Journal of Engineering and Applied Sciences 12 (21): 5548-5553, 2017
ISSN: 1816-949X

Evaluating the Effectiveness of Scoring Rubrics Assisted Reading (SRAR) Technique in the Reading of Software Requirements Work Documents

Emmanuel O.C. Mkpojiogu and Nor Laily Hashim
School of Computing, Universiti Malaysia, 06010 Sintok, Malaysia

Abstract: The quality of requirements work products and of software depends on how effective the reading technique and mechanism is. A number of reading techniques have been proposed and in use. These techniques assist in the reading process, in the detection of defects and errors and in the enhancement of the quality of software artifacts. Nevertheless, the effectiveness of a range of reading techniques is at varying degrees. The most familiar and common reading technique is the checklist-based reading technique. This technique, although, it is an enhancement over the ad hoc reading technique is however, bugged with some limitations that affect its usability and effectiveness. It provides too wide-ranging questions to readers without any precise guidance on how they can go about the reading exercise. No previous work has been done on evaluating the effectiveness of SRAR. This study evaluates the effectiveness of the Scoring Rubric-Assisted Reading (SRAR) technique using two metrics: defect density and defect criticality rate. The method used is as follows: the use of SRAR, detection of defects, counting of defects present in the artifacts and refining of the artifact for each round/iteration of review. The defects found by reviewers were counted and presented as averages and percentages. Preliminary result reveals the effectiveness of this technique. SRAR is effective because it is helpful and successful in detecting defects, reducing the amount of defects (including critical defects) in a software artifact and in improving the quality of such software artifact.

Key words: Requirements documents, effectiveness, scoring rubrics assisted reading, SRAR, enhancement, percentages

INTRODUCTION

Software reading is part of the process of building high quality software. It offers assurance on the quality of a software work documents (Berling and Runeson, 2003). According to IEEE Std 830 (1998), a software inspection or review is the visual examination of a software product to detect and identify software anomalies, including errors, violation of development standards and other problems inspections have been performed on artifacts from all phases of the software lifecycle including requirements (Aceituna *et al*., 2011; Albayrak and Carver, 2014). The quality of work products such as the software requirements specification and requirements models among others is central to the success of the software development process. The reviews for the verification and validation of software documents and artifacts are widely employed and in practice but the approaches used are rather ad hoc or even better, checklists-based, however, they mainly rely on the knowledge, intuition and skill of the review personnel (Shull *et al*., 2000). There are several reading techniques, these include inter alia: ad hoc reading, checklist reading, defect-based reading and perspective-based reading. Inspection is a commonly employed methodology in verification (Sommerville, 2007).

As mentioned before, many techniques exist for individual or collective reading and reviewing of requirements/software work products and documents. Ad hoc review or reading technique is at one extreme end. The technique has no formal or systematic procedure and it is based exclusively on the reviewer's skill, experience and intuition (Shull *et al*., 2000). The checklist reading technique offers a little enhancement over the ad hoc approach. It makes the inspection process a little better defined through the offering of a catalog of items on which to focus on to the reviewers. Furthermore, defect-based reading technique is a scenario-based reading technique that makes available a collection of systematic steps and procedures which reviewers can follow as guides in the review process. Finally, the perspective-based reading is also a scenario-based reading technique that offers step by step guidance that is geared and suitable to requirements expressed in a natural language like English. Reader's involvement is

Corresponding Author: Nor Laily Hashim, School of Computing, Universiti Malaysia, 06010 Sintok, Malaysia

Table 1: A comparison of reading techniques; adapted from Shull *et al.* (2000)

Reading type	Systematic?	Focused?	Controlled?	Customizable?	Training?
Ad hoc reading	No	No	No	No	No
Checklist reading	Partly	No	Partly	Partly	Partly
Defect-based reading	Yes (Complex)	Yes	Yes	Yes	Yes (Detailed)
Perspective-based reading	Yes (Complex)	Yes	Yes	Yes	Yes (Detailed)
Scoring Rubrics Assisted Reading (SRAR)	Yes (Simpler)	Yes	Yes	Yes	Yes (Simpler)

based on some perspectives which they center and focus on. Perspective-based reading requires the identification of users of definite software artifact (such as requirements, model, etc.) (Shull *et al*., 2000). Basili *et al.* (1996) and Ciolkowski *et al.* (1997) reported from their experiments that perspective-based reading performed better than checklist-based reading technique. Laitenberger *et al.* (2001) report based on three experiments revealed that perspective-based reading improved best the performance of reviewers that have medium skills and less for those with very little or very high skills. This observation revealed that very experienced and skilled reviewers do not actually rely on the use of perspective-based reading while they read documents and work products. It also revealed that little skilled reviewers find it difficult to cope since scenarios use individual abilities for a given model construction and task (Laitenberger *et al*., 2001).

