Expert Rating Based Software Quality Evaluation

[1] Dr. B. B. Jayasingh, [2] Sri Datta Virivinti, [3] N.S. Pranav

[1] *Professor, Dept. of IT, CVR College of Engineering, Vastunagar, Mangalpalli (V), Ibrahimpatan (M), RR District – 501510, India.*

[2] *Department of Information Technology, CVR College of Engineering, Hyderabad, Andhra Pradesh, India.*

[3] *Department of Information Technology, CVR College of Engineering, Hyderabad, Andhra Pradesh, India.*

Abstract

The quality of the software is very essential before the deployment. Quality check of software can be done at any level of the software, be it in the initial or final stages of development. The software should be tested rigorously in order to avoid any future inconvenience. ISO/IEC 9126-1 selects 6 criteria along with 27 sub criteria for determining the quality of the software. The main challenge faced by Software Quality Assurance (SQA) is that it should apply more comprehensive techniques, and decide whether the software is meeting the good standards in terms of quality. The proposed approach is to evaluate the software by the rating given by a group of experts. The ratings are direct rating in the scale of 1 to 9. We calculate the arithmetic mean of all the experts to find the level of quality. We also narrated the calculation of low level metrics of each criterion. It can help the developers to decide whether to go ahead or make any changes in the faulty areas of software.

1. Introduction

There're many standards and metrics to evaluate the quality of software. The research in this field is growing day by day in order to meet the demands of the software managers. The software managers have a great deal of work to ensure that the quality of the software is up to the mark, so they're very concerned about the quality of software [2]. The quality of software is checked at various levels and is tested before the actual usage of it in the company or an organization. According to Nan-Hsing Chiu [3], the software modules are classified into two categories, fault proneness (fp) or non fault proneness and the software experts can concentrate on the fp modules early to prevent poor quality of software. The suggested approach in this paper, can easily give the quality of the software by taking the ratings given by the different experts. The ratings given by the various experts can be helpful in finding the critical areas where the software can be improved, thereby decreasing the damages caused by poor quality of software.

Software quality determination described in this paper is useful for evaluating the software quality and reporting it back to the user. It's very essential to the developer or a vendor to know the quality of the software. The software quality determination process can help in finding the critical areas where the software needs to be improved.

Measurement of the software quality includes the measurement of in-house developments and a selection of vendors' products. SQA takes the responsibility to make the ''go/not go'' decision in this matter. If the quality of the product released or purchased is below standard, the company will suffer a significant loss. If the product development is behind schedule, the company also loses a lot. As it is difficult to build a perfect or error free software system or to purchase highly compatible software components, SQA must apply comprehensive techniques to determine whether the systems reach the right level of quality. These techniques include the clear definition of quality attributes, measurement tools, and the integration framework.

In this paper, the approach is that, the software is evaluated by a group of experts and based on the results the developer can get to know the quality level of the software. The criteria for evaluating the software are discussed in the implementation section.

2. Background

As discussed by Robyn R. Lutz [5], there are many software's which help in functioning of safety critical systems like traffic control, smart vehicles. Software used in defense and nuclear applications also comes under safety critical software. Safety critical software is software which on its failure can cause life loss and huge property damage. This software needs be built and designed up to the software standards to ensure safety. The testing and development of the software needs to meet the software standards set by the software managers. To ensure this, the software quality assurance can help in determining or assessing the quality of the software.

According to Christian Murphy [1], the flaws may show up later when the software is deployed in the system, the various cases considered while testing might not be sufficient to know any errors in it, in-vito testing refers to continuous testing of the software even after deployment. This can be helpful if there are any remaining flaws in the software. The tests done don't affect or alter the state, which is potentially visible to the users. This way the software which is to be deployed in the real time system will have as fewer bugs or faults as possible.

