2018 5th IEEE Uttar Pradesh Section International Conference on Electrical, Electronics and Computer Engineering (UPCON)

Semi-Automated System Based Defect Detection in Software Requirements Specification document

Nancy Vaish
CSE Department GCRG Group Of Institutions
Lucknow,India

vaishnancy048@gmail.com

Ashish Sharma
CEA Department GLA University
Mathura, India
ashish.sharma@gla.ac.in

Abstract— Software Requirement Specification is a document that defines capabilities of any software. Requirement phase plays an important role and it is the most critical phase among all SDLC phases as requirements about any software should be mentioned properly otherwise it will lead to incorrect product. Software requirement specification document consists of defects due to this and defects are need to be detected to save time and money. There are some reading techniques with the help of which defects can be identified. There are certain types of defects definitions on which basis defects are categorized accordingly. Previously defects are identified manually by reading document but this work is based on detecting defects semi-automatically which help in improvement of quality of document as well as developed software.

Keywords— Software Requirement Specification, Defect, Error, Checklist, IEEE standard

I. INTRODUCTION

The most important role in Software Development Life Cycle (SDLC) model is of requirement engineering phase. In this phase basically requirements are analyzed or customer needs are gathered. Success or failure of a project completely depends on this requirement engineering phase. Incomplete or wrong requirements leads to unwanted product by customer. Most of software projects cause failure due to insufficient requirements.

Requirement engineering phase includes requirement elicitation, requirement identification, requirement specification, requirement verification and validation. Our proposal is mainly considered with defect detection from requirement specification document.

Software Requirement Specification is a document which is prepared for the ease of customer as well as service provider to build a software project for describing requirement at a particular time prior to actual development of any software product. Software Requirement Specification is written in natural language to make easier to understand the defined functions and also constraints of a document. Software Requirement Specification is a document that consists of thorough description about how the system would perform. It consists of functional as well as non-functional requirements only; it does not contain any kind of design suggestions.

The requirement documents are written in natural language de-tailing about the developed software to be. It consists of all the external and internal requirements but it may consist defects. Previously detection of defects [9] from Software requirements specification document is done manually but it consumes lot of human effort and time, so there is need to build and maintain semi-automated system to detect defects from any document with the help of rules based classification to make it easier to perform.

This paper discusses about the proposal of semi-automation in detecting some defects for Software Requirement Specification documents needed to reduce the work done manually to save time as well as effort made by human being while inspecting any SRS document. This paper is divided in to five main sections: Firstly an introduction is done to make us understand about the area work and main concerns. Second, related work is carried out to give some knowledge about previous approaches. Third, proposed work is explained which is the main concern of this paper, how the work is performed, what dataset is used, how it is to be done etc. Then results are provided and finally conclusion of the work done and future scope is discussed to improvise further more.

II. RELATED WORK

A. Requirement Defect Detection Reading Techniques

There are some reading techniques with the help of which the defect detection is carried out in documents. Inspection is basic methodology which is basis of each technique while detecting defects from documents. The three main techniques which are known as main reading techniques are:

- *Checklist-based Reading technique* [4] [7]: M.E Fagan [4] described this CBR technique used in inspection methodology in 1970s. In this technique this is in the form of questions which is answered by the inspector if asked for a particular software to find any kind of faults present in it. Questions are general, not specific to any particular development and should be prepared in a single page. Answers to the question should be yes or no. It is difficult and non-systematic technique because inspectors are not provided required information on how to perform it easily. Checklist questions are limited as it may not focus on more domains.

- *Defect-based Reading technique* [7]: Porter et al. [3] developed this technique which is more refined version of Checklist-based reading technique as it focuses on specific kind of defects instead of general. It is more systematic and distinct as it defines scenarios which inspector follows while performing inspection of any document. Due to scenario based all types of defect or extra defects are not covered whereas in CBR all defects are covered.
- *Perspective-based Reading technique* [14]: Perspective-based Reading technique is non overlapping, structured way of examining faults from artifacts from perspective of different stakeholders or users [7]. It improves efficiency as compare to other techniques by minimizing overlap among the defects found by the reviewers. PBR technique provides proper understanding to inspectors in the form of scenarios on how to read and find the defect in the document.

B. *Requirement Error Taxonomy*

Walia et al. [5] proposed requirement error taxonomy or classified in error classes. Errors are categorized in three main classes i.e. People errors, Process errors and Documentation errors and then each class of error is further divided.

