J. Appl. Environ. Biol. Sci., 6(4S)137-146, 2016

ISSN: 2090-4274
Journal of Applied Environmental and Biological Sciences
www.textroad.com

Quality Software Requirement Specification(SRS) and suitable SDLC Leads to Quality Software

Sadia Ashraf[1], Rashid Khan[1], Dr. Khalid Iqbal[2], Mr. Rehan Chohan[2]

[1]Department of Computer Science, COMSATS Institute of Information Technology Attock And wahcantt.
[2]Department of Computer Science, COMSATS Institute of Information Technology Attock

Received: January7, 2016
Accepted: March 22, 2016

ABSTRACT

SDLC is a process of developing software project which have the phases of requirement, design, development, testing and maintenance. The first phase is about requirement gathering, elicitation, negotiation and specification which is the most important, lengthy time consuming and error prone task. Most of project failure is due to less concentration on quality of requirement specification. Our project will easily be able to complete within given time and cost if and only if the requirement is quite clear to us. For the collection of best requirement for development many researchers are working. They have defined many techniques and tools to improve the quality of SRS. We also discuss Requirement Specifications Language (RSL) which is a language for gathering quality requirement that are technical concerns of the system that affects the project throughout the lifecycle. RSLingo is an information extraction approach based on linguistic patterns, which follows a multi-language strategy which is based on two languages (RSL-PL and RSL-IL) and mapping between them.

This paper emphasis that how Quality Requirement Specification are playing an important role in the success of any project in the context of SDLC and quality attributes of a quality software. We focus on automated Requirement Specification techniques and tools which are used for the improvement of SRS. A literature review on the SDLC, Quality Requirement Specification and on the objective" Quality software" also examined in three sections. Furthermore we also found the strength and limitations of these sections and critically evaluated the reviewed literature with the help of a comprehensive diagram by joining these sections. This literature reviews its critical evaluations and summarization provides a concise knowledge and its different relations to academic circles on automated Requirement Specification techniques. Finally, in future on the bases of these relations, tools features and Requirement technique can be enhanced.

KEYWORDS: SDLC (Software development life cycle), Natural Languages, Requirement Specification linguistics (RSLingo), Requirement Specification linguistics (RSL-IL) Intermediate language and RSL-PL (Requirement Specification Patterns language)

1. INTRODUCTION

To achieve the goal of quality software we need to study software engineering process in multi dimensions. There's no single best way to view software quality. Many achievable aspects of software quality are functionality, Reliability, Reusability, portability, Maintainability and Efficiency. The three main aspects of software quality are Structural, functional and process quality. [16] As the entire field of software quality model is diverse and fuzzy in nature. To cure it, we accounts and collects critique of existing models by analyzing purposes and usage scenarios of these models which are based on literature and author's experiences. Then the general requirements are derived for quality model. . A three level schema (DAP) is proposes for the classification of quality model w.r.t proposed purpose. DAP is the combination of three models (Definition, Assessment and Prediction model) which test a quality model based on model purpose.[8]

As higher rate of project failure is common to software development industry.

There are many reasons and one is improper selection of SDLC model [18]. Heavyweight and lightweight are the two basic classes to which SDLC model falls. As Waterfall, Spiral, Iterative, RUP (Rational Unified Process) are the traditional methodologies and categorized as Heavyweight. Feature Driven Development, Scrum and XP etc are lightweight and known as agile methodologies [19]. The System Development Life Cycle framework provides a sequence of activities for system designers and developers to follow for developing software. Understanding the basic concepts of software development methodologies is necessary to enable evaluation of best software development life cycle (SDLC) methodology. [17]

Most of project failure in the field of software engineering is due to less concentration on software requirement specification (SRS) [1] [11]. The most important sub class of requirement is "quality requirement" that determine the characteristics of large set of systems described as performance, security, usability and scalability. Many organizations produce their products/projects with less

* **Corresponding Author:** Sadia Ashraf, Department of Computer Science, COMSATS Institute of Information Technology Attock And wahcantt. sadi.ash333@gmail.com

attention and implicit understanding of quality requirement and results as failure product/projects. Some quality SRS characteristics are Unambiguous, Correct, Complete, Consistent, Modifiable, Traceable and Verifiable [11]. Commonly Natural languages are popular for SRS's documentation but have many limitations and causes quality deficiency as in the form of "incorrectness", "inconsistency" "incompleteness" and "ambiguousness".[1][2][3][4] To cope up these characteristics and for removal of defects many techniques or approaches are using.

