A Quality Assurance Model for Analysis Phase

Reham Ejaz Mubina Nazmeen Maryam Zafar
University Institute of Information Technology, PMAS Arid Agriculture University
Rawalpindi, Pakistan
+92-51-9290154
reham_ejaz@yahoo.com, mubi_139@yahoo.com, oiyam_9@yahoo.com

ABSTRACT
Quality Assurance should be carried out in each phase of the development life cycle. For high quality software, its requirements document should be a quality product. It should be complete, unambiguous and understandable**.** If the documents produced in the early life cycle phases are left over with defects then it results in a greater number of defects in the final product. A defect may appear to be a low risk defect in the early stages but it may take a very serious shape in latter phases. If the analysis people provide a quality document the development effort of the software is reduced to nearly 50 percent. This is why the quality assurance activity is more critically needed during the analysis phase. In this paper we have proposed a model for the quality assurance people for the systematic verification and assessment of the analysis phase.

Categories and Subject Descriptors
D.2.5 [**Software Engineering**]: Quality, Testing

General Terms
Measurement, Documentation, Reliability, Experimentation, Verification.

Keywords
Requirements, requirements analysis, software quality, quality assurance, quality assurance techniques

1. INTRODUCTION
Quality of requirements is a vital element in a project's success. If in the beginning of a project, time is spent for reviewing requirements then it adds to the project's growth. In the analysis phase when the requirements are organized they are reviewed by the Quality Assurance (QA) people for the quality factors like clarity, redundancy, completeness and testability. The QA people have to make sure that each requirement is specific enough to be testable and is specifying what is needed to be done; each requirement is uniquely identified for later tracing to code or tests. Ultimately the QA people have to verify the implementation of each requirement in the finished application and the correspondence of each feature of finished application to a requirement. All this work has to be done by QA as defects in an application are sorted out as missing requirements, wrong requirement is considered as a failure to provide an agreed upon functionality and extra features. Also a non implemented system. Functionality with no matching requirement is an error that can increase schedule and budget. The analysis phase should be the start of a period of ongoing auditing and reviewing by QA for verification of conformity with requirements, standards and procedures. The effort spent in these activities during early phase is enormously valuable, since defects during early stages are much cheaper to detect and fix than those found in later phases of development and testing. The QA people perform detailed desk checking against the standards during this phase [15]. By applying the quality assurance strategies in early phases, high level of customer confidence can be maintained [15].

Software testing and verification do not measure the quality requirements of software that is why we need a process that does it. Unlike testing and verification the quality measurement process can be carried out during requirements and design phase. Assessing the software quality during earlier phases can save a lot of time and money [2].

Many approaches, tools and methodologies are introduced for measurement of software quality. Software quality factors are also identified which help in analyzing software quantitatively. But there is a need for a framework to assure the quality specifically in the requirements analysis phase. The contribution of this paper is such a model that will provide a roadmap to the quality assurance people to conduct their activities in a systematic manner during the requirements analysis phase. The proposed framework is also verified with the help of a questionnaire distributed in various software houses. The results prove the importance of carrying out the QA activities during the analysis phase for effort and cost reduction in software development.

The rest of the paper is organized as: Section 2 presents our proposed quality assurance model for the analysis phase of software development. In section 3 we have discussed the results from various software houses. Section 4 contains related work, section 5 concludes the paper and finally section 6 includes the references.

2. PROPOSED QUALITY ASSURANCE MODEL
The requirements analysis phase of software development is the first and the most important phase of whole development life cycle. Software is only successfully developed if its requirements are clearly understood, stated and properly followed. The defects left off in this phase can cause major problems in later phases and can result in increased development effort and cost. The quality assurance team can play an important role during this phase. If they assure the quality of each step in this phase not only cost but efforts can also be reduced.

NSEC'10, 04-OCT-2010, Rawalpindi, Pakistan

In this section we propose a Quality Assurance model for requirements analysis phase that evaluates all the main elements to be assessed for quality in a requirements analysis phase. The main elements to be considered are the people, the process and the product. Figure 1 shows the proposed model. This model is a generic model that is applicable to all types of systems.

