



## **Tools, Strategies & Models for Incorporating Software Quality Assurance in Risk Oriented Testing**

**VINITA MALIK<sup>1\*</sup> and SUKHDIP SINGH<sup>2</sup>**

<sup>1</sup>Information Scientist, Central University of Haryana, Mahendergarh, Haryana-123029, India.

<sup>2</sup>Assistant Professor, Deenbandhu Chhoturam University of Science and Technology, Murthal, Haryana-131039, India.

### **Abstract**

Evolution of software is cumbersome process and also needs many iterations of software testing for satisfying some quality criteria. Software quality assurance activities must be effectively used for the proper software quality management and to achieve good product quality. Effective quality management is related to Value Engineering and Risk Management. In the present paper we will study relevance of quality assurance tools, strategies and models while doing risk based testing for the proper product function orientation. By analysing risks we can get to know how much we need to do the testing of software and further can assure the software quality.



### **Article History**

Received: 19 June 2017  
Accepted: 05 July 2017

### **Keywords**

Risk Based Testing,  
Quality, Software Quality Assurance,  
Quality Models.

### **Introduction**


Software projects can get delayed due to improper definition of evolutionary scope, unplanned and untracked project resources, unclear roles and responsibilities and lack of software quality assurance by auditing and reviewing. Software quality management is an ongoing process to establish a kind of environment that can allow middle and top management for jointly facilitating software testing which is again evolutionary in nature<sup>1</sup>. Before releasing software there can be various software quality assurance activities which involve code inspection, review and software testing. The main aim of these activities is to detect and fix all the defects before they get converted into

failures. It is one of the most crucial activities by the project manager to allocate the resources for software quality assurance to make the system failure free<sup>2</sup>.

This research aims to focus on various tools, methodologies and models related to quality assurance in the domain of risk based testing. As while testing the software or before testing all risks must be mitigated by proper assessment and that only can lead to software quality improvement. Number of remaining defects can be one of the effective measure for evaluation of QA prioritization<sup>2</sup>. For maintaining quality of the system we should be able to dynamically update the level of non reviewed

**CONTACT** Vinita Malik ✉ [is@cuh.ac.in](mailto:is@cuh.ac.in) 📍 Information Scientist, Central University of Haryana, Mahendergarh, Haryana-123029, India.

© 2017 The Author(s). Published by Enviro Research Publishers

This is an  Open Access article licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License (<https://creativecommons.org/licenses/by-nc-sa/4.0/>), which permits unrestricted NonCommercial use, distribution, and reproduction in any medium, provided the original work is properly cited.

To link to this article: <http://dx.doi.org/10.13005/ojcs/10.03.08>

files dependent on co fixes done while doing review of previous files .All the software artifacts i.e. classes or methods or components can be assigned for proper investigation of quality assurance. Software quality.

prediction rarely considers the defects impact while provision of recommendation of software location that can be addressed. Such predictions are need of hour as they can inform about the issues in the software beforehand and help in improving the quality of software.

Various search engines i.e. Elsevier, Springer, IEEE, Google Scholar, Taylor and Francis have been explored for collecting data related to quality assurance, quality models and risk oriented testing.

The research has been organized into various sections. In section 2 we will go through various aspects of software quality, software quality assurance, tools, techniques and models for risk oriented testing of software and section 3 we conclude our research by discussing about the future scope .

#### various concepts related to software quality assurance and risk based testing

##### Quality

The IEEE/ISO 24765 defines the quality as degree to which various components of system meets the requirements. It is also the ability of system to

meet the user requirements or we can define it as conformity to expectations of customer<sup>3</sup>. Software quality factors is defined as management oriented software attribute which contributes to quality. These are attributes that affect a component or service quality. Software quality is both process and product oriented<sup>4</sup> as shown in the figure1 below:

Software product is tested by an independent test centre and development is accompanied by preventive assurance measures. According to ISO 9126<sup>4</sup> software quality characteristics can be as follows as shown in the figure 2 given below:

**Various components are considered for system under test and these components can be shown as in figure 3.**

##### Quality Assurance

System quality is assured when quality risks are reduced by performing various activities like reviews, model checking and inspection which consumes project 30% to 90%. So right planning and control can contribute to quality assurance. The planned activities must be effective enough to handle the risk. So the indicators for the expected number of defects are required for planned QA activities<sup>5</sup>.