According to Robert Baggen[10] software benchmarking is often associated with the code functionality and the number of lines of that code. This gives only a part of overall quality and neglects the terms of Maintainability which is an important part of the software life cycle. Robert Baggen[10] establishes some metrics that deal with the Maintainability through the SCM[4](Source Code Metrics). These metrics are a combination of two or more metrics low level product metrics like Volume, Duplication, Unit complexity, Unit size, Unit interfacing, Module coupling [10]. With the combination and usage of low level metrics with high level metrics we can find the accurate quality of the software.

[7] The importance of software in our daily life is immense. Almost every field in the present world has the needs of software, for example the automobile industry, which is increasing rapidly day by day and flight control systems use software which is helpful to them. The software's need to be bug free and should be of very good quality in order to avoid any unwanted results in the future. The deployment of such special kind of software needs to be taken extra care. This can be made easy with the assurance of the quality of software is known to the purchaser, where he can decide whether to buy the software or not. The developer work can also be known with the software quality, if the quality of the software is good, then the developer has done a good job, which means that the software can be deployed in the real time system. If software quality is below the standards, necessary changes can be made to the software and tested again for good quality.

The main challenges faced by organizations is that how to we ensure the quality in Agile [8], the most traditional practices which have proven good in the past are no longer useful for the Agile environment. The main challenge faced by Software Quality Assurance is that it should apply more comprehensive techniques, and decide whether the software is meeting the good standards in terms of quality. [9]The defect density of the software is generally measured after the implementation of the project, but it's argued that an early warning of the defect levels in the system will help in the development of software and will be helpful for better defect management strategies.

3. Analysis

The proposed approach uses six criteria and 27 sub-criteria, which are defined in ISO/IEC 9126-1, 2001[11]. The evaluator considers the sub characteristics of six criteria by calculating the weights and applying Analytical Hierarchy Process (AHP) [12] before given the direct ratings.

3.1. Direct Rating

Table 1. Direct rating criteria

Criteria	Sub Criteria
Functionality	Suitability, Accuracy, Interoperability, Security, Functionality compliance
Reliability	Maturity, Recoverability, Fault tolerance, Reliability compliance
Usability	Understandability, Learnability, Operability, Attractiveness, Usability compliance
Efficiency	Time behaviour, Resource behaviour, Efficiency compliance
Maintainability	Analyzability, Changeability, Stability, Testability, Maintainability compliance
Portability	Adaptability, Installability, Co-existence, Replaceability, Portability compliance

The direct rating uses 6 criteria [11] and 27 sub-criteria to rate the software, by which the quality of the software can be known.

Functionality: The ability of the software to meet the required needs when in use under specific conditions is called as Functionality of the software. The attributes regarding to this criteria are Suitability, Accuracy, Interoperability, Security, Functionality compliance.

Reliability: The ability of the software to maintain the specified level of performance when used under specific conditions is called Reliability. Maturity, Recoverability, Fault tolerance, Reliability compliance is the attributes relating to it.

Usability: The software is measured on how easy it is to understand, learn and use it under the required conditions for its usability. Understandability, Learnability, Operability, Attractiveness, Usability are the attributes regarding Usability.

Efficiency: The correctness or the accurate results of the software product is taken into consideration for measuring efficiency. Time behavior, Resource behaviour, Efficiency compliance are the three attributes relating to it.

Maintainability: How easy it is to maintain a software, that is, how is easy it is to make any changes etc are taken into consideration for measuring this criteria. Analyzability, Changeability, Stability, Testability, Maintainability compliance are the attributes relating to it.

Portability: How easy software is to deploy or install in other systems, how easy the software is to adapt to the required environment etc comes under Portability. Adaptability, Installability, Co-existence, Replaceability, Portability compliance are the attributes which come under it.

The rating depends on six criteria with 27 sub criteria in ISO/IEC 9126-1 (2001)[11], which is the revision of 1991 version (ISO/IEC 9126, 1991). The direct rating is given on the basis of the above criteria, and is given as" poor, weak, good, satisfactory, excellent", by the judgment of experts.

