Volume 5, Issue 3, March 2015 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Defect Fixing and Improving Code Quality-Study

Aswini.G, Suganya.D, Harunya.B
ME II[nd] Year-Software Engineering, Coimbatore Institute of Engineering and Technology,
Narasipuram, Coimbatore, Tamil Nadu, India

Abstract- Defect fixing and improving code quality study emphasis the relation between Software Improvement Group (SIG) quality model and the Test code quality Model. In particular, it has the method of issue churn view to handle the issues in data repository. This study reveals that the Software Improvement Group has the significance of testing maintainability, and the Test code quality model is used to improve the test code quality by completeness, effectiveness and maintainability.

Keywords-Quality, Issue, SIG, Test code, Defect

I. INTRODUCTION

Software testing is the process of evaluation a software item to detect differences between given input and expected output and also to assess the feature of a software item. Testing assesses the quality of the product. Software testing is a process that should be done during the development process. The four level of software testing are unit, integration, system and acceptance testing.

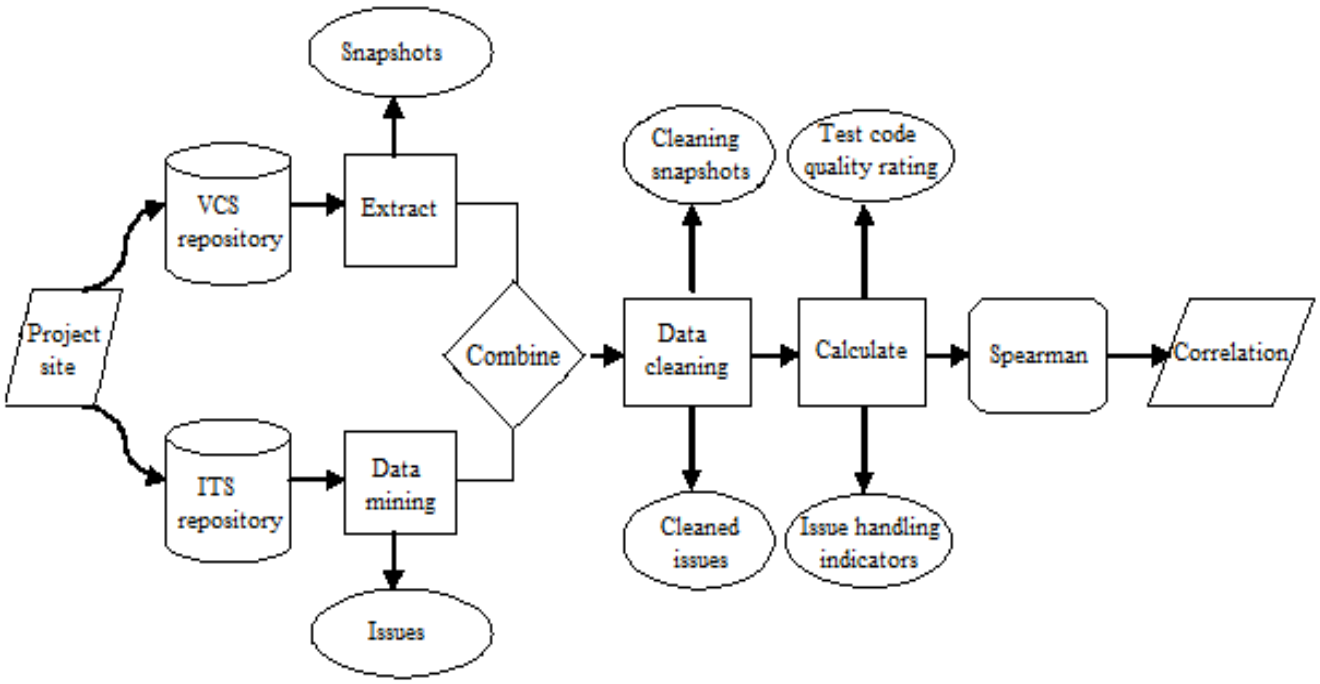


Fig.1 Test Code Quality Model (image taken from [1])

The Unit testing is the process of testing individual unit of the system. The main purpose is to validate that each unit of the software performs as design. Integration testing is the process where the individual units are combined and testing as a group. It also helps to expose fault in the integration between integration units. System testing is the process where a complete integrated system is tested. This testing is for evaluating the system compliance with the specified requirement. Acceptance testing is the process done to verify if system meets the customer requirement.