The one approach is "An automatic validation approach with a proper tool set (SpecQuA software)" [1]. RSLingo is an information extraction approach based on linguistic patterns, which follows a multi-language strategy which is based on two languages and the mapping between them. 1st language is RSL-PL; an extensible language for dealing with information extraction from requirements written in natural language [4] and 2nd is RSL-IL, a formal language with a fixed set of constructs for representing and conveying RE-specific concerns [4] [5]. A novel tool in the area of RE, The basic objective is to facilitate RE view of ProjectIT approach to integrate RE with the MDE processes also introduces a CSCW (Wiki based) platform. [9]

The paper is consisted on three sections. Section I is about critical evaluation of Quality SRS, Section II is about SDLC phases, classes and models and Section III is about the Quality Software. Where software quality is checked in the context of quality attributes which can get by the proper selection of SDLC model selected on the bases of Requirements. So we deduce Quality is dependent on SDLC and SDLC is dependent on Requirements. All these dependencies are handled or deal by the different stakeholders from end users to development team. Before the conclusion some attributes which have purpose, methodology, characteristic and limitations in tabular form are given. This literature review gives a comprehensive diagram which shows that how these sections are interrelated with each other and combined and affects the quality of software. So we critically evaluated that Quality software is a multidimensional field and needs improvement in all the sections especially the one dimension which is the stakeholders.

2. LITERATURE REVIEW

The paper is consisted on three sections. The literature review is given as following.

A. Quality SRS

Most of project failure in the field of software engineering is due to less concentration on software requirement specification (SRS) [1] [11]. The most important sub class of requirement is "quality requirement" that determine the characteristics of large set of systems described as performance, security, usability and scalability. Many organizations produce their products/projects with less attention and implicit understanding of quality requirement and results as failure product/projects. Some quality SRS characteristics are Unambiguous, Correct, Complete, Consistent, Modifiable, Traceable and Verifiable [11].

Writing a good and quality SRS is the key feature for the success of a project. For this purpose a lot of techniques are used and can be produced quality product. As there are many resources for gathering knowledge and are used for defining requirements, so there is no lack of knowledge and methods. But the problem is disobedience the defined definition so we should improve SRS not only by making good documentation but also in field work. Today systems are complex and small in size so requirement specification is big challenge. To face and compete these challenges IEEE is an excellent source for definition of system and software specification. A lot of standards are available for different kinds of systems. [12]

Commonly Natural languages are popular for SRS's documentation but have many limitations and causes quality deficiency as in the form of "incorrectness", "inconsistency" "incompleteness" and "ambiguousness".[1][2][3][4][22] Natural languages (NL) are traditionally used for the documentation of software requirements. Natural languages are being easy to understand and having high expressivity, this leads to well-known requirements quality problems.

To mitigate these problems introduces RSL-PL, is a pattern definition language based on linguistic patterns. The purpose of the RSL-PL to define linguistic patterns that enable one to deal with the linguistic concerns of the textual representations of NL requirements that improves the quality of requirements specifications and productivity of requirements engineers. RSL-PL is useful language for the automation of the requirements linguistic analysis within the information extraction process [4].

To cope up these characteristics and for removal of defects many techniques or approaches are using. The one approach is "An automatic validation approach with a proper tool set (SpecQuA software)" .The purpose is to increase quality of "Requirement Specifications" that are technical concerns of the system that affects the project throughout the lifecycle. RSLingo Approach where requirements are given in RSL-IL (Requirement specification language is an Intermediate language) automatically requirements are extracted from natural language specification or authored by the user [1].

RSLingo can be used to take and validate the software requirement specification. The purpose of this technique is to automate different manual performed tasks. RSLingo is an information extraction approach based on linguistic patterns, which follows a multi-language strategy which is based on two languages and the mapping between them. 1st language is RSL-PL; an extensible language for dealing with information extraction from requirements written in natural language [4] and 2nd is RSL-IL, a formal language with a fixed set of constructs for representing and conveying RE-specific concerns [5].

RSLingo approach (conversion informal requirements to formal) by considering that the requirements are represented in RSL-IL (Requirement specification language)[2]. RSL-IL is a formal language with a fixed set of constructs for representing and conveying

RE-specific concerns.[3]

Perversely to other approaches, this uncoupling allows one to deal with requirements as "white-box "items. Thus, RSLingo enables the automation of some verification tasks that prevent common requirements quality problems.

The RSLingo approach has two distinct stages definition at process-level and usage at project-level [3] RSLingo allows one to automatically verify some quality criteria and generate approving representation. RSLingo gives a deeper view of system as compare to other specification techniques. However it has more potentiality for quality requirement specifications [6].

A novel tool in the area of RE, The basic objective is to facilitate RE view of ProjectIT approach to integrate RE with the MDE processes. The paper also introduces a CSCW (Wiki based) platform. Its purpose is to give improved interaction and flouting the natural limitations between stakeholders by obliging different background and individual skills. During requirement development process CSCW ease the communication, cooperation and negotiation of stakeholders. The benefits of the model and platform can be combined with natural language processing techniques, which are already experienced by ProjectIT's CASE tools [7] to power to act effectively for quality and rigor of requirement engineering workflows. Wiki based tool chain the requirement management thus covering all RE activities. Another feature of supreme importance is to discover traceability mechanisms at a high quality granularity level. Future planning work is to tackle the requirement management support both in terms of ProjectIT approach and respective supporting tool. Many public and teamwork issues regarding RE can be proposes and explore. [9]

A. Software Development Life Cycle SDLC:

As higher rate of project failure is common to software development industry. There are many reasons and one is improper selection of SDLC model. Best quality model software can be developed by selecting appropriate or best suitable model within budget and time. Best suitable model selection can be gained by studying the considerable characteristics of the project proposed by different researchers. If the developer is sure about the requirements of project and their suitability to SDLC model then model selection is very easy. The one trade-off in model selection is the tie of two conditions as requirement suggests one model and theoretically it may suggest another model. [18] Heavyweight and lightweight are the two basic classes to which SDLC model falls. As Waterfall, Spiral, Iterative, RUP (Rational Unified Process) are the traditional methodologies and categorized as Heavyweight. Feature Driven Development, Scrum and XP etc are lightweight and known as agile methodologies. Different models are used by different organizations depending upon their needs [19].