The rating depends on six criteria with 27 sub criteria in ISO/IEC 9126-1 (2001)[11], which is the revision of 1991 version (ISO/IEC 9126, 1991). The direct rating is given on the basis of the above criteria, and is given as" poor, weak, good, satisfactory, excellent", by the judgment of experts.

3.2. Source Code Metrics

The Source Code Metrics is also considered for rating, which is given as "good, average, poor" depending on the software. The experts give the source code metric rating by assessing the software quality and answering the different questions provided to rate the software. These criteria's for source code metrics are discussed by P´eter Heged [4]**.** The following are the 5 categories [4] discussed and are organized into 5 questions:

• *Analyzability* - how easy it is to diagnose the system for deficiencies or to identify where to make a change?

• *Changeability* - how easy it is to make a change in the system?

• *Stability* - how well does the system avoid unexpected effects after a change?

• *Testability* - how easy it is to validate the software after a change?

• *Comprehension* - how easy it is to comprehend the source code of a method?

3.3. Calculating weights of sub criteria

Software quality depends on low level as well as high level metrics. Ratings can't be solely given based on high level metrics as this can be accurate. Code quality in software can be calculated using four characteristics and their respective sub characteristics [12] selected from ISO/IEC-9126. They are functionality, efficiency, maintainability, portability and their respective sub characteristics. The source code attributes that have an effect on ISO/IEC-9126 characteristics are Volume, Complexity, Abstraction, Encapsulation, Coupling, Cohesion, Messaging, Polymorphism, Composition, and Inheritance. Weights are determined by applying AHP at low level, intermediate level and high level to evaluate sub characteristics using the source code attributes and pair wise comparison table for each is constructed. The values for each entity for the ISO/IEC-9126 quality characteristics are calculated [12] using the following utility function U(Ci):

$U(C_i) = v(sc_1)*w(sc_{1i}) + v(sc_2)*w(sc_{2i}) + ...+ v(sc_n)*w(sc_{ni})$ (Equation 1) where

$v(sc_i) = v(d_1)*w(d_{1i}) + v(d_2)*w(d_{2i}) + ...+ v(d_n)*w(d_{ni})$(Equation 2)

$v(d_i) = v(m_1)*w(m_{1i}) + v(m_2)*w(m_{2i}) + ...+ v(m_n)*w(m_{ni})$(Equation 3)

$U(C_i)$ = Utility Function of ISO/IEC-9126 characteristic I

$v(sc_i)$ = Value of Sub-characteristic j

$w(sc_{ji})$ = Weight of Sub-characteristic j for ISO/IEC-9126 Characteristic i

$v(d_i)$ = Value of Source Code Attribute d_i

$w(d_{ji})$ = Weight of Source Code Attribute d_{ji} for Sub - Characteristic i

$v(m_i)$ = Value of Metric m_i

$w(m_{ji})$ = Weight of Metric m_j for Attribute i

Sub criteria like time behavior (under the criteria "efficiency") are difficult to determine. It can vary on a number of factors like the processor speed on which the software is being tested etc. Such kind of attributes must be calculated at run time. The method used in evaluating the rating is arithmetic mean. For instance, for evaluating the rating for criteria 1 arithmetic mean of the sub criteria is considered, that is (1.1 + 1.2 + 1.3 + 1.4 +1.5)/ 5. The rating for each criteria is measured and then the arithmetic mean of the all the six criteria is taken for giving the overall rating of the software for one expert.

If there are more number of experts, then rating is evaluated considering each individual expert and then arithmetic means of the rating obtained from different experts is calculated. Consider the following analysis given where there are 3 experts.

Expert 1 rating = (Criteria 1 rating +criteria 2 rating +criteria 3 rating+ criteria 4 rating + criteria 5 rating + criteria 6 rating) / 6

Each criterion is calculated based on its sub criteria rating, and arithmetic mean is obtained for total criteria to get rating given by one expert.