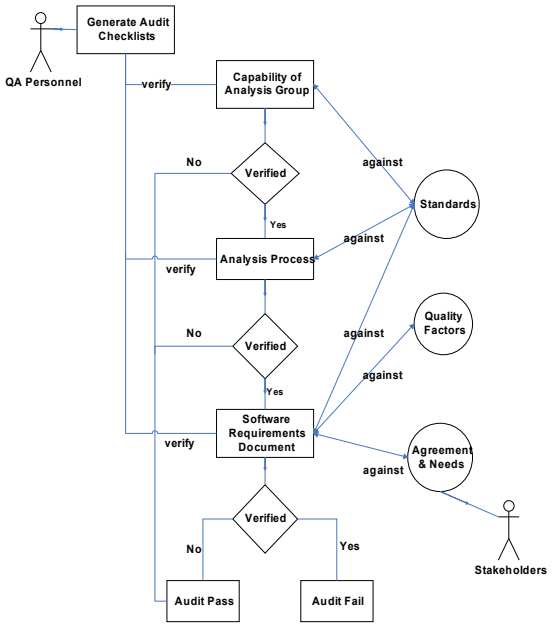

Figure 1. Quality assurance model for analysis phase

Following is the description of each element of the proposed QA model.

2.1 Generate Audit Checklists

The first and foremost step of the quality assurance team for assessing the quality of analysis phase is to generate checklists. It would serve as a basis for the QA team to conduct their assessment. The checklist should include queries regarding all the three pieces of focus in the analysis phase i.e. the people, the process and the product.

2.2 Verification of the Capabilities of the Analysis Group

After the generation of the checklist, the actual verification process starts. The QA team has to assess the abilities, competence and experience of the group of people carrying out the analysis activities. The analysis group should be capable of performing the most difficult and challenging task in the software development. If their capabilities do not satisfy those written in standards then they are not worthy of it, the audit is failed. If they are competent enough to meet the standards SQA team approves them worthy of performing their jobs.

2.3 Verification of the Analysis Process

Once the analysis group is verified, next comes the process that group is following to fulfill their activities. The QA team has to verify the following

- Is the process being followed a standard process?
- Is the process being followed in a standard manner?

The QA team evaluates the process of analysis phase on the basis of standards. If any of the task performed is not according to standard and not performed well then it fails the audit. But if the analysis group is working according to standards and successfully produce the requirements documents then the next responsibility of the QA team would be to verify those documents.

2.4 Verification of the Requirements Document

The software requirements analysis phase produces two kinds of outputs – Software Requirements Specifications (SRS) and Interface Requirements Specifications (IRS).

These documents are the foundation for the whole software development effort. The SQA team should have to verify that the requirements documents contain all the contents that were agreed upon with the stakeholders. It includes all the needs specified by the stakeholders. The SQA team also has to review the requirements traceability. All the requirements specified at the higher level during initial analysis should be traceable to the requirements in the SRS and IRS. Also the other way round is considered that is each requirement in the SRS and IRS should be traceable to a higher level requirement meaning that it should have a basis. If in case some requirement has no base then it should be determined by the SQA team that whether it is some valid requirement or some derived one If it is not proved to be any of these then it is marked as an invalid requirement.

SQA team also has to determine whether the SRS and IRS conform to their standards or not.

The SQA team must also evaluate the adequacy of quality factor requirements. The quality factors that are specifically related to the verification of a requirements document are ambiguity, completeness, redundancy, clarity/understandability, traceability, volatility and testability. The SQA team should ensure that

- The requirements document is not ambiguous; each requirement statement is itself interpreting its meaning.
- The requirements document is complete enough to specify all the necessary details for proceeding to design.
- Same requirement is not specified more than once in a different manner to increase the complexity.
- All the requirements are clearly stated and are understandable by the developers to follow them for design and implementation.
- The requirements should be traceable. All the software requirements should have some basis as mentioned earlier and each software requirement should be mapped onto some higher level stated requirement. Also all the requirements stated in requirements documents should be visible in the design and latter phases.
- The changing requirements during the development are incorporated into the requirements document. The requirements document has been maintained by the configuration management.
- The documented requirements are verifiable. That is it can be checked from the software whether or not the requirements have been fulfilled.

- The requirements document includes the performance requirements of the software. That is it should be checked for steps included for speed and efficiency.

Some special areas regarding safety, security and design constraints should also be reviewed openly by the SQA team.

2.5 Audit Pass and Audit Fail

If all the elements are verified i.e. the analysis group passes the assessment, the analysis process meets the standards and the procedures; the requirements documents is proved according to the standards, the signed agreement and needs of all the stakeholders and for the quality factors then the audit carried out by SQA is passed. If any of these assessments fail to meet the standards then the audit fails.