Quality assurance is a part of software quality management where the later is an umbrella of all processes that software services and life cycle implementations meet quality objectives<sup>3</sup>. SQM comprises of three main categories which can be quality planning , assurance and control . Software

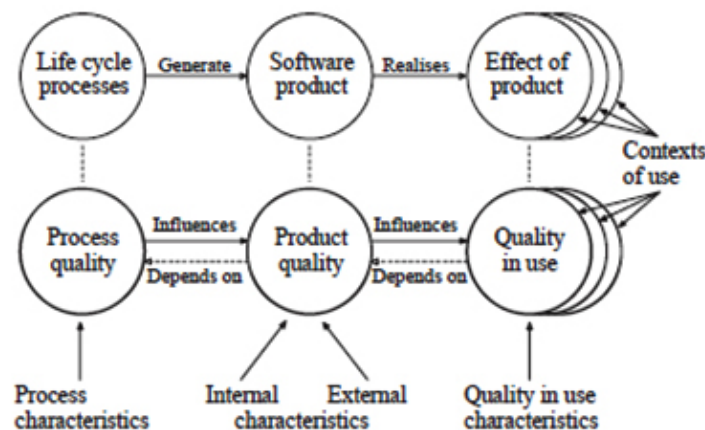


Fig. 1: process and product oriented software quality



Fig. 2: ISO 9126 software quality characteristics

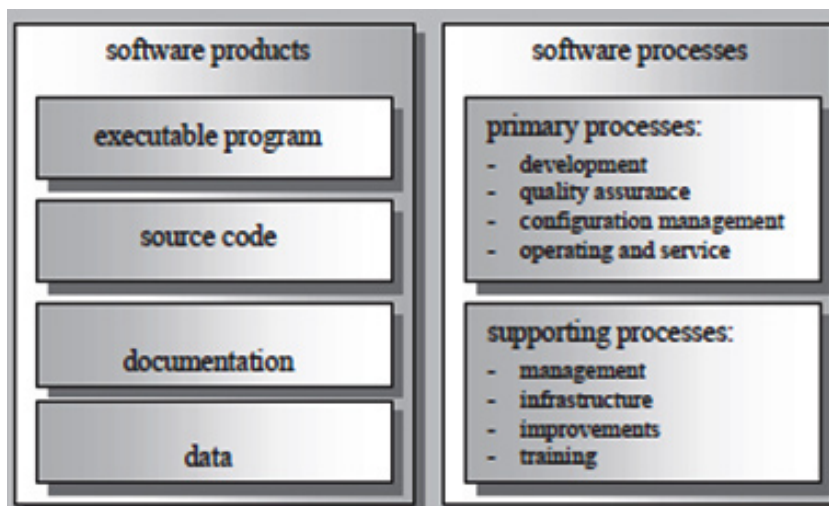


Fig. 3: Software components under test

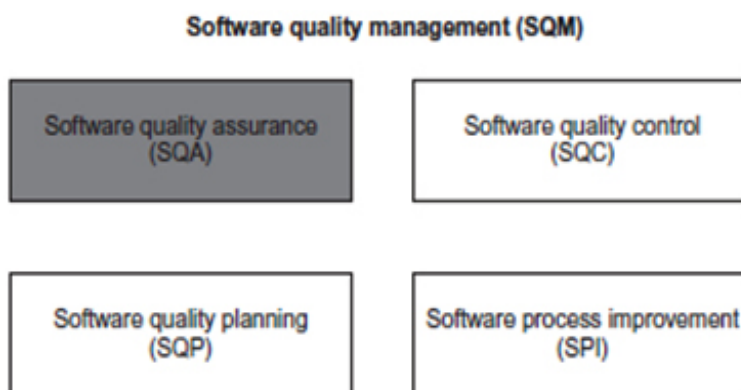


Fig. 4: SQA a part of SQM

quality assurance is basically a quality guide which is independent of a particular project. It is basically a set of standards or regulations to verify and conform the product characteristics. In the following figure 4 SQA can be seen as a part of SQM.