Similarly the rating is calculated for Expert 2 and Expert 3.To get the overall rating of the software given by three experts arithmetic mean of three expert ratings is to be evaluated in the following manner,

Overall rating = (Expert 1 rating + Expert 2 rating + Expert 3 rating) / 3

The rating scale is given in the below table 2 for Direct ratings,

Table 2. Rating scale for direct ratings

Poor	1
Weak	3
Good	5
Satisfactory	7
Excellent	9

If the rating of the software is below good, then we can say that the software is not up to the standards and it needs revision.

The rating scale for Source Code is given in below table 3,

Table 3. Rating scale for source code metrics

Poor	1
Average	3
Good	5

4. Implementation

This system is designed for the software engineering department who determines the quality of a module or complete software. The users can upload their code to the system and can view the ratings given by different experts with their quality level. If the quality level more than acceptable limit then the product can be released to the market otherwise the user has to redesign their code.

An expert who's registered can log into his account and rate the software. The registration of the expert is done taking account of the work experience, in which field he is an expert, etc which are helpful in rating and assuring the quality of the software. The developer who wants to know the ratings of the software given by various experts can check the ratings given by different experts. Here, the developer can know the quality of the software based on the ratings given by the group of experts.

There are two kinds of login i.e. user and expert. User and expert both provide the username and password before entering into the system.

User Login

User provides his details to the system like username and password. The system verifies the username and password in the database before using the system. If anyone among them is wrong the system prompts with a message to retype the same. Whenever there is match with the database the system provides with a page to upload the code and give the destination path. User also can see his previous uploaded codes and user can view the ratings. Users are not permitted to give ratings; it is protected by the system. Each uploaded code has a unique ID called CID and each user has a unique ID called UID.

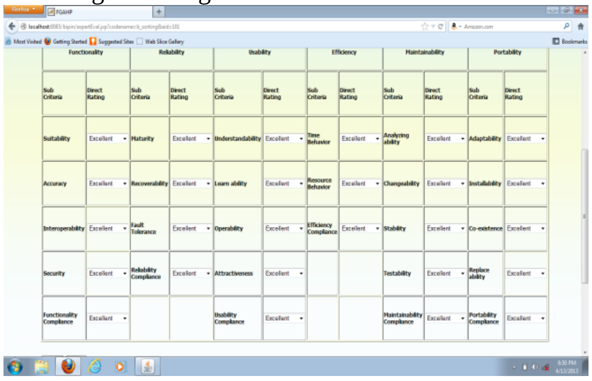
Expert Login

Expert provides his details to the system like username and password. The system verifies the username and password in the database before using the system. If anyone among them is wrong the system prompts with a message to retype the same. Whenever there is match with the database the system provides with a page to select the code to give ratings. Experts are not permitted to upload codes. Experts are not permitted to view the ratings of other experts as well as to view the final ratings though it is protected by the system. Each expert has a unique ID called UID. Once expert has given rating for one code with a model his second rating for the same are not permitted by the system.

Direct Rating

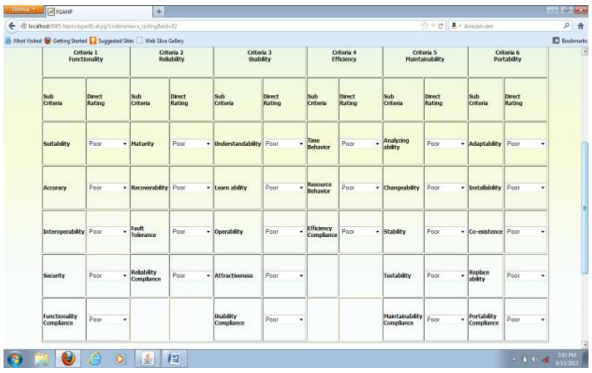
After clicking the direct rating button the expert provided with an input form that contains six criteria and twenty seven sub criteria. The direct rating contains five scale values i.e. Poor, Weak, Good, Satisfactory and Excellent. After filling the form expert has to click submit button. Whenever there is new code to provide rating he has to login and do the same.

