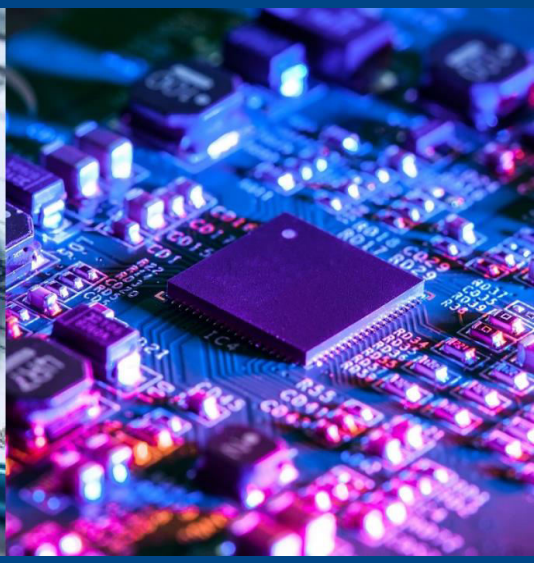
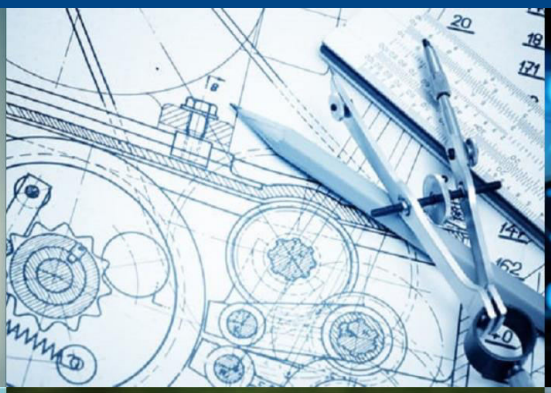
ISSN: 2582-7219

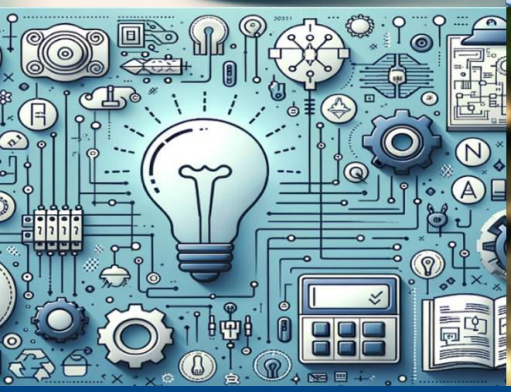




International Journal of Multidisciplinary Research in Science, Engineering and Technology

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)



Impact Factor: 7.521

Volume 8, Issue 1, January 2025

© 2025 IJMRSET | Volume 8, Issue 1, January 2025| DOI: 10.15680/IJMRSET.2025.0801030

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 7.521| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

An Efficient Approach of Software Clone Detection and Risk Assessment

Dr.A. Senthilkumar

Assistant Professor, Department of Computer Science with Data Analytics, Sri Ramakrishna College of Arts & Science, Coimbatore, India

ABSTRACT: The paper helps to identify the quality of the software using different metrics. The main aim of this project is to create an analysis tool to find the quality of the software. The user can identify whether their program is a quality one or not. This work contains 5 modules, which are as follows Software project selection module, LOC (Lines of Count) calculation module, Methods and attribute extraction, Quality analysis and report. The first module in the project is the program/code selection process, which the user needs to test. The user can give any program file or software files to test the quality. The next module is calculating LOC ie, line of code calculation which calculates total number of lines used in the program and finding unused lines from that. This module extracts total functions/methods used in the program, using this process; user can get the details of the objects, variables and a method Based on the above modules, the quality of the software has been tested. This gives the overall quality of the program in percentage. The report gives the graphical representation of the quality verification data. And also list of tests made by the user will be stored in the database for future references

I.INTRODUCTION

As demand of software increases, software development organizations are in the greater need to develop products rapidly. This in turn demands highly potential software developers or knowledge workers in different technologies. Since we have multiple technologies for code development, discover and elicit code according to the requirement is the initial task and supply appropriate code to the need is focused in the next phase. Code knowledge base consists of valuable data that can be inferred for variety of usages. So far research focus was on topic models, association rules, and heuristics to mine code repositories for traceability and for monitoring changes in source code [2, 3, 4, 5]. It is challenging to identify the code fragment s purpose for every code fragment in the knowledge base with the focus of improving programmer's potential in terms of code construction. Code knowledge base consists of code segment and its purpose. Knowledge mapping to the current necessity of the project requirement scenario is very important and it has to be substantially substituted by the time of software development. Programmers often ignore to find optimum code; their main focus is on completion of modules in the projects.