Quantitative measure for quality is defined and it tells the degree by which an item can have a quality attribute.

Conceptual model for assuring software quality considers a software quality framework which is composed of software quality factors that further includes software quality criteria.

Software quality metric is composed of various metrics and these can be product, process and

project metrics. Measurements are used to measure the software quality metric. Following in the figure 5 it has been depicted conceptually as shown.

#### Tools used for SQA

For assuring the software quality the tool support is required.

Various tools has been used in the past. For failure pattern classification FOCUST was developed that included the relations to roles and defect classes. The fault section and the failure section can be selected. By listing the matching possibilities can be considered by the given input. We can develop scenarios for user as well as tester<sup>6</sup>. A popular language for risk based analysis is TTCN-3 by which

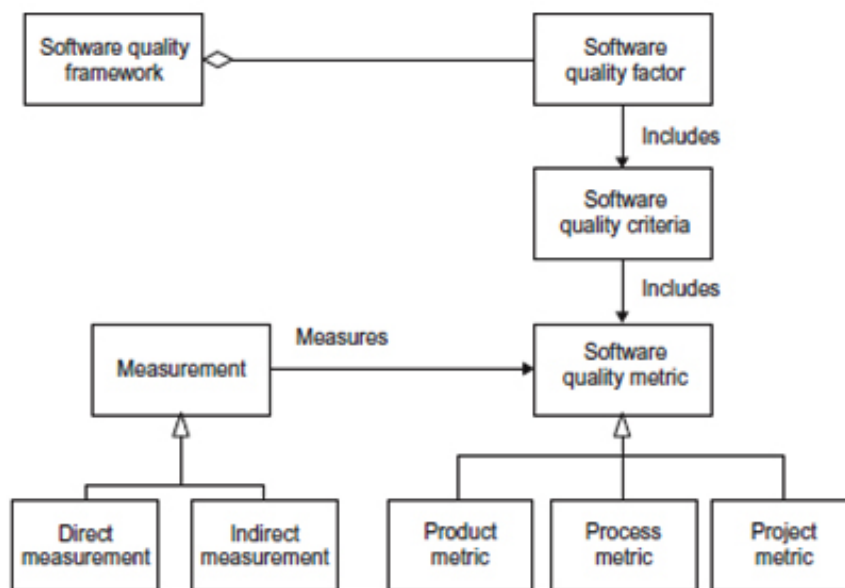


Fig. 5: Conceptual model for software quality assurance



Fig. 6: The module for jLearn

we can use the tree and tabular notations based on programming languages. It has been used in a variety of systems in embedded systems and web application system. A framework named as jABC can be used as a modelling framework for automata learning. This framework is used for service oriented design and uses the extreme model driven design. The reusable components can be used into graph structures which are well defined in form<sup>7</sup>. Clone detection which has been a technique for software quality analysis uses the ConQAT for inspection of analysis metrics and also provides a view for

comparison of files having clone instances which further helps in recommending corrective actions. It helps in architecture conformance analysis too. For bug detection we can use Find Bugs 1.3.9 for java based systems<sup>8</sup>. j Learn is another tool which is used for integration of test development and SQA activities. This tool has various views for programming and testing<sup>9</sup>. The instructional module for it is given in the figure 6 as shown below:

X Query based analysis framework which is very flexible in nature can support the automated analysis of software artefacts. Such framework is language independent and easily adaptable in nature. Here query description is done on a very high level of abstraction so that more analysis rules can be easily added. New types of patterns can be extended easily. Query patterns are independent from domain specific languages<sup>10</sup>.