The screen shots of the direct ratings given by 3 experts is given below in the figure 1, 2, 3 respectively and the result is given in figure 4



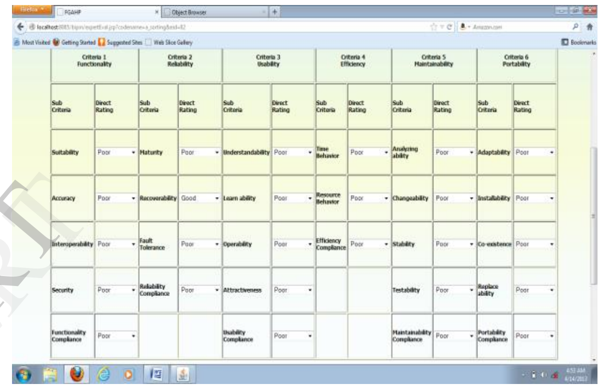
Functionality		Reliability		Usability		Efficiency		Maintainability		Portability	
Sub Criteria	Direct Rating	Sub Criteria	Direct Rating	Sub Criteria	Direct Rating	Sub Criteria	Direct Rating	Sub Criteria	Direct Rating	Sub Criteria	Direct Rating
Suitability	Excellent	Maturity	Excellent	Understandability	Excellent	Time Behavior	Excellent	Analyzing ability	Excellent	Adaptability	Excellent
Accuracy	Excellent	Recoverability	Excellent	Learn ability	Excellent	Resource Behavior	Excellent	Changeability	Excellent	Installability	Excellent
Interoperability	Excellent	Fault Tolerance	Excellent	Operability	Excellent	Efficiency Compliance	Excellent	Stability	Excellent	Co-existence	Excellent
Security	Excellent	Reliability Compliance	Excellent	Attractiveness	Excellent			Testability	Excellent	Replace ability	Excellent
Functionality Compliance	Excellent			Usability Compliance	Excellent			Maintainability Compliance	Excellent	Portability Compliance	Excellent

Fig. 1 Screenshot of direct rating expert1 i.e. all excellent



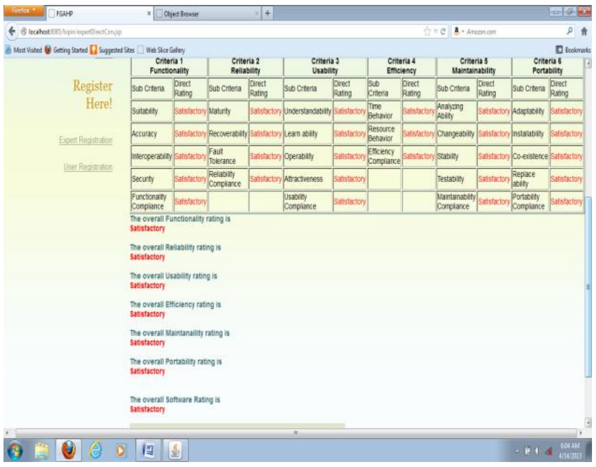
Criteria 1 Functionality		Criteria 2 Reliability		Criteria 3 Usability		Criteria 4 Efficiency		Criteria 5 Maintainability		Criteria 6 Portability	
Sub Criteria	Direct Rating	Sub Criteria	Direct Rating	Sub Criteria	Direct Rating	Sub Criteria	Direct Rating	Sub Criteria	Direct Rating	Sub Criteria	Direct Rating
Suitability	Poor	Maturity	Poor	Understandability	Poor	Time Behavior	Poor	Analyzing ability	Poor	Adaptability	Poor
Accuracy	Poor	Recoverability	Poor	Learn ability	Poor	Resource Behavior	Poor	Changeability	Poor	Installability	Poor
Interoperability	Poor	Fault Tolerance	Poor	Operability	Poor	Efficiency Compliance	Poor	Stability	Poor	Co-existence	Poor
Security	Poor	Reliability Compliance	Poor	Attractiveness	Poor			Testability	Poor	Replace ability	Poor
Functionality Compliance	Poor			Usability Compliance	Poor			Maintainability Compliance	Poor	Portability Compliance	Poor