Errors which are caused due to people during preparation of requirements comes under people errors. Errors caused due to scanty process of requirement engineering and not selecting the right means for achieving goal falls in process errors category. Finally documentation errors are those which are caused due to making mistakes while organizing requirements specification document.

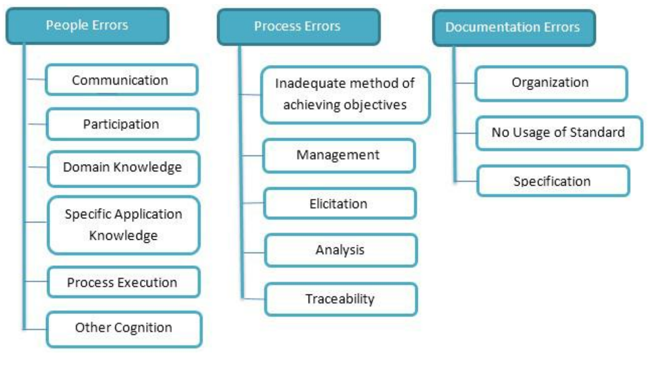


Fig 1: Requirement Error taxonomy [5]

III. Proposed Methodology

This section of paper defines the proposed methodology to detect defect semi-automatically in SRS documents. Previous approaches have certain drawbacks due to which automation is needed.

Checklist based reading technique [4] do not cover all parts of SRS documents and defects too. Defect based reading technique [7] focuses on various defects but not all parts of SRS documents and also do not look over reasons for emergence of defects.

Perspective based reading technique [7] do not consider all modules of the document. Although, previous methods exist that detect defects from document manually i.e. manual approach consumes lot of time and human effort. Manual approaches are always time consuming and waste efforts taken by human in detecting defects accordingly. So, there is need to automate the defect detection process from Software Requirement Specification document to save time and effort.

This proposal is based on rule based detection of defects so certain rules are made and rules are matched with the text content of SRS documents. For semi-automation we are considering mainly 3 defects which an SRS document contain i.e. Omission, Incorrect and Not Conforming to Standards which are explained in previous section.

The proposed model is shown in fig. 2.

This process of defect detection is explained as follows:

Step 1: Read SRS document in PDF format.

Step 2: Extract the text contents from SRS document which includes only text of document, no image is extracted.

Step 2.1: Check the text format of extracted text content which includes font checking. If in the full SRS document similar font is not found in full text then it display message with name Incorrect defect, further it increases count in total number of incorrect, otherwise count in number of defects do not increase.

Step 2.2: In the extracted text content two main rules are per-formed for detection of defects:

Step 2.2.1: If list of figures found then match figure name of full document with list of figures whether they are at defined page location, if not found on same page as defined then display message with defect name incorrect and increase count in total number of incorrect defects otherwise do not increase count.

If list of figures is not found in document display message with defect name Not Conforming to Standards.