Knowledge base consists of code segments which are extracted from code logs according to comment lines or syntax lines (symbols). The purpose of the code segment is known only when the developer documents the purpose either as comments or as a documentation note. In this KM framework knowledge acquisition happens only from the code files, so there is no detail documentation are acquired, Now the query is whether the extracted code segment can fulfill any software requirement or not, Only when the purpose of the code is known, it can be utilized.

In order to understand the extracted code, precise perception for a particular code segment is essential. Constraint satisfaction and retrieval are the steps followed for supplying solution code knowledge. Code retrieval from knowledge base is based on key word/ text search that acts as input parameter for querying in the knowledge base. Substantial supply of code knowledge is an experimental process for agile code building in software industries. This article shows an overview of the process of substantial supply of code knowledge and describes the model which is designed for substantial supply of code knowledge. User specifies the requirement in simple sentence, our model explains the technique to discover and elicit the appropriate code from the knowledge base by mining through the keywords. Any search application, has some typical phases with indexing on right and retrieval of results on left. This work proposes a solution to the coding problem that makes it easier for programmers to follow the offered recommendations during development. The premise is that source code projects can be mined to create a knowledge base and build a suggestion

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 7.521| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

model that provides substantial optimum code.

To that end, this work features a model that analyzes source code and uses data from the Solution Knowledge Database (SKB) to create feature sets for recognizing a set of code patterns. Specifically, source code is represented using an Abstract Syntax Tree [5], which provides the ability to extract statements. This work is based on KM Trajectory Service Frame work [7] which is framed to reduce the dependency on human resources in software development organizations. Knowledge Champions and KM team members have to contribute their time in creation and maintenance of solution knowledge base. This provides functional KM solutions for software process improvement. This work proposes the extraction of code sets from project repositories to present to the user a set of methods that supplies code segment for each requirement.

- Software development process, utilizes code data from the Solution knowledge Database.
- Knowledge Map algorithm processes thousands of code fragments, discover and supply the required ones.
- Discovery of the required code fragment from code base.
- Utilization of code fragment increases the success count each time.

II. PROCESS DESCRIPTION

The implementation phases of this project are construction, operation and the installation lies on the new system. The most crucial and very important stage in achieving a new successful system and is giving confidence on the new system that it will work efficiently and effectively. The following modules are involved to predict the clone in the respective software.

A. Software project selection module:

The first module in the project is the program/code selection process, which the user needs to test. The user can give any program file or software files to test the quality.

B. LOC (Lines of Count) calculation module:

The next module is calculating LOC ie, line of code calculation which calculates total number of lines used in the program and finding unused lines from that.

C. Methods and attribute extraction

This module extracts total functions/methods used in the program, using this process; user can get the details of the objects, variables and methods.

D. Clone detection using operation code sequences:

This module finds the duplication between the software's using the operation code sequences. Using this module, the admin can detect duplicate projects and programs easily.

E. Quality analysis

Based on the above modules, the quality of the software has been tested. This gives the overall quality of the program in percentage. The user can check whether the given program is efficient or not. Using this module two or more program will be compared with time and use of unique code features.

F. Report.

The report gives the graphical representation of the quality verification data. And also list of tests made by the user will be stored in the database for future references.

III. RESULTS AND DISCUSSION

All the modules were tested individually using both test data and live data. After each module was ascertained that it was working correctly and it had been "integrated" with the system. Again the system was tested as a whole. We hold the system tested with different types of users. The System Design, Data Flow Diagrams, procedures etc. were well documented so that the system can be easily maintained and upgraded by any computer professional at a later.