Fig. 2 Screenshot of direct rating expert2 i.e. all poor



Criteria 1 Functionality		Criteria 2 Reliability		Criteria 3 Usability		Criteria 4 Efficiency		Criteria 5 Maintainability		Criteria 6 Portability	
Sub Criteria	Direct Rating	Sub Criteria	Direct Rating	Sub Criteria	Direct Rating	Sub Criteria	Direct Rating	Sub Criteria	Direct Rating	Sub Criteria	Direct Rating
Suitability	Poor	Maturity	Poor	Understandability	Poor	Time Behavior	Poor	Analyzing ability	Poor	Adaptability	Poor
Accuracy	Poor	Recoverability	Good	Learn ability	Poor	Resource Behavior	Poor	Changeability	Poor	Installability	Poor
Interoperability	Poor	Fault Tolerance	Poor	Operability	Poor	Efficiency Compliance	Poor	Stability	Poor	Co-existence	Poor
Security	Poor	Reliability Compliance	Poor	Attractiveness	Poor			Testability	Poor	Replace ability	Poor
Functionality Compliance	Poor			Usability Compliance	Poor			Maintainability Compliance	Poor	Portability Compliance	Poor

Fig. 3 Screenshot of direct rating expert3 i.e. all poor and one good



Criteria 1 Functionality		Criteria 2 Reliability		Criteria 3 Usability		Criteria 4 Efficiency		Criteria 5 Maintainability		Criteria 6 Portability	
Sub Criteria	Direct Rating	Sub Criteria	Direct Rating	Sub Criteria	Direct Rating	Sub Criteria	Direct Rating	Sub Criteria	Direct Rating	Sub Criteria	Direct Rating
Suitability	Satisfactory	Maturity	Satisfactory	Understandability	Satisfactory	Time Behavior	Satisfactory	Analyzing Ability	Satisfactory	Adaptability	Satisfactory
Accuracy	Satisfactory	Recoverability	Satisfactory	Learn ability	Satisfactory	Resource Behavior	Satisfactory	Changeability	Satisfactory	Installability	Satisfactory
Interoperability	Satisfactory	Fault Tolerance	Satisfactory	Operability	Satisfactory	Efficiency Compliance	Satisfactory	Stability	Satisfactory	Co-existence	Satisfactory
Security	Satisfactory	Reliability Compliance	Satisfactory	Attractiveness	Satisfactory			Testability	Satisfactory	Replace ability	Satisfactory
Functionality Compliance	Satisfactory			Usability Compliance	Satisfactory			Maintainability Compliance	Satisfactory	Portability Compliance	Satisfactory

The overall Functionality rating is
Satisfactory

The overall Reliability rating is
Satisfactory

The overall Usability rating is
Satisfactory

The overall Efficiency rating is
Satisfactory

The overall Maintanaility rating is
Satisfactory

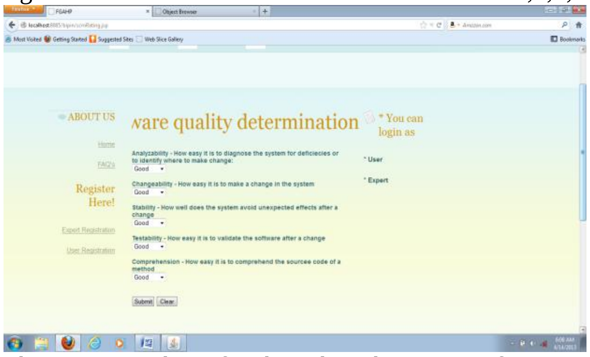
The overall Portability rating is
Satisfactory

The overall Software Rating is
Satisfactory

Fig. 4 Screenshot of direct rating given by all experts i.e. satisfactory.