### Methodologies for Quality Assurance

Quality assurance in embedded domain can be optimized with the help of metrics. Different analysis and testing techniques can be used to optimize the overall quality of the software. Such integrated quality assurance methodology can help in providing a basis for integration of various static and dynamic quality assurance and to generate the knowledge between integrated techniques for further improvement of quality assurance<sup>11</sup>. Such technique includes following steps:

1. Definition of objective and context
2. Calibration of integrated quality assurance approach :
  - Select the quality assurance technique
  - Definition of type of integration
  - Selection of data
  - Definition of assumptions and selection rules

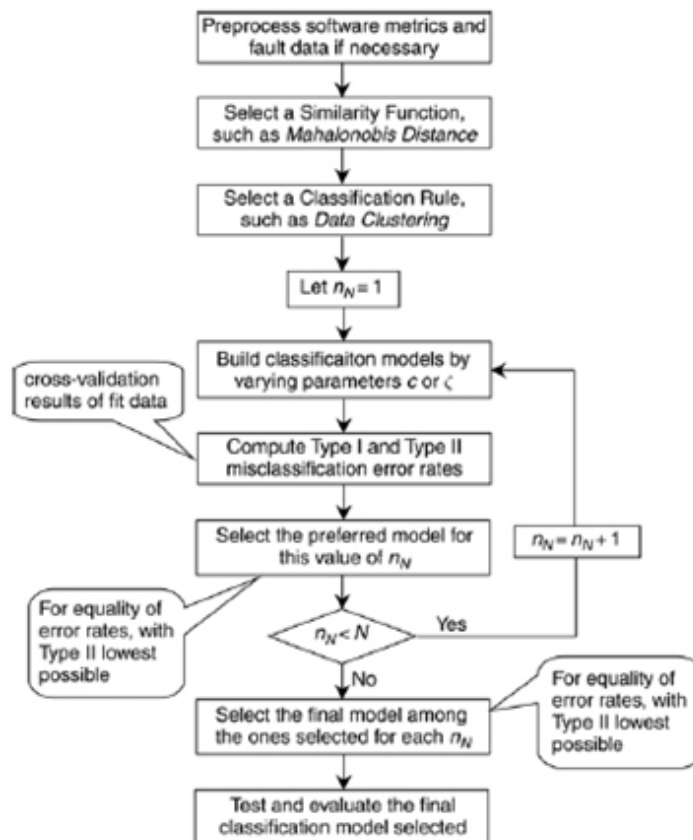


Fig. 7: Analogy based classification for SQA

- Prioritization and evaluation of assumptions and selection rules
- Packaging

### Applying integrated quality assurance approach

Cooperative agent based approach can be used for quality assurance and web software testing to support development and software maintenance activities<sup>12</sup>. Software services agents can be used for software development in evolutionary manner. Management agents are used for managing the services agents which can be used for task scheduling, monitoring these agents states. Various agents co exist with the software throughout the life cycle.

One more method named as QATAM (Quality Assurance Trade off Analysis Method) which helps in identification of candidate QA strategy and then

evaluate this strategy in the given project context<sup>13</sup>. First the context and scope is defined and then QA method repositories are build up and then QA strategy development is done and finally QA Strategy evaluation and selection is done. Quality profile building can be another basis for user based testing. It can be a strategy matrix which shows expression of selected quality characteristics. In the strategy matrix it is done by concept of relative importance. In the determination process they stress on through communication with users<sup>14</sup>.

Analogy based classification systems has been used in the past which aims to find out the solution for a given problem based on previous experience represented by cases in a case library. Such type of system acts as quality classification model before unit testing and various system operations. Threshold value of quality factor which is taken as number of faults define whether the module is fault