© 2025 IJMRSET | Volume 8, Issue 1, January 2025| DOI: 10.15680/IJMRSET.2025.0801030

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 7.521| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

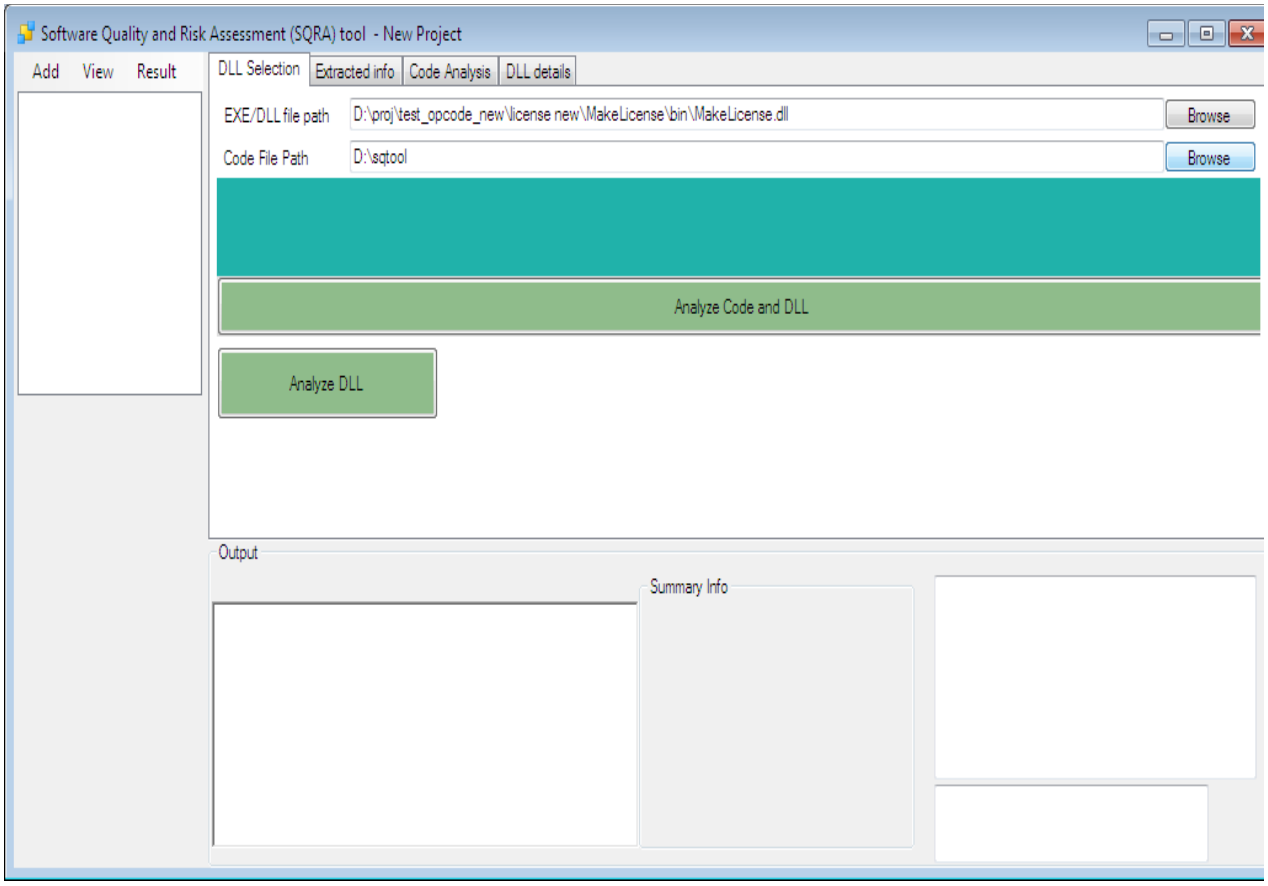


Fig 1 Design for OP Code

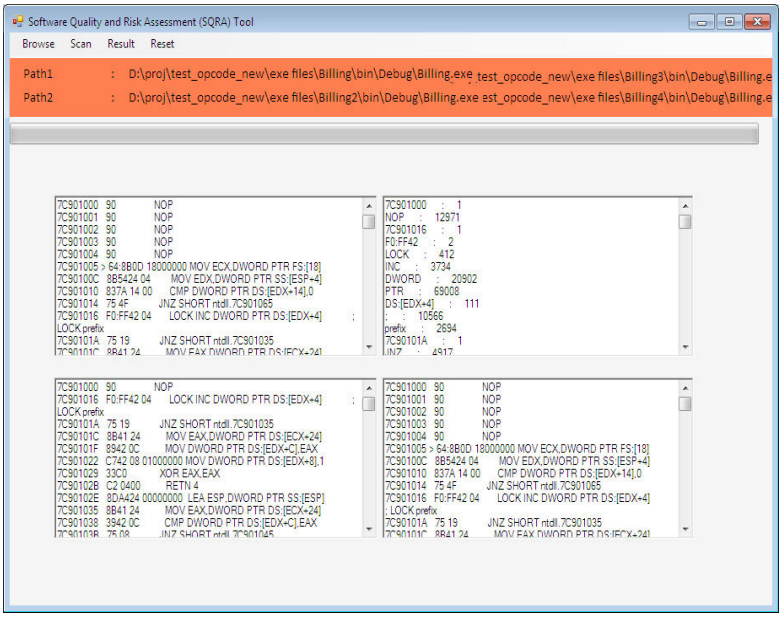