If the overall rating of the software is satisfactory, this can be inferred from the figure. The arithmetic mean of the three experts is the overall rating for the software rated in the figure. The expert needs to rate each sub criteria, he can't leave any of the criteria or sub criteria as it may not give accurate rating of the software. After rating each criterion and their respective sub criteria are rated, the rating of the software is calculated using arithmetic as discussed in "Analysis" section. We can infer from the figure that the quality of the software is satisfactory; hence we can say that the software is of acceptable quality.

The screenshots of Source Code rating are given in figures 5,6,7,8



ware quality determination

Analyzability - How easy it is to diagnose the system for deficiecies or to identify where to make change:
Good

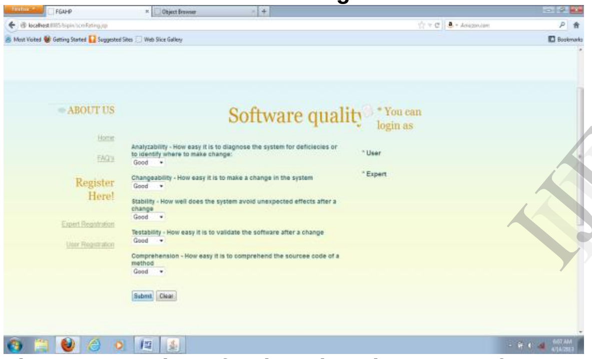
Changeability - How easy it is to make a change in the system
Good

Stability - How well does the system avoid unexpected effects after a change
Good

Testability - How easy it is to validate the software after a change
Good

Comprehension - How easy it is to comprehend the sourcee code of a method
Good

Fig. 5 Screenshot of rating given by expert1 for SCM model i.e. all good.



Software quality

Analyzability - How easy it is to diagnose the system for deficiecies or to identify where to make change:
Good

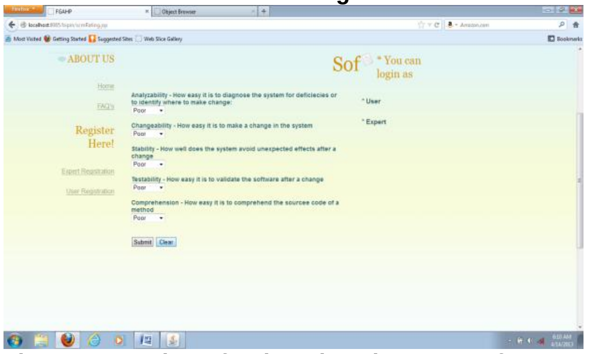
Changeability - How easy it is to make a change in the system
Good

Stability - How well does the system avoid unexpected effects after a change
Good

Testability - How easy it is to validate the software after a change
Good

Comprehension - How easy it is to comprehend the sourcee code of a method
Good

Fig. 6 Screenshot of rating given by expert2 for SCM model i.e. all good.



Sof

Analyzability - How easy it is to diagnose the system for deficiecies or to identify where to make change:
Poor

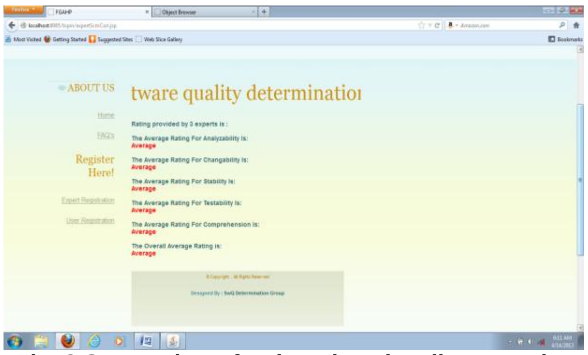
Changeability - How easy it is to make a change in the system
Poor

Stability - How well does the system avoid unexpected effects after a change
Poor

Testability - How easy it is to validate the software after a change
Poor

Comprehension - How easy it is to comprehend the sourcee code of a method
Poor

Fig. 7 Screenshot of rating given by expert3 for SCM model i.e. all poor



tware quality determinatio

Rating provided by 3 experts is :