The Maintainability Model of SIG combines several important source code metrics into a single rating of the maintainability of a software product .The Maintainability Model is used by SIG to determine software product quality in software risk assessment and during software monitoring. The Test code quality has three dimensions namely completeness, effectiveness and maintainability.

The rest of this paper is structured as follows: we have the study about the Test code quality model in section 2. In section 3 we describe about the Software Improvement Group (SIG) quality model in section 4 we describe the related work based on those models and Section 5 tells about the Conclusion and future work about this study.

II. TEST CODE QUALITY MODEL

Dimitrios Athanasiou suggests that the Test code Quality model [1] is based on the source code metrics and it does not require any other source of information. (see Fig 1)The metric of test code quality are Code Coverage, Assertion-McCabe Ratio, Assertion Density, Directness and Maintainability. We already discussed that the test code quality has three dimension , it is based on the quality of the test code. We have the table about the mapping of test code quality model with the properties of its sub-characteristics (see TABLE 1).

Table 1 Mapping of test code quality with the properties of its sub- harateristics

		Test Code Quality		
		Completeness	Effectiviness	Maintainability
Properties	Code Coverage	O		
	Assertion Mc-Cabe ratio	O		
	Assertion Density		O	
	Directioness		O	
	SIG quality model			O

A. *Metrics*

The metrics of the test code quality model were used to design the model at every place and it is taken as the indicators of the test code quality model and we can discuss briefly about the metrics.

1) Code Coverage

Code coverage is the most frequently used metric for test code quality assessment. The code coverage can be measured by calculating the covered lines of code. It is a measure used to describe the degree to which the source code of a program is tested by a particular test suite. The tool which is used to calculate the coverage is Clover which is better then Emma while calculating the code coverage.

2) Assertions-McCabe Ratio

The Assertions-McCabe ratio metric indicates the ratio between the number of the actual points of testing in the test code and of the decision points in the production code. The actual point is the starting point and the decision point is the ending point. It is calculated by the measure of Cyclomatic Complexity and the number of assertion statement in the test code.

$$\text{Assertion- McCabe Ratio} = \frac{\text{Number of assertions}}{\text{Cyclomatic Complexity}}$$

3) Assertion □ □ □ □ □ □ □

The Assertion Density is measuring the ability of the test code to detect defects in the parts of the production code. The indicator for testing effort is test lines of code and with the number of assertions and it is shown below:

$$\text{Assertion Density} = \frac{\text{Number of Assertion}}{LOC_{test}}$$

4) Directness

Directness measures the extent to which the production code is covered directly. And it is measured by the Clover.

5) Maintainability

To measure the maintainability of test code various metrics are used and combined in a model which is based on the SIG quality model. SIG quality model is an operational implementation of the maintainability characteristic of the software quality that is defined in the ISO/IEC 9126.The analyzability, changeability; stability and testing are the aspect of the test code maintainability. [5]

Analyzability is the process of the software product to be diagnosed for the causes of failures in the software to be identified.

Changeability is the process to enable a specified modification to be implemented in the software.

Stability is the process of the software product to avoid unexpected effects from modifications in the software.

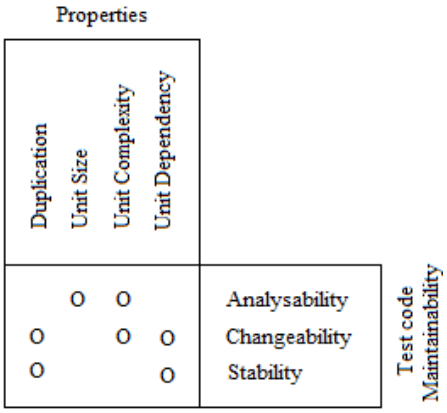
Testability is the process of the software product to enable modification in the software (see TABLE 2)

B. Handling Issue

1) Issue Resolution Time

The issue resolution time is the time taken to solve the defect and the period of time which is taken is known as issue resolution time. Here we use the Issue Tracking System (ITS) as a tool to collect the ITS repository for the mining of data. The Version Control System (VCS) is the process of solving the defect and which is kept as separate version and it makes a VCS repository. To link the Version Control System to Issue Tracking System a fix-induction change identification algorithm is used [3].

Table 2 The test code maintainability as adjusted from sig model



2) Issue Churn View

Issue Churn view is used to perform the high level assessment of the issue handling process and it displays the issue at the monthly basis [8].