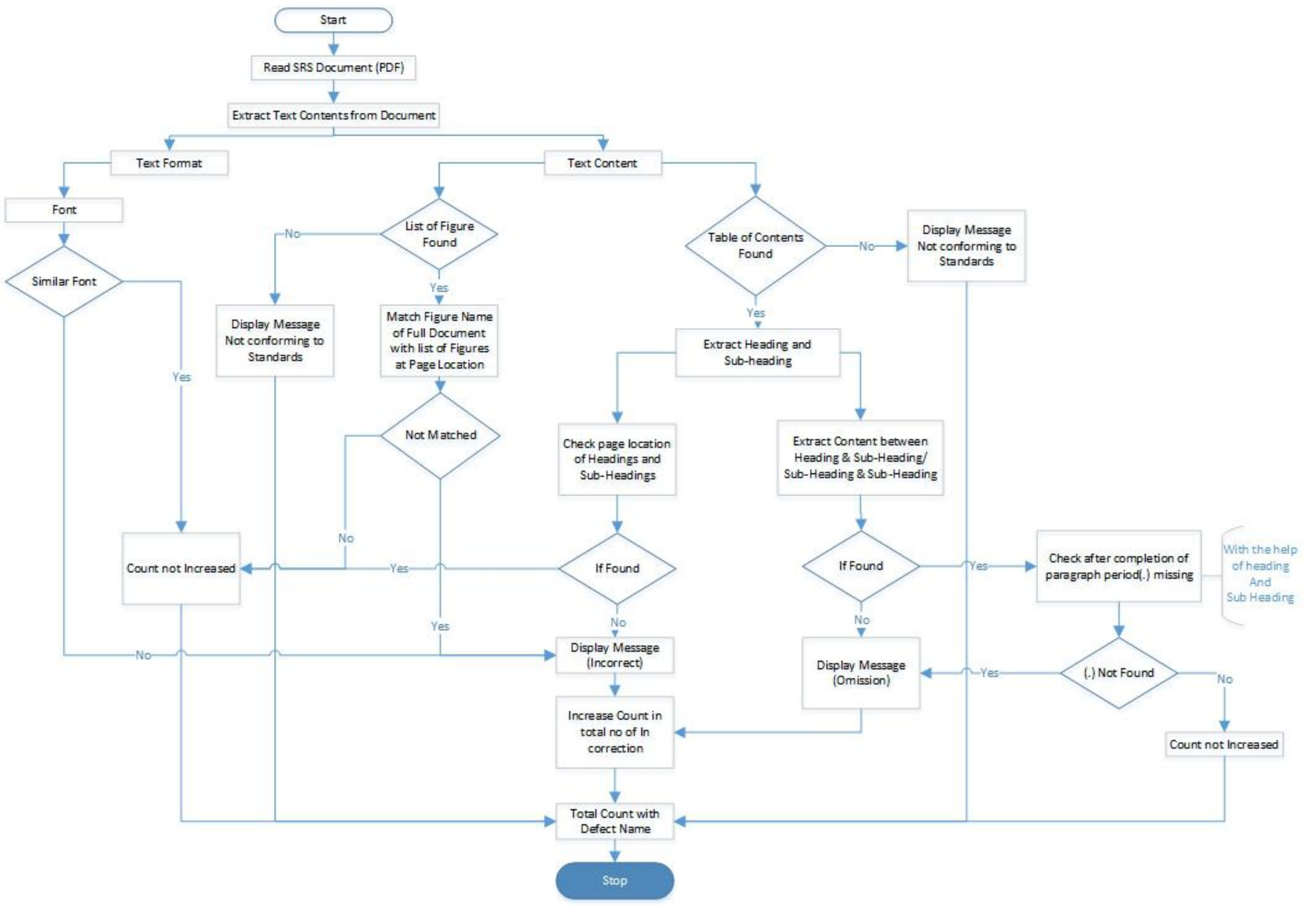
Step 2.2.2: Similarly if table of contents found in SRS document then extract headings and subheadings from it otherwise display message with defect name Not Confirming to Standards.

With the help of headings and subheadings check page location of each heading and subheading in full document. If exact location matches then do not increase count otherwise display message with defect name Incorrect and further increase in count in total number of incorrect defects.

With the help of headings and subheadings extract text content between heading-subheading and subheading-subheading. If con-tent is not found display message with defect name Omission and increase in count in total number of defects and if found then check after completion of each paragraph period (.) missing. If not found display message omission defect and if found do not increase count.

Step 3: At the end count total number of defects with its particular defect type name.

Step 4: Stop.



Start
Read SRS Document (PDF)
Extract Text Contents from Document
Text Format
Text Content
Font
Similar Font
Yes
No
List of Figure Found
No
Yes
Display Message Not conforming to Standards
Match Figure Name of Full Document with list of Figures at Page Location
Not Matched
No
Yes
Table of Contents Found
No
Yes
Display Message Not conforming to Standards
Extract Heading and Sub-heading
Check page location of Headings and Sub-Headings
Extract Content between Heading & Sub-Heading/ Sub-Heading & Sub-Heading
Count not Increased
Yes
If Found
No
If Found
Yes
No
Check after completion of paragraph period(.) missing
With the help of heading And Sub Heading
Display Message (Incorrect)
Display Message (Omission)
Yes
(.) Not Found
No
Increase Count in total no of In correction
Count not Increased
Total Count with Defect Name
Stop

Fig 2: Process Flow for Semi-Automatically defect detection in Software Requirement Specification document

IV. Result Analysis

The results shown here are showing the number of defects detected from each document of the dataset to show how many defects are detected of type Omission, Incorrect and Not Conforming to Standards defect [7]. Comparison results with other previous approaches are also shown. The proposed methodology is implemented on the HP machine having configuration i5 processor with 4GB RAM on Windows platform with the help of Visual Studio 2013 using .Net framework 4.5 with C#, Window based application.