The Average Rating For Analyzability Is:
Average

The Average Rating For Changability Is:
Average

The Average Rating For Stability Is:
Average

The Average Rating For Testability Is:
Average

The Average Rating For Comprehension Is:
Average

The Overall Average Rating Is:
Average

© Copyright . All Rights Reserved

Designed By : SwQ Determination Group

Fig. 8 Screenshot of rating given by all experts i.e. average

The overall rating given by three experts for Source Code is given in the figure 8. This can be calculated using the arithmetic mean as discussed in "Analysis" section. Since the quality of the software is average we can say that the quality of the software is acceptable.

5. Conclusion

The system is developed to help the developer to know the level of quality of the software before release to the market. The software is evaluated by the domain experts to maintain its effectiveness. We have shown the result by taking care of all possible cases where experts are not biased to provide the ratings. Thus software quality assurance helps reducing a great deal of bugs, which in turn can help a developer save a lot of money which is used to find the bugs.

6. Acknowledgement

The authors would like to thank CVR College of Engineering, Hyderabad (AP), India for providing its amenities.

7. References

[1] Christian Murphy, Gail Kaiser, Ian Vo, Matt Chu, Quality Assurance of Software Applications Using the In Vivo Testing Approach, 2009 International Conference on Software Testing Verification and Validation, IEEE DOI 10.1109/ICST.2009.18, 2009, pp. 111-120.
[2] Kevin Kam Fung Yuen, Henry C.W. Lau, A fuzzy group analytical hierarchy process approach for software quality assurance management: Fuzzy logarithmic least squares method, Expert Systems with Applications 38, ELSEVIER Publication, 2011, pp. 10292–10302.
[3]Nan-Hsing Chiu, Combining techniques for software quality classification: An integrated decision network approach, Expert Systems with Applications 38, ELSEVIER Publication, 2011, pp. 4618–4625.
[4] P´eter Heged˝us, Tibor Bakota, L´aszl´o Ill´es, Gergely Lad´anyi, Rudolf Ferenc, and Tibor Gyim´othy, Source Code Metrics and Maintainability:A Case Study, ASEA/DRBC/EL

2011, Springer-Verlag Berlin Heidelberg 2011, CCIS 257, pp. 272–284.
[5] R. R. Lutz. Software engineering for safety: a roadmap. In ICSE ’00: Proceedings of the Conference on The Future of Software Engineering, pages 213–226, New York, NY, USA, 2000. ACM.
[6] L. Osterweil. Perpetually testing software. In The Ninth International Software Quality Week (QW’96), May 1996.
[7] M. Lyu: Software Reliability Engineering: A Roadmap, in Future of Software Engineering 2007, L. Briand and A. Wolf (eds), IEEE-CS Press, 2007.
[8] P. McBreen, “Quality Assurance and Testing in Agile Projects”, McBreen.Consulting, 2003
[9] T. R. Gopalakrishnan Nair, R. Selvarani, Defect proneness estimation and feedback approach for software design quality improvement, Information and Software Technology 54, ELSEVIER Publication, 2012, pp. 274–285
[10] Robert Baggen, Jose´ Pedro Correia, Katrin Schill, Joost Visser, Standardized code quality benchmarking for improving software maintainability, Software Qual J, Springer Pub., DOI 10.1007/s11219-011-9144-9, 2012, pp. 287–307.
[11] ISO/IEC 9126-1. (2001). Software engineering-product quality – Part 1: Quality model.
[12] Yiannis Kanellopoulos, Panos Antonellis, Dimitris Antoniou, Christos Makris, Evangelos Theodoridis, Christos Tjortjis, Nikos Tsirakis, Code Quality Evaluation Methodology using the ISO/IEC 9126 Standard, International Journal of Software Engineering & Applications (IJSEA), Vol.1, No.3, July 2010.