Likewise, we first measure the test code quality and the information needed to assess the aspect of the test code. And we use Goal-Question-Metric (GQM [4]) technique to evaluate the quality to the test code. This model combines the metrics and extracts the information for the technical quality of the test code. Then the benchmarking technique is applied to calibrate the model to convert its metrics into quality rating.

Table 3 Thresholds for test code quality metrics

Metrics	Scope	Values
Code coverage (%)	System	92.7
Assertion McCabe Ratio	System	1.826
Assertion density	System	37.5
Directness (%)	System	79.4
Duplication (%)	System	19.6
Unit size	Unit	542
Unit Complexity	Unit	76
Unit dependency	Unit	97

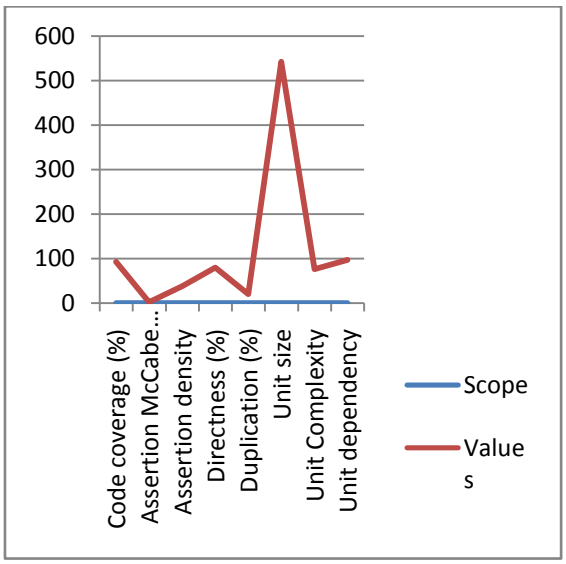


Fig 2. The distribution of the metrics

III. SIG QUALITY MODEL

I. Heitlager suggest that the Software Improvement Group (SIG) quality model is the tool-based consultancy firm and that is specialized in the maintainability process shown in fig 3. This model identifies the source code metric and maps their metrics to the characteristics of the ISO/IEC 9126 and that is related to the maintainability process. The SIG quality model is also known as McCall's Triangle of Quality.

In SIG model the source code metrics were present and they are Volume, Duplication, Unit Complexity, Unit size, Unit interface and Module coupling. The properties are mapped to the ISO/IEC 9126 process.

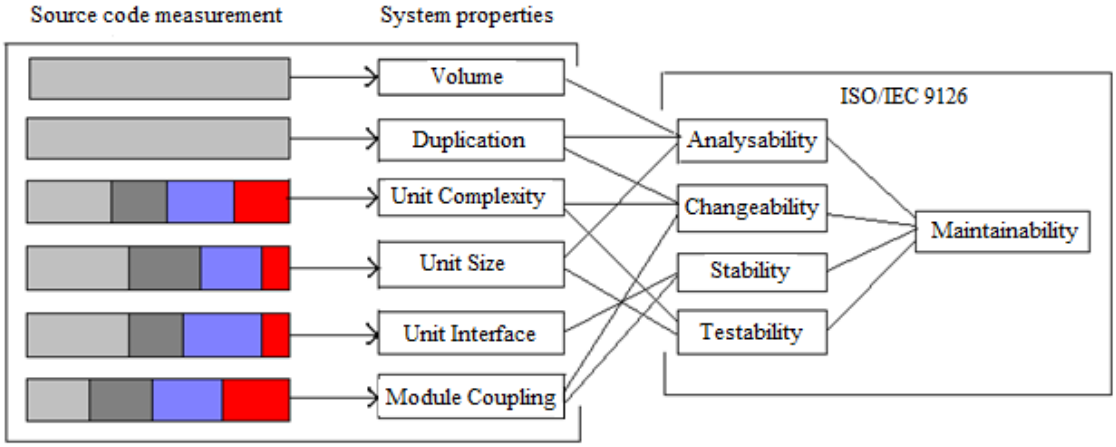


Fig. 3 Software Improvement Group Quality Model (Image taken from [9])

Each of the metrics are mapped to the similar maintainability process. The volume is mapped to the analyzability, Duplication is mapped to the analyzability and changeability, unit complexity is mapped to the changeability and testability, unit size is mapped to the analyzability and testability, unit interface is mapped to the stability of the process and finally module coupling is mapped to the changeability and stability. Likewise, the system properties is mapped to the maintainability process.

