed. The recall of 92 percent is achieved using Linear SVM classifier

Table 4: *F-measure Analysis*

Algorithm	*Recall*
SVM	*87 percent*
LLE-SVM	*89 percent*
LTSA-SVM	*90 percent*
Linear SVM	*93 percent*

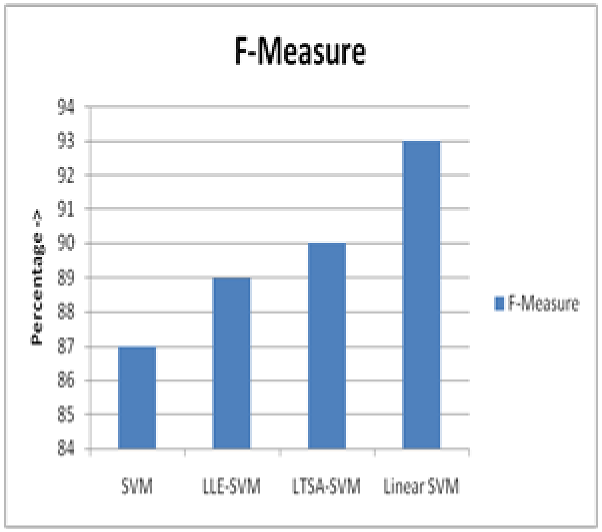

Figure 6 F*-measure Analysis*

As shown in figure 6, the f-measure of for the software defect prediction is analyzed. The f-measure of 92 percent is achieved using Linear SVM classifier

8 CONCLUSION

The experimental results of this research have shown that for generating source code, reverse engineering is known to be a highly efficient technique. For generating the source code, the abstract-present model was applied in the base paper. For generating a new sequence code, the code of previous sequence is applied in the abstract-present model. To generate source code, extraction, abstraction and visualization processes are applied. This research applies classification technique prior to the visualization with the help of which the reliability of reverse engineering based abstract-present model is increased. The proposed algorithm is implemented in python and results are analyzed in terms of certain parameters. In the proposed method is the based on linear SVM. The parameters are analyzed in terms of accuracy, precision, recall and F-measure

9 REFERANCES

[1] Jirayus Jiarpakdee, Chakkrit Tantithamthavorn, Akinori Ihara, Kenichi Matsumoto, "A Study of Redundant Metrics in Defect Prediction Datasets", IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW), volume 5, issue 14, pp- 1937-1946, 2016.

[2] Chakkrit Tantithamthavorn, Shane McIntosh, Ahmed E. Hassan, Kenichi Matsumoto, "Comments on Researcher Bias: The Use of Machine Learning in Software Defect Prediction", IEEE Transactions on Software Engineering, Volume: 42, Issue: 11, pp- 758-764, 2016.

[3] B. Shakya, Mark M. Tehranipoor, Swarup Bhunia, Domenic Forte, "Introduction to Hardware Obfuscation: Motivation, Methods and Evaluation", Hardware Protection through Obfuscation. Springer, ch. 1, pp. 3–32, 2017.

[4] Shahed E. Quadir, Junlin Chen, Mark Mohammad Tehranipoor, "A survey on chip to system reverse engineering", Journal on Emerging Technologies in Computing Systems, vol. 13, no. 1, pp. 6:1–6:34, 2016.

[5] Pramod Subramanyan ; Nestan Tsiskaridze ; Wenchao Li ; Adrià Gascón ; Wei Yang Tan ; Ashish Tiwari, "Reverse Engineering Digital Circuits Using Structural and Functional Analyses", IEEE Trans. Emerging Topics Computing, vol. 2, no. 1, pp. 63–80, 2014.

[6] U. Guin, Ke Huang, Daniel DiMase, John M. Carulli, Mohammad Tehranipoor, Yiorgos Makris, "Counterfeit Integrated Circuits: A Rising Threat in the Global Semiconductor Supply Chain," Proceedings of the IEEE, vol. 102, no. 8, pp. 1207–1228, 2014.

[7] S. Bhunia, Michael S. Hsiao, Mainak Banga, Seetharam Narasimhan, "Hardware Trojan Attacks: Threat Analysis and Countermeasures," Proceedings of the IEEE, vol. 102, no. 8, pp. 1229– 1247, 2014.

[8] A. Vijayakumar, Vinay C. Patil, Daniel E. Holcomb, Christof Paar, Sandip Kundu "Physical Design Obfuscation of Hardware: A Comprehensive Investigation of Device and Logic-Level Techniques," IEEE Transaction Information Forensics and Security, vol. 12, no. 1, pp. 64–77, 2017.

[9] J. Nam, W. Fu, S. Kim, T. Menzies, and L. Tan, "Heterogeneous Defect Prediction," Transactions on Software Engineering (TSE), vol. 44, no. 9, pp. 874–896, 2017.

[10] M. Ortu, G. Destefanis, B. Adams, A. Murgia, M. Marchesi, and R. Tonelli, "The jira repository dataset: Understanding social aspects of software development," in Proceedings of the International

Conference on Predictive Models and Data Analytics in Software Engineering (PROMISE), vol. 16, no. 19, pp. 921–928, p. 1, 2015.

[11] Senay Tuna Demirel, Resul Das, “Software Requirement Analysis: Research Challenges and Technical Approaches”, 6th International Symposium on Digital Forensic and Security (ISDFS), vol. 16, no. 15, pp. 994-999, 2018.

[12] Syed Waqas Ali, Qazi Arbab Ahmed, Imran Shafi, “Process to Enhance the Quality of Software Requirement Specification Document”, International Conference on Engineering and Emerging Technologies (ICEET), vol. 40, no. 19, pp. 878–895, 2018.

[13] Marc Fyrbiak, Sebastian Strauß, Christian Kison, Sebastian Wallat, Malte Elson, “Hardware Reverse Engineering: Overview and Open Challenges”, IEEE 2nd International Verification and Security Workshop (IVSW), vol. 19, no. 55, pp. 1951–1958, 2017.

[14] Paula Fraga-Lamas, Tiago M. Fernandez-Carames, “Reverse Engineering the Communications Protocol of an RFID Public Transportation Card”, IEEE International Conference on RFID (RFID), vol. 47, no. 17, pp. 190–194, 2017.

[15] Tuan Anh Nguyen, Christoph Csallner, “Reverse Engineering Object-Oriented Applications Into High-Level Domain Models With Reoom”, IEEE/ACM 39th IEEE International Conference on Software Engineering Companion, vol. 54, no. 17, pp. 374–385, 2017.

[16] Claudia Raibulet, Francesca Arcelli Fontana and Marco Zanoni, “Model-Driven Reverse Engineering Approaches: A Systematic Literature Review”, IEEE, vol. 22, no. 61, pp.109–117, 2017.

[17] Navid Asadizanjani, Mark Tehranipoor, and Domenic Forte, “PCB Reverse Engineering Using Nondestructive X-ray Tomography and Advanced Image Processing”, IEEE Transactions on Components, Packaging and Manufacturing Technology , Volume: 45, Issue: 6, pp- 106-118, 2017.