Furthermore, Porter and Votta (1994) in their work compared scenario-based reading, checklist-based reading and ad hoc reading in requirements specification review and inspection. The result of their study indicated that scenario-based reading was the most effective in finding defects. They found no significant difference between ad hoc and checklist-based reading techniques. In a related study, Porter *et al.* (1995) replicated the above experiment. The researchers assessed and compared defect-based reading, ad hoc reading and checklist-based reading and established that: the defect-based readers performed better than ad hoc and checklist readers; the defect-based reading procedures assisted reviewers to focus on specific defect classes but were still effective in finding defects; checklist reading was not more effective than ad hoc reading. In addition, Basili *et al.* (1996) evaluated and compared perspective-based reading and NASA's current reading technique with regard to defect detection effectiveness. They found that: both individual and team scores improved when perspective reading was applied to generic documents; team scores improved when perspective-based reading was applied to NASA documents. From the forgoing comparisons, it is evident that the effectiveness of the reading techniques grows in the following order: ad hoc, checklist and scenario-based reading techniques. The list grows from the most rudimentary and less defined to the more defined and complex techniques. The SRAR reading technique is conceptually somewhere in the middle. SRAR is defined and yet simple to use.

Researchers who have widely studied software inspections agree that they are effective techniques of identifying faults and improving software quality (Aurum *et al*., 2002; Fagan, 1986; Kollanus and Koskinen, 2009). The choice of inspection technique impacts inspection effectiveness. Prior research in this area is inconclusive about which techniques are better (Ciolkowski, 2009; Winkler *et al*., 2010). An early study by Porter *et al.* (1995) found that scenario-based reading techniques were more effective than either a checklist or an ad hoc approach. Later studies contradicted this result by showing no benefit for scenario-based techniques (Sandahl *et al*., 1998; Fusaro *et al*., 1997). In their review of the impact different scenarios have on inspection effectiveness, Regnell *et al.* (2000) found the results to be inconclusive. Furthermore, their own replication of an earlier study showed that specific perspectives do not seem to affect effectiveness (Regnell *et al*., 2000).

As can be seen in the Table 1, Scoring Rubric Assisted Reading (SRAR) has the following features: systematic: the items in the rubrics follow a systematic ordering of the expected attributes in the work product. It is systematic and yet simple when compared to the two scenario-based reading techniques; focused: like the two scenario-based reading techniques, SRAR is focused in the sense that it focuses on the expected attributes of the work product, of which a deviation from, implies a defect. It has an in-built standard that guides. It focuses on the expected structure form, standard and documentation of the particular document; controlled: SRAR is also controlled in the sense that its attributes and dimensions are properly defined in the rubrics; customizable: SRAR is customizable and can be tailored to a particular software/requirements document to be reviewed. It can also be tailored to any organization and situation; training: SRAR requires simple and minimal training for users as it is a simple mechanism, however, defect-based and perspective-based reading approaches are more complex having more detailed training. In all these characteristics, SRAR towers above both ad hoc and checklist reading techniques.

The place of the effectiveness of reading techniques cannot be over emphasized. The degree of success achieved in the process of reading requirements documents and work products and in the detection of defects in such documents using a given reading technique defines the effectiveness of reading technique used. Software requirements reading effectiveness, a quality that shows how successful a reviewer is in detecting errors in a requirements document using a given reading technique is an important dimension in the evaluation of a reading technique. ISO (2011) provided a quality model consisting of five characteristic as quality in use, inter alia: effectiveness, efficiency, freedom from risk, satisfaction and context coverage. It also itemized eight more characteristics as internal and external qualities, these include: functionality, suitability, compatibility, security, reliability, usability, performance efficiency, maintainability and portability. This study uses effectiveness (strongly associated with usability) (IEEE Std. 830, 1998; ISO, 2011). Effectiveness of a reading technique thus is the level of success at which the reading technique can detect defects and errors in a software requirements work products (Basili *et al*., 1996; Porter and Volta, 1994; Porter *et al*., 1995). The amount of defects and errors detected from a work product and the thoroughness in the detection process indicate the effectiveness of the reading/ review technique used. This study employed a reading technique that is conceptually better than the ad hoc and checklist reading techniques. So, the study seeks to evaluate this reading technique. No preview work has been done on evaluating the effectiveness of SRAR. This study thus seeks to investigate the effectiveness of SRAR approach in reading requirements documents. However, the study is an exploratory one and hence will not compare existing techniques with SRAR.