A. Dataset Used

The dataset here consists of four SRS documents on different topics:

The Online Shopping Mall (OSM) [13]: The Online Shop-ping Mall is a site to enable retailers to create online shops, help customers to browse through the shops, purchase on-line things without actually visiting the shop and providing feedback for the product or service. The application also provides a system administrator which has the ability to approve and reject requests for new shops and for maintaining lists of more categories of shops.

The Massive Multiplayer Online Role Playing Game (MMORPG) [15]: MMORPG is an online game to encourage cultural tourism in Turkey. The player travels over the cities in order to perform missions and collect coins and the gold dispersed over some secret places.

The Parking Garage Control System (PGCS) [11]: PGCS is a system that controls and manages entries and exits of parking garage. The system allows or discards entries into the parking garage depending on the number of available parking spaces.

The ABC Video System [12]: is a system that automates and reports the video rental and return processes.

The process of defect detection using semi-automated approach was when applied to the above dataset gave improvement in the process of defect detection as compared to manual approaches. The results were analyzed for each type of defect which we are considering in our semi-automated approach. The table 1 is shown to display the results of number of detected defects of proposed as well as previous approaches i.e. Checklist-based reading (CBR), Defect-based reading (DBR), Perspective-based Reading (PBR). The table of results shown below for 3 defect types are from the same dataset to get an overall idea of the process.

This table consists result analysis of our proposed approach as well as comparison results with other approaches. Parameter is number of defects which we have used to show the comparison between approaches. Three defects are taken in our proposal that

Table 1: Results Analysis

Defect Type		CBR	DBR	PBR	Proposed
		(Manually calculating defects approaches)			(Semi-Automated approach)
Omission	SRS 1	10	14	18	21
	SRS 2	9	13	17	27
	SRS 3	5	9	10	10
	SRS 4	0	0	4	5
Incorrect	SRS 1	5	6	3	12
	SRS 2	8	7	5	32
	SRS 3	4	5	3	4
	SRS 4	2	3	1	4
Not Conforming To Standards	SRS 1	0	0	0	2
	SRS 2	0	0	0	1
	SRS 3	0	0	1	2
	SRS 4	1	1	0	2

are Omission, Incorrect and Not Conforming to Standards. Four SRS which we have explained in dataset are used for validating our approach. In table 1, SRS1 (Online Shopping Mall) [13], SRS2 (Massive Multiplayer Online Role Playing Game) [15], SRS3 (Parking Garage Control System) [11] and SRS4 (ABC Video System) [12]. The comparison has been performed on the following aspect: the total number of detected-defects.

From results table 1 we can conclude that the proposed semi-automated approach is most efficient in detecting defects in terms of the number of detected-defects followed by the Perspective-Based-Reading (PBR). The Defect-Based-Reading (DBR) is ranked third in terms of the number of detected defects and Checklist-Based-Reading (CBR) is ranked number 4.

To conclude the efficiency of each technique in detecting each type of defects, we have to examine the areas of differences and commonalities among the given approaches. We are explaining these results collectively of four SRS document of each defect type. Firstly, for the omission defect type, the semi-automated approach is the best as it detected 63 defects, as compared to other approaches. The PBR follows the semi-automated approach with 49 omission defects. The DBR detects 36 defects. In the third rank is the DBR and in the fourth the CBR with 24 number of defects.

Regarding the second type of defect i.e. incorrect defect, the semi-automated approach is far better of the rest of the manual approaches. The CBR (19 defects) and the DBR (21 defects) are almost equal in detecting the incorrect defects. The PBR technique detects only 12 defects and is the worst in detecting the incorrect defects.

Now last defect which we have considered is Not Conforming to Standards defect, the proposed approach provides remarkable results as compare to other approaches from each SRS document i.e. number of detected defects are 7. All other approaches CBR, DBR and PBR found only one defect as result. Based on the above analysis, the proposed semi-automated approach is better than other approaches, followed by the PBR approach. Then, in the third rank comes the DBR approach and in the fourth rank the CBR technique comes.