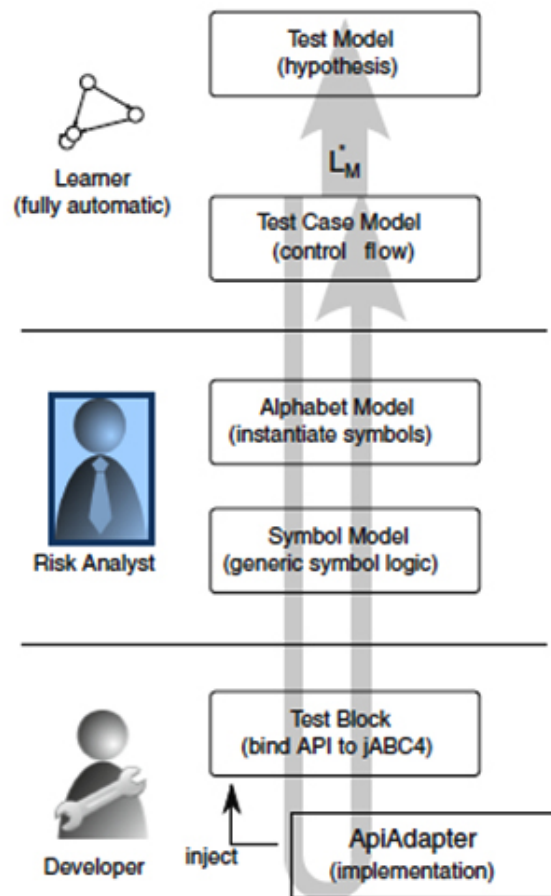


Fig. 8: Layers of ACQC



prone or not. Here similarity is defined in terms of case attributes<sup>15</sup>.

The methodology flow chart is as given below in the figure 7. Another Methodology for SQA that has been used in data migration projects is described as follows<sup>16</sup>:

- First we require an external data migration team
- Define project Scoping
- Application of a data migration platform
- Analysis and cleansing of data
- Data Migration in an incremental form

Collective Intelligence based quality assurance includes advantages of FMEA and inspection which helps in early defect prediction and making information explicit by using checklists and outcomes of inspection<sup>17</sup>. In one more approach for QA for mobile applications we use FIT4Apps methodology

which combines testing and inspection for assuring quality<sup>18</sup>. Risk based testing and continuous quality control can be implemented by alphabetic models used for risk coverage. Various modelling layers for active quality control are given in the Figure 8:

### Models for Quality Assurance

Firstly, Mc Call introduced the concept of software quality model used for US Air Force system development.

It considered all software quality models from the user and customer perspective [8]. It considered three factors as most important which were:

- Product Revision
- Product Transition
- Product Operations

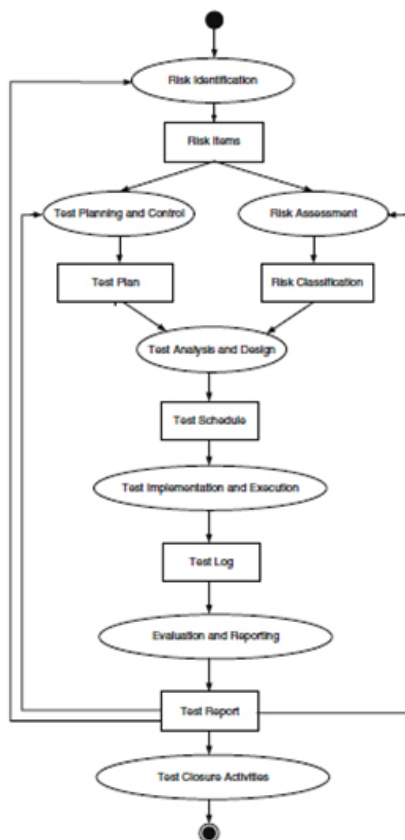
ISO/IEC 25010 software quality standards were used to categorize product quality into characteristics. Deissenbock et al. approached the software quality models and talked about software quality measurement and divided the quality models based on purpose. ISO/IEC 25010 standards became operational by QuaMoco quality model. In the QuaMoco model various static measures were applied which were collected by ASA tools or by reviews manually. The model was divided into modules in which one was root and others were technology specific ones.

Wager et al. grouped quality models into two types in which he defined the composition of software quality and quality models. Product factors introduced by Wager were used to improve the traceability mapping on the quality characteristics.

Quality models are used to make the abstract concepts of software quality more tangible in nature. Taxonomies for check listing the quality requirements and overall quality assessment metrics can be formulated<sup>19</sup>.

Dromey utilized a quality model having quality attributes, components and their interrelationships. He demonstrated the impact of properties on attributes in a precise manner.