MATERIALS AND METHODS

In this study, the SRAR technique was used to assist in the detection of defects in requirements work products. It was also used to evaluate the quality of the requirements products. A scoring rubric is a two-dimensional likert-like instrument. The columns represent a 4-point likert-type rating scale (for example, Not acceptable, below expectations, meet expectations, exceeds expectations); a fifth column is added for "not applicable". The rows consist of the attributes of the given work products. In some cases, the attributes are further defined into criteria. In addition, the cells created by the intersection of the rows and columns represent a clear description of the work product's attributes with respect to the corresponding rating scale. Each attribute is scored and the scores of all attributes are totaled to form a total score for the given work product. Rubric scores are performance metrics. SRAR is both reliable and valid.

A number of metrics were employed to evaluate the effectiveness of the scoring rubric-assisted reading technique. They include: defect density and criticality rate. These metrics assesses the effectiveness of SRAR technique. Defect density and criticality rate evaluate the effectiveness of the SRAR (Basili *et al*., 1996; Porter and Volta, 1994; Porter *et al*., 1995). Effectiveness is the determination of how successful the SRAR technique is in detecting defects and evaluating the quality of a work product. These metrics measures the effectiveness of SRAR in detecting defects and faults in the work documents. In this study, to measure the effectiveness of SRAR, defect density and defect criticality rate were used. Defect density is the amount (count) of defects per page of a requirements work product report while criticality rate is the frequency (count) of critical defects per page of a given work document (Basili *et al*., 1996; Porter and Volta, 1994; Porter *et al*., 1995).

This study was conducted in School of Computing, Universiti Utara Malaysia. The Software Engineering sub-department has for some years now been using scoring rubrics in the assessment of the quality of student's software/requirements documents and models. But the effectiveness of this measure has not been empirically validated. This study is the first attempt to proffer an empirical evaluation of the effectiveness of the measure and technique. This study was part of the study carried out to develop an e-Health awareness system (Hussain and Mkpojiogu, 2016; Hussain *et al*., 2015; 2016; Mkpojiogu and Hashim, 2017). After the requirements documents and models were produced, they were read and reviewed in two rounds by 4 reviewers who detected defects in the documents. Each reviewer read and reviewed the requirements documents and models independently. After each round of review, the requirements work products were refined and the defects removed. The review of requirements models was done by 4 senior lecturers (in Software Engineering) in the School of Computing, Universiti Utara Malaysia. The following research question guided the study: Is SRAR technique effective? Descriptive statistics was used in answering the research question. The following metrics was used in the study: defect density and criticality rate. Defect density is computed by counting the defects per document and dividing the total count by the total number of pages of the document. Also, criticality rate is

computed by averaging the total critical defects by the total number of document's pages. Details are described in the following study.

RESULTS AND DISCUSSION

Defect density: The defect density for all work products read and inspected in the first round were higher than those of the second round of the inspection process, indicating improvements in the quality of the work documents. This explains the effectiveness of the SRAR approach. To show how effective SRAR the expected outcome of the metric is a percentage reduction in defect density. Defect density is the amount of defects per given work product/document. After a round of review and subsequent refinement of the work product if there is an increase in defect density, it implies a low effectiveness of the reading technique and mechanism. However, a reduction in the defect density for the given work product indicates good effectiveness of the reading technique/ mechanism. The result as presented in Table 2 shows that all the work documents examined in this study had a good percentage reduction in defect density. This reveals that SRAR is effective in reducing the defects in the work products after the two iterations of review. Vision and scope document has a 50% reduction in defect density. Use case description has a 37% reduction in defect density. For the software requirements specification, the percentage reduction in defect density is 59%. For the test plan, the percentage reduction in defect density is 44%. The percentage defect density reduction for test cases is 58%. The percentage reduction in defect density for all the requirements models combined is 79%. The average defect density reduction for all textual documents examined in the study is 74% while that of all textual documents combined with all requirements models put together is 77%. In essence, there is a 100% success rate as all work products examined in the study had observable reduction in their defect densities. This attests to the effectiveness of the reading technique employed (that is SRAR). The overall defect density is 1.14. There is an observed improvement in the quality of each requirement work product, occasioned by the percentage reduction in their respective defect density, this thus, explains the effectiveness of the SRAR technique.