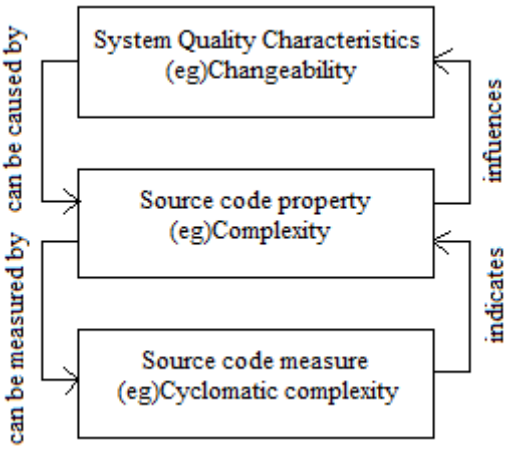


Fig. 4 Mapping of Maintainability Model(Image taken from [2])

Finally, the maintainability process of analyzability, changeability, stability and testability is combined to form the Maintainability model, were the SIG model was formed

The definition of software quality characteristics of ISO quality model provides the frame of reference and standardized terminology which facilitates communication concerning of software quality. The maintainability Index (MI) is the process which is used to measure the performance and calibrate the result with the opinion of the developers.

The maintainability model has to be mapped to the system quality characteristics to the source code measure in two steps shown in Fig 4. Firstly,the system quality characteristics can be caused by the source code properties like complexity .And the source code property can be measured by the source code measure for example cyclomatic complexity, likewise the system quality characteristics is mapped to the source code measurement.

A. Characteristics of Maintainability Process

The maintainability model can be done by the 180/IEC 9126 process. And the model has four characteristics and they are analyzability, changeability, stability and testability. [2]

Analyzability: How easy or difficult to modify the system.
Changeability: How easy or difficult to adapt to the system.
Stability: When a system is modified then the system should be in consistent state.
Testability: When the system is modified it checks whether the system is easy or difficult to test.

IV. RELATED WORK

B.W.Boehm represent the Boehm's quality model [6] and which is the attempts to identify the software quality from a given sets to metrics and attributes. And grady proposed the FURPS quality model and later on it extended by IBM Rational into FURPS+ model and also Dromey's quality model[7].

V. CONCLUSION

The paper focused on the test code quality model and the software improvement group model. The test code quality model is used to improve the quality of the test code by handling the issues and in software quality model to concentrate on the maintainability of the software product.

In future work we expect to refine these model by make another model to improve the source code quality.

REFERENCE

[1] Dimitrios Athanasiou, Ariadi Nugroho, Joost Visser, Andy Zaidman, "*Test Code Quality and its Relation to issue Handling Performance*" IEEE Trans. Softw. Eng .,2014

[2] I. Heitlager, T. Kuipers, and J. Visser, "*A practical model for measuring maintainability*," in Proc. Int'l Conf. on Quality of Information and Communications Technology. IEEE, 2007, pp. 30–39.

[3] J. Sliwerski, T. Zimmermann, and A. Zeller, "*When do changesinduce fixes*?"SIGSOFT Softw. Eng. Notes, vol. 30, no. 4, pp. 1–5,2005.

[4] V. R. Basili, "*Software modeling and measurement: theGoal/Question/Metric paradigm*," Techreport UMIACS TR-92-96, University of Maryland at College Park, College Park, MD,USA, Tech. Rep., 1992.

[5] I. 9126-1:2001, "*Software engineering - product quality - part 1:Quality model*," ISO, Geneva, Switzerland, pp. 1–32, 2001.

[6] B. W. Boehm, J. R. Brown, and M. Lipow, "*Quantitative evaluation of software quality*," in Proc. Int'l Conf. on Software Engineering (ICSE). IEEE Computer Society, 1976, pp. 592–605.

[7] R. G. Dromey, "*A model for software product quality*," IEEE Trans.Softw. Eng., vol. 21, no. 2, pp. 146–162, 1995.

[8] B. Luijten, J. Visser, A.Zaidman,"*Assessment of Issue Handling Efficiency*,"Empir. Software Eng., 2010.

[9] B. Luijten, J. Visser,"*Faster Defect Resolution with Higher Technical Quality of software*", Empir. Software Eng., 2010.