Wager first described the product factors and then



**Fig. 9: Generic Process of risk based testing**

gave a model for quality assessment based on this model. He formulated aggregation rules in the collected data for measurement and provided the evaluation specifications which defined the measurement for product entities. Quality models for open source systems were used by Lochmann which described product factors impact on quality characteristics<sup>20</sup>.

### Risk Oriented Testing

Risk based testing is a composition of various stages i.e. risk identification then assessment and finally analysis and evaluation. We have shown the generic risk based testing in the figure 9 as follows<sup>21</sup>:

### Conclusion and future work

Quality assurance in risk oriented testing has been explored very less now. Once we mitigate the risk items in the software by testing then quality of software gets automatically improved for mitigating the risks. Quality modelling helps in quality assessment of the software. Various quality models have been proposed in past but very little work has been done to actively monitor quality of software by regular risk assessment. So in future we can work out for giving a new quality assurance model that can assess the software risks and mitigate them for quality improvement. Furthermore we can apply quality assurance in risk oriented testing with the help of evolutionary algorithms/genetic programming/mathematical modelling as none of these have been explored in past.

### References

1. A. Mathrani, "Quality Assurance Strategy for distributed software development using Managed test lab model", IEEE, 2014.
2. H. Hemmati , M. Nagappan , "Investigating the effect of "defect co fix" on quality assurance resource allocation :A search based Approach" ,Elsevier, 2014.
3. B. Tekindorgan, "A chapter on quality concerns in large scale and complex software intensive systems", Elsevier,2016.
4. N. Greif, "Software Testing and preventive quality assurance", Computer standards and interfaces, Elsevier, 2005.
5. M. Klas , "Support planning and controlling of early quality assurance by combining expert judgment and defect data—a case study", *Empirical software engineering*, 2010.
6. K. Holl, "Towards a perspective based usage of Mobile failure patterns to focus quality assurance", Springer International Publishing, 2015.
7. J. Neubauer, "Risk based testing via active continuous quality control", *International Journal of software tools technology transfer*, 2014.
8. M. Gleirscher, "Introduction of static quality analysis in small and medium sized software enterprises: Experiences from Technology Transfer", *Springer International Journal of Science+ Business Media*, 2013.
9. A.A. Hernandez, "JLearn : An Instructional Environment for Java Program Composition Integrating Test Driven Development and Life cycle management for software Quality Assurance", International Conference on Networking and Information Technology, 2010.
10. J. Nodler, "A flexible framework for quality assurance of software artefacts with applications to java, UML, and TTCN-3 Test Specifications , International Conference on Software Testing Verification and Validation, 2009.
11. F. Elberzhager, "An Integrated Analysis and testing methodology to support model based quality assurance", *Springer*, 2014.
12. H. Zhu, Cooperative Agent approach to quality assurance and testing web software , 28<sup>th</sup> International computer software and application conference, IEEE, 2004.
13. D. Winkler, "Software Process Improvement initiatives based on Quality Assurance Strategies : A QUATAM Pilot application", CCIS, 2010.
14. Van Veeneendaal, Chapter on Testing based on user quality goals, Reliability, Quality and



- safety of software intensive systems, 1997.
15. T.M. Khoshgoftaar, "Analogy based practical classification rules for software quality estimation", *Empirical Software Engineering*, 2003.
  16. F. Matthes, "Testing and Quality Assurance in data migration projects", IEEE, 2011.
  17. D.Winkler, "Collective Intelligence based quality assurance: Combining Inspection and risk assessment to support process improvement in Multi Disciplinary Engineering", *Springer*, 2016.
  18. K .Holl, "Towards a perspective based usage of mobile failure patterns to focus quality assurance", Springer International Publishing, 2015.
  19. H.Foidal, "Integrating software quality models into risk based testing", *Software quality journal*, 2016.
  20. B. Tekinerdogan, "Quality concerns in large –scale and complex software –intensive systems", *Software quality assurance, Elsevier*, 2016.
  21. M.Felderer, R. Ramler, "Integrating risk-based testing in industrial test processes", Springer Science+ Business Media, *Software Qual J*, vol.**22**, pp.543-575, 2014.