V. Conclusion And future work

The paper is actually concerned with the process of defect detection from any Software Requirements Specification document. Chapter 1 gives the Introduction about the topic and research area, discusses issues and motivation, and also gives a brief about problem statement. Chapter 2 gives the related work in this domain so that we can understand about previous approaches. Chapter 3 gives an architecture of the proposed process flow designed to overcome the challenges faced solving the issues identified in the manually defect detection approaches. Chapter 4 gives the result analysis of the implemented proposed methodology to show how our approach is better than the previous approaches. It can be concluded that the results of the proposed approach gives much more better results as compared to the previous approaches due to the automation.

The proposed approach in this paper works on mainly three defect types Omission defect, incorrect defect and Not Conforming to Standards defect. For future work, there are more type of defects like Ambiguous, Inconsistent, Superfluous, Not-implementable and Risk-prone [7] which can be included in semi-automatic detection. To improve its efficiency more SRS can also be taken to provide better results.

References

[1]G. Sabaliauskaite, F. Matsukawa, S. Kusumoto and K. Inoue, "An Experimental Comparison of Checklist-Based Reading and Perspective-Based Reading for UML Design Document Inspection", Presented at the Proceedings of the *International Symposium on Empirical Software Engineering,*2002.

[2]B. Anda and D.I.K. Sjoberg, "Towards an inspection technique for use case models", Presented at the Proceedings of the *14th International Conference on Software Engineering and Knowledge Engineering* (SEKE'02), Ischia, Italy, 2002.

[3]T. Thelin, P. Runeson and C. Wohlin, "An experimental comparison of usage-based and checklist- based reading", *IEEE Trans Softw. Eng.*, 29, 687–704, 2003.

[4]DA. McMeekin, B. R. von Konsky, E. Chang and D J. A. Cooper, "Checklist based reading's influence on a developer's understanding." Software Engineering, 2008. ASWEC 2008. *19th Australian Conference on*. IEEE, 2008.

[5]G.S. Walia and J.C. Carver, "A systematic literature review to identify and classify software requirement errors", *Inform. Softw. Technol.*, 51, 1087–1109, 2009.

[6]I.L. Margarido, J.P. Faria, R.M. Vidal and M. Vieira, "Classification of defect types in requirements specifications: Literature review, proposal and assessment", Presented at the *6th Iberian Conference on Information Systems and Technologies (CISTI),*Chaves / Portugal, 2011.

[7]A. A. Alshazly, A. M. Elfatatry and M. S. Abougabal, "Detecting Defects in Software Requirements Specification", *Alexandria Engineering Journal*, 53, 513-527,2014.

[8]IEEE, "IEEE Recommended Practice for Software Requirements Specifications, IEEE Std. 830–1998 (Revision of IEEE Std. 830–1993)", pp. 1–40, 1994.

[9] IEEE, "IEEE Standard for Software Reviews, IEEE Std 1028–1997", pp. i–37, 1998.

[10]A.A. Porter and L.G. Votta, "An experiment to assess different defect detection methods for software requirements inspections", Presented at the Proceedings of the *16th international conference on Software engineering*, Sorrento, Italy, 1994.

[11]R.J. Schneider, "Requirements document for a parking garage control system (PGCS)", Private Communication, pp. 1–18, 1996.
[12]R.J. Schneider, "Requirements Document for ABC Video System", Private Communication, pp. 1–14, 1996.

[13]S. Ray, S. Bhattacharya, S. Shaw and S. Sett, "Online Shopping Mall Project Report", Department of Information Technology, B.P.PODDAR Institute of Management and Technology, West Bengal University of Technology, pp. 1–50, 2009.

[14]M. Ciolkowski, "What do we know about perspective-based reading? An approach for quantitative aggregation in software engineering", Presented at the Proceedings of the 2009 *3rd International Symposium on Empirical Software Engineering and Measurement*, 2009.

[15]C. Kilcioglu, M. Degirmenci and U.C. Buyuksahin, "Software Requirements Specifications: Massively Multiplayer Online Role Playing Game Project (MMORPG)", Department of Computer Engineering, Middle East Technical University, pp. 1-32, 2010.

[16] G.S. Walia and J.C. Carver, "A Systematic Literature Review to Identify and Classify Software Requirements Errors", University of Alabama MSU-071207, 2007.

[17] R. Chughtai, "A Domain-Specific approach to Verification & Validation of Software Requirements", M.Sc. thesis, Arizona State University, 2012.