Fig 2 load the data

© 2025 IJMRSET | Volume 8, Issue 1, January 2025| DOI: 10.15680/IJMRSET.2025.0801030

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 7.521| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

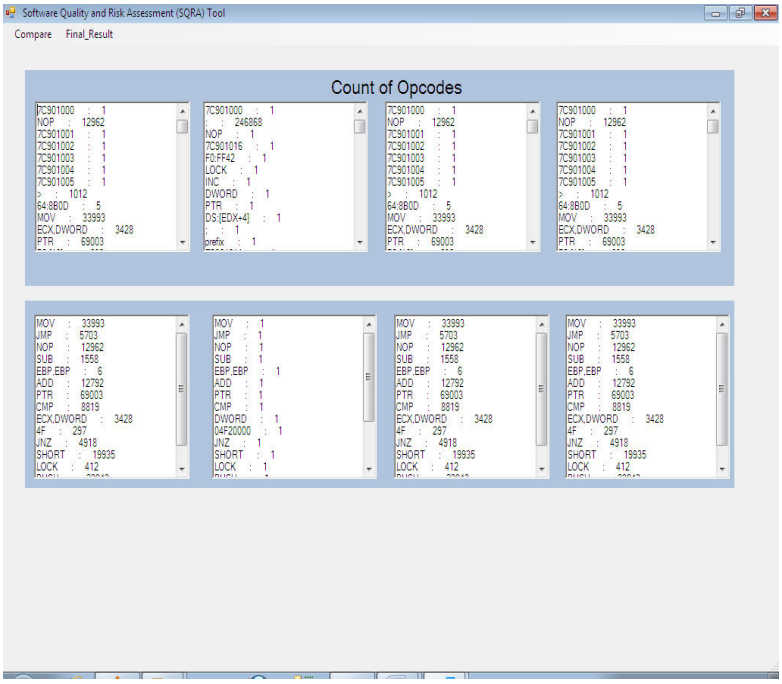
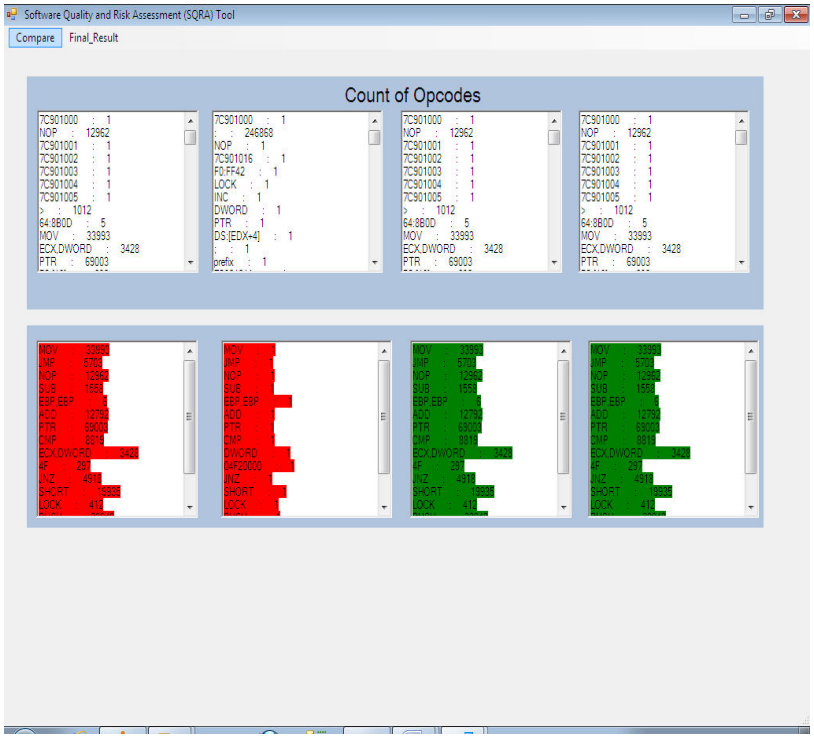