Criticality rate: Criticality rate is the degree of critical defects in a particular work product. In this study, the frequency of critical defects found in the work documents on the average is lower in the second round in comparison to those found in the first round. This indicates an improvement in the quality of the work products inspected. It also shows the effectiveness of the reading mechanism. As afore stated, this exploratory reading study was assisted using scoring rubrics. The results as shown in Table 3 indicates a good improvement in the quality of the requirements work products. It provides some evidence to show that its use enhanced the quality of the inspected products. This attests to SRAR's effectiveness. A percentage reduction in criticality rate is the expected outcome of this metric as a reduction in the work product's criticality rate indicates a good effectiveness of the reading technique. The reverse is the case for an increase in the criticality rate of the work product. All the requirements documents examined had some percentage reduction in criticality

Table 2: Defect density for two rounds (iterations) of reading

Work product	Defect density (Round 1)	Defect density (Round 2)	Difference	Reduction in defect density (%)	Improvement?
Vision and scope document	0.20	0.10	0.10	50	✓
Use case description	0.37	0.00	100	37	✓
Software request spec	0.59	0.00	100	59	✓
Test plan	0.39	0.22	0.17	44	✓
Test cases	0.40	0.17	0.23	58	✓
Request models	1.80	0.38	1.42	79	✓
Average defect density for documnets	0.39	0.10	0.29	74	✓
Average defect density for documents and request models	0.62	0.14	0.48	77	✓

Table 3: Average criticality rate for two rounds (iterations) of reading

Work product	Criticality rate (Round 1)	Criticality rate (Round 2)	Difference	Reduction in criticality rate (%)	Improvement?
Vision and scope document	0.00	0.00	0.00	0	✓
Use case description	0.38	0.00	0.38	100	✓
Software requests spec	0.50	0.00	0.50	100	✓
Test plan	0.00	0.00	0.00	0	✓
Test cases	0.00	0.00	0.00	0	✓
Reqs models	0.56	0.70	-0.14	-25	×
Average reading time for documents	0.50	0.00	0.50	100	✓
Average reading time for documents and requests models	0.35	0.22	0.13	37	✓

rate except one (that is requirements models). The overall criticality rate is 0.33. Using SRAR can achieve effective defect detection and work product quality improvement.

In sum, two metrics were employed to evaluate the effectiveness of SRAR: defect density and criticality rate. In all the metrics, it is observed that after round two of the reading and review, there was observable improvement in the work products quality levels as evidenced in all the metrics used. The average defect density for non-model products improved by 74% and when combined with requirements model products, there was a 77% improvement. This indicates the effectiveness of the reading mechanism. Also, there was 100% reduction in the criticality rate for requirements models and 37% reduction for a combination of requirements models and textual documents. This also, indicates the effectiveness of the reading technique. Also as can be seen from the analysis in Table 2 and 3, there was consistent improvement in the quality of requirements work product.

SRAR offers to be an effective reading technique that overcomes the inadequacies of the ad hoc and checklist reading techniques. It is simple and well defined technique that guides reviewers in the review process. It is also consistent. It has efficacy in assisting reviewers/readers detect defects in software/requirements work products. In addition, it can be used to monitor the progress of quality improvements of work products.

CONCLUSION

However, this study was an exploratory one and was limited in the sense that effectiveness of SRAR was not empirically compared with checklist or any other reading technique but its effectiveness was assessed based on the defect defection capability and quality improvement efficacy as observed from the study.

RECOMMENDATIONS

Future researches will evaluate and compare the effectiveness of SRAR with that of other reading techniques using both professional and non-professional users/readers. Furthermore, in the future, the efficiency of the SRAR reading technique will be assessed.

REFERENCES

Aceituna, D., H. Do, G.S. Walia and S.W. Lee, 2011. Evaluating the use of model-based requirements verification method: A feasibility study. Proceedings of the 2011 1st International Workshop on Empirical Requirements Engineering (EmpiRE), August 30, 2011, IEEE, Trento, Italy, ISBN:978-1-4577-1075-9, pp: 13-20.

Albayrak, O. and J.C. Carver, 2014. Investigation of individual factors impacting the effectiveness of requirements inspections: A replicated experiment. Empirical Software Eng., 19: 241-266.

Aurum, A., H. Petersson and C. Wohlin, 2002. State-of-the-art: Software inspections after 25 years. Software Testing Verification Reliab., 12: 133-154.

Basili, V.R., S. Green, O. Laitenberger, F. Lanubile and F. Shull *et al*., 1996. The empirical investigation of perspective-based reading. Empirical Software Eng., 1: 133-164.