3. RESULTS & DISCUSSIONS

To prove the correctness and applicability of our framework we developed a questionnaire including some qualitative and quantitative questions relevant to our framework and distributed it among various software houses. The questionnaire contained 19 questions relevant to the size of the software house, the CMMI level it has achieved and the various practices mentioned in our framework. We got response from 10 different software houses standing at different CMMI levels. We divided the results in three groups on the basis of the first three CMMI levels. The average results of these groups are shown in table 1. All the software houses laying in these three groups practice the activities mentioned in our framework. Their analysis teams are either fully aware of the standards or at least few of them are aware of them. The questionnaire contained questions regarding the reduction of development effort if the analysis group has done their job right. The answers received from various groups laid in the ranges 20 – 30%, 40 – 50% and more than 50%. We calculated an average of the answers in different groups and the summary is presented in table 1. Similarly another question was included regarding the reduction in cost if the defects are identified and removed at early stage contained answers in ranges 10 – 20%, 20 – 30%, 40 – 50% and more than 50%. The average of replies from different groups is also shown in table 1.

Table 1. Analysis of results from software houses

CMMI Level	Reduction in Development Effort	Reduction in cost of defect identification and Removal
3	51.67%	28.33%
2	55%	55%
1	35%	60%
Average	47.22%	47.78%

The analysis table shows results from various software houses standing at CMMI level 1, 2 and 3. The figures shown in table are the average results of the software houses. The average of the three levels is also shown in the table. We can clearly see from it that on average the development effort of any software system is reduced to 47.22% if the analysis team puts in their best efforts to do the job right. While the cost of defect identification and removal is reduced to 47.78% if they are identified in early phases i.e. during the analysis phase.

From the above discussion it is clear that if much effort is put in the analysis phase it can be helpful in later phases. It is very important to pay attention in the quality checking of this phase as if the work is done right and errors are removed here a lot of time, effort and money can be saved.

4. RELATED WORK

Some work is seen in the quality assurance of the requirements analysis phase and the quality of the requirements and its documents. A brief discussion is included in this section.

Various quality assurance techniques e.g. Fagan inspections [4], Gibs inspections [5], or reading techniques for inspection [1] are there to help in locating defects in early documents e.g. requirement specifications. These techniques are also proved effective and efficient through empirical studies [10]. Inspection is a three step quality assurance procedure involving individual defect detection, defect collection and defect repair [4] [5].

Everhart et al [3] suggested Total Quality Management (TQM) for improving quality of systems that can be applied successfully to system development life cycle phases.

A quality model consisting of goals, attributes and metrics was presented by Hayyat et al [8]. The corresponding attributes includes ambiguity, completeness, understandability, requirements volatility and traceability [8]. Their focus is on complete, unambiguous, understandable requirements document, steady requirements and requirements traceability from source to all the phases of software development.

Williams et al. [16] presented a framework to improve the effectiveness of requirements engineering process. They have modeled the key factors of improvement in their framework and quality assurance is one major factor which is needed for user's satisfaction and for decision making.

Sawyer et al. [14] presented some good practices for the requirements process. These include identifying each requirement uniquely, defining policies for requirements management and traceability, maintaining traceability manual, managing requirements in a database, definition of change management policies, identification of global system requirements and volatile requirements and recording the rejected requirements. All these should and can be assessed by the quality assurance people.

According to Hills [9] QA strategies must be integrated within the procedures of a project specifically during early analysis phase; otherwise they themselves become an entry point for error.

Grunbacher et al. [6] proposed an Easy Win Win approach that focuses on requirements negotiations. In this approach the entire negotiation period is checked for quality. Also an inspection process is introduced that systematically detects the defects in requirements and reports and documents along with its solution for easy traceability [6].

A quality assessment company [12] has proposed two types of QA methodologies, Process Focused Quality Assurance (PFQA) and Focused Quality Assurance (AFQA). The PFQA methodologies and frameworks measure the quality of underlying development process while the AFQA focuses on measuring quality of application through all phases of the application development process and guides the application development process itself. According to them a quality measurement model in requirements

analysis phase requires the assurance of user quality, internal quality and external quality [12].

Lee et al. [11] proposed a quantitative software quality evaluation model for component based software. They have incorporated the international standards for software quality model and evaluation process into the proposed model. Particularly, they have adopted weights of quality characteristics [11].