Fig-3 Count of Opcodes



ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 7.521| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

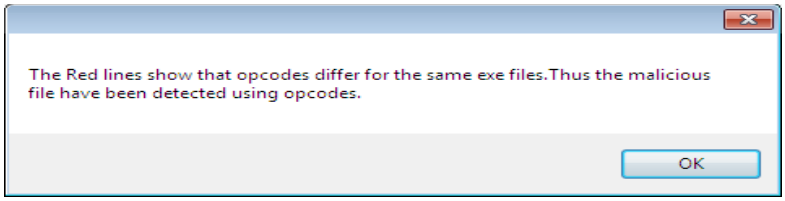


Fig-4 Results of Clone detection

IV.CONCLUSION AND FUTURE WORK

The proposed SQA system used Opcode with sequence and frequency analysis patterns generated by disassembling the inspected executable files to extract features from the inspected files, the system extracts the methods and body content to get the opcodes. Opcode extraction algorithms with sequence analysis are used as features during the testing process with the aim of identifying unknown malicious code and quality of the program. The proposed system performed an extensive evaluation using a test collection comprising more than 30 methods with 4 assembly files. The evaluation consisted of three experiments. In the first experiment, this found that the frequency and sequence representation then it will count the opcodes and provides the results. The proposed tool has been named as SQA, which provides a time-consuming test results for malicious data finding.

REFERENCES

[1] Eytan Adar. GUESS: a language and interface for graph exploration. In Proceedings of the 2006 Conference on Human Factors in Computing Systems (CHI'06), pp. 791-800, Montr´eal, Qu´ebec, Canada, April 2006. (PDF)
[2] Eytan Adar and Miryung Kim. SoftGUESS: Visualization and Exploration of Code Clones in Context. In the proceedings of the 29th International Conference on Software Engineering (ICSE'07), Tool Demo, pp.762-766, Minneapolis, MN, USA, May 2007 .
[3] R. Agrawal and R. Srikant. Mining Sequencial Patterns. In Proceddings of the 11th Infernation Conference of Data Engineering (ICDE'95), pp. 3-14, Taipei, Taiwan, March 1995.
[4] Alfred Aho, Ravi Sethi and Jeffrey Ullman. Compilers, Principles, Techniques and Tools. Addition-Wesley, 1986.
[5] A. Aiken. A system for detecting software plagiarism (moss homepage). URL http: //www.cs.berkeley.edu/aiken/moss.html. 2002.
[6] Raihan Al-Ekram, Cory Kapser, Michael Godfrey. Cloning by Accident: An Empirical Study of Source Code Cloning Across Software Systems.International Symposium on Empirical Software Engineering (ISESE'05), pp. 376-385, Noosa Heads, Australia, November 2005.
[7] Giuliano Antoniol, Gerardo Casazza, Massimiliano Di Penta, Ettore Merlo. Modeling Clones Evolution through Time Series. In Proceedings of the 17th IEEE International Conference on Software Maintenance (ICSM'01), pp. 273-280, Florence, Italy, November 2001. (PDF)
[8] G. Antoniol, U. Villano, E. Merlo, and M.D. Penta. Analyzing cloning evolution in the linux kernel. Information and Software Technology, 44 (13):755-765, 2002.
[9] Lerina Aversano, Luigi Cerulo, and Massimiliano Di Penta. How Clones are Maintained: An Empirical Study. In Proceedings of the 11th European Conference on Software Maintenance and Reengineering (CSMR'07), pp. 81-90, Amsterdam, the Netherlands, March 2007.
[10] Brenda S. Baker. Finding Clones with Dup: Analysis of an Experiment. IEEE Transactions on Software Engineering, Vol. 33(9): 608-621, September 2007
[11] Kumar, Anil, et al. "An intrusion identification and prevention for cloud computing: From the perspective of deep learning." Optik 270 (2022): 170044.

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

INTERNATIONAL JOURNAL OF MULTIDISCIPLINARY RESEARCH

IN SCIENCE, ENGINEERING AND TECHNOLOGY

| Mobile No: +91-6381907438 | Whatsapp: +91-6381907438 | ijmrset@gmail.com |

www.ijmrset.com