Berling, T. and P. Runeson, 2003. Evaluation of a perspective based review method applied in an industrial setting. IEEE. Proc. Software, 150: 177-184.

Ciolkowski, M., 2009. What do we know about perspective-based reading? An approach for quantitative aggregation in software engineering. Proceedings of the 3rd International Symposium on Empirical Software Engineering and Measurement ESEM, October 15-16, 2009, IEEE, Lake Buena Vista, Florida, USA., ISBN:978-1-4244-4842-5, pp: 133-144.

Ciolkowski, M., C. Differding, O. Laitenberger and J. Munch, 1997. Empirical investigation of perspective-based reading: A replicated experiment. Master Thesis, Fraunhofer IESE, Kaiserslautern, Germany.

Fagan, M.E., 1986. Advances in software inspections. IEEE. Trans. Software Eng., 12: 744-751.

Fusaro, P., F. Lanubile and G. Visaggio, 1997. A replicated experiment to assess requirements inspection techniques. Empirical Software Eng., 2: 39-57.

Hussain, A. and E.O.C. Mkpojiogu, 2016. An application of Kano method in the elicitation of stakeholder satisfying requirements for an E-ebola awareness system. Intl. J. Syst. Appl. Eng. Dev., 10: 169-178.

Hussain, A., E.O.C. Mkpojiogu and F.M. Kamal, 2015. Eliciting user satisfying requirements for an e-health awareness system using kano model. Proceedings of the 14th WSEAS International Conference on Computer and Conputational Science (ACACOS'15), April 23-26, 2015, WSEAS Press, Kuala Lumpur, Malaysia, ISBN:978-1-61804-297-2, pp: 156-165.

Hussain, A., E.O.C. Mkpojiogu and M.N.M. Nawi, 2016. Requirements model for an E-health awareness portal. Proceedings of the International Conference on Applied Science and Technology (ICAST'16) Vol. 1761, April 11-13, 2016, AIP, Kedah, Malaysia, pp: 020048-020048.

IEEE Std 830, 1998. IEEE recommended practice for software requirements speci?cations. Software Engineering Standards Committee of the IEEE Computer Society, pp: 11

ISO., 2011. IEC25010: 2011 Systems and software engineering-systems and software quality requirements and evaluation (SQuaRE)-system and software quality models. International Organization for Standardization, Geneva, Switzerland.

Kollanus, S. and J. Koskinen, 2009. Survey of software inspection research. Open Software Eng. J., 3: 15-34.

Laitenberger, O., E.K. Emam and T.G. Harbich, 2001. An internally replicated quasi-experimental comparison of checklist and perspective based reading of code documents. IEEE. Trans. Software Eng., 27: 387-421.

Mkpojiogu, E.O. and N.L. Hashim, 2017. Improving the quality of requirements work products using scoring rubrics-assisted reading. Proceedings of the 6th International Conference on Computing and Informatics, April 25-27, 2017, Universiti Utara Malaysia, Changlun, Malaysia, pp: 656-662.

Porter, A.A. and L.G. Votta, 1994. An experiment to assess different defect detection methods for software requirements inspections. Proceedings of the 16th International Conference on Software Engineering, May 16-21, 1994, IEEE Computer Society Press, Los Alamitos, California, USA., ISBN:0-8186-5855-X, pp: 103-112.

Porter, A.A., L.G. Votta and V.R. Basili, 1995. Comparing detection methods for software requirements inspections: A replicated experiment. IEEE. Trans. Software Eng., 21: 563-575.

Regnell, B., P. Runeson and T. Thelin, 2000. Are the perspectives really different? further experimentation on scenario-based reading of requirements. Empirical Software Eng., 5: 331-356.

Sandahl, K., O. Blomkvist, J. Karlsson, C. Krysander and M. Lindvall *et al*., 1998. An extended replication of an experiment for assessing methods for software requirements inspections. Empirical Software Eng., 3: 327-354.

Shull, F., I. Rus and V. Basili, 2000. How perspective-based reading can improve requirements inspections. Comput., 33: 73-79.

Sommerville, I., 2007. Software Engineering. 8th Edn., Addison-Wesley, Boston, MA.

Winkler, D., S. Biffl and K. Faderl, 2010. Investigating the temporal behavior of defect detection in software inspection and inspection-based testing. Proceedings of the 11th International Conference on Product-Focused Software Process Improvement, June 21-23, 2010, Springer, Limerick, Republic of Ireland, pp: 17-31.