Salem [13] suggested that QA people need to verify the requirements and assure their quality as these have to be their in the finished application otherwise it is considered to be a failure, so there is a need to trace the requirements to the design and later phases. For this purpose there is a requirements traceability model through which the software quality can be improved [13].

According to Hattori et al [7] the major issue for budget overrun and exceeding deadline is the lack of systematic quality control and assessment methods for requirements specification.

5. CONCLUSION

Quality assurance plays a vital role in the software development particularly during the analysis phase. The efforts put in the analysis phase of the software development can help in reducing the development effort to nearly 50 percent. Moreover the cost of defect identification and removal is reduced to nearly 50 percent if it is carried out during analysis phase instead of in later phases. If quality assurance strategies are integrated into the requirements analysis phase of any software then we can assure their implementation into the finished application. This is why the importance of quality assurance during this phase increases. This paper presents a quality assurance model that can help the respective team to carry their activities during analysis phase in a systematic manner. If the proposed model is followed to assess the analysis phase then the quality assurance team can successfully measure its quality.

6. REFERENCES

[1] Basili, V., Green S., Laitenberger, O., Lanubile, F., Shull, F., Soerumgaard, S., and Zelkowitz, M. 1996. The Empirical Investigation of Perspective-Based Reading. *Empirical Software Engineering: An International Journal,* vol. 1, no. 2, 1996, pp. 133-164.

[2] Cavano, J.P. and McCall, J.A. 1978. A Framework for the Development of Software Quality. In *Proceedings of the software quality assurance workshop on Functional and performance issues,* ACM 1978.

[3] Everhart, R., Salle, A.J.L. and Shahgol, R.K. 1995. Applying Total Quality Principles to the Requirements Phase of System Development. In *Proceedings of Engineering Management Conference, 1995* IEEE.

[4] Fagan, M. 1976. Design and Code Inspections to Reduce Errors in Program Development. *In IBM Systems Journal*, vol. 15, no. 3, 1976, pp. 182-211.

[5] Gilb, T. and Graham, D. 1993. Software Inspection, Addison- Wesley, 1993. ISBN 0-20163181-4, DOI: 10.1002/smr.4360070306.

[6] Grunbacher, P., Halling, M. and Kepler, J. 2002. Repeatable Quality Assurance Techniques for Requirements Negotiation.In *Proceedings of the 36th Hawaii International Conference on System Sciences,* IEEE, 2002.

[7] Hattori, N. and Yamamoto, S. 2008. Quality Analysis of Information Technology Requirements to Support Knowledge Innovation. *NTT Technical Review*, vol. 6, no. 7, July 2008.

[8] Hayyat, L. and Rosenberg, L., NASA Goddard Space Flight Centre, United States of America,1996. A Software Quality Model and Metrics for Identifying Project Risks and Assessing Software Quality, ESA 1996.

[9] McCain, M.J. and Masters, W.C. 1998. Integrating Quality Assurance into the GIS Project Life Cycle. DOI= http://proceedings.esri.com/library/userconf/proc98/proceed/to150/pap124/p124.htm

[10] Laitenberger, O. 2000. Cost-effective Detection of Software Defects through Perspective-based Inspections. PhD thesis, University of Kaiserslautern, Germany, May 2000.

[11] Lee, K. and Lee, S.J. 2005. A Quantitative Software Quality Evaluation Model for the Artifacts of Component Based Development. In *Proceedings of the Sixth International Conference on Software Engineering, Artificial Intel ACIS International Workshop on Self-Assembling Wireless Networks (SNPD/SAWN'05)*, 2005 IEEE.

[12] Systems, Q.A. 2004. Application Focused Quality Assurance. Software Assessment Company. November 2004, Version 1.1. DOI: http:www.qa-systems.com

[13] Salem, A.M. 2006. Improving Software Quality through Requirements Traceability Models. Department of computer science, California State University, Sacramento 2006 IEEE.

[14] Sawyer, P., Sommerville, I. and Viller, S. 1999. Capturing the Benefits of Requirements Engineering. *IEEE Software*, March/April 1999.

[15] Tierstein, J.L., Benoit, H. and Systems,W.R. 2009. Integrating Quality Assurance into the Software Development Life Cycle. DOI= http://www.docstoc.com/docs/20091854/Integrating-Quality-Assurance-into-the-Software-Development-Life-Cycle

[16] Williams D. and Kennedy, M. 1999. A Framework for Improving the Requirements Engineering Process Effectiveness. *Symposium on Systems Engineering.* DOI= http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.110.3542&rep=rep1&type=pdf