Methodology analysis for successful comparison of SDLC models and deep study of various tools, techniques and methodologies is given that extract the most simple and easy method for the comparison of SDLC method. SDLC provide a layout for software development and a proper employment of SDLC permit the management to normalize the complete development plan of the software. Comparison between SDLC models is a complex task because these models give a theoretical layout and there is no mathematical proof or any device to do it. Line of code method (LOC) is used when multiple SDLC are used to develop a software. COCOMO is the most simple and easiest mathematical method for comparison.

From many SDLC models the best suited is entirely dependent on developer end and client end constraints. [20] A hypothetical model, explains the interaction of users and developer that is converted into a three dimensional model as user, owner and the developer in a usual SDLC model. Hypothetical model the novel approach attempts to remove the defects of conventional SDLC "one size fits all". SDLC enhanced the control of management by splitting work into different manageable categories, but ignored the change, release and incident management which are the big issues. To resolve this hypothetical model is introduced. The limitation of addressing these issues under the project management are the discussion of these at surface level not on ground level. [21]

According to the problem, different types of models such as water fall, iterative and spiral models. These models have the stages of Systems Investigation, Analysis, Design, Implementation and at last evaluate the results of the solutions.

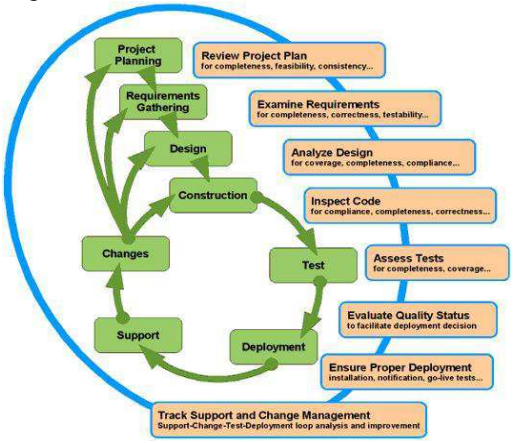

Figure 1.1

For system designers and developers SDLC structure provides a chain of proceedings to follow for developing software. It is necessary to understand the core concepts of software development methodologies that enables evaluation of best software development life cycle (SDLC) methodology. [17]

B. Quality Software

There's no single best way to view software quality. Many achievable aspects of software quality are functionality, Reliability, Reusability, portability, Maintainability and Efficiency. Some conventional aspects of the software quality are discussed here. The three main aspects of software quality are Structural, functional and process quality. Each one has its own value and detail. The 1St aspect of software quality is functional quality that defines the software performs correctly the tasks which are the demand of users and proposed for the system are. The 2nd aspect is structural quality that is about the code is itself well structured. Structural quality is hard to test as compare to functional quality. Structural and functional quality aspects are usually gets a keen attention in the dialogue of software quality. The 3rd and critical aspect is process quality is also has a significant place. The development process quality is considerably affected by the value which is received by the users like sponsors, development teams and the end users. So types of users have a chance to improve the given aspect of software quality. Given how important software has become to the world, it's hard to overemphasize the importance of software quality. Recognizing the broad scope this idea encompasses is a useful step along this path. [16]

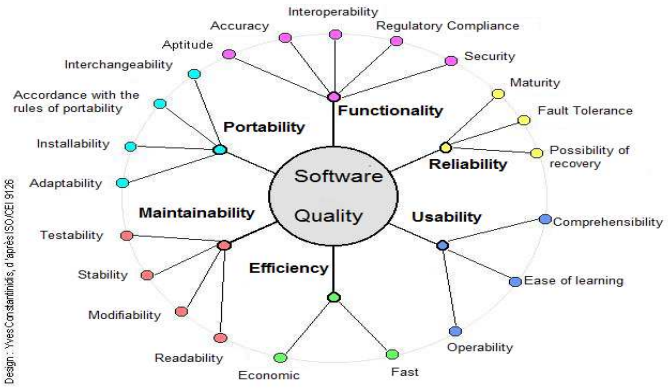

Figure 1.2

As the entire field of software quality model is diverse and fuzzy in nature. To cure it, we accounts and collects critique of existing models by analyzing purposes and usage scenarios of these models which are based on literature and author's experiences. Then the general requirements are derived for quality model. On the bases of these requirements we can support existing quality models and can also evaluate for a given context. Guidance for Further development in quality models can also be deduced by these requirements.

A three level schema (DAP) is proposes for the classification of quality model w.r.t proposed purpose. DAP is the combination of three models (Definition, Assessment and Prediction model) which test a quality model based on model purpose.[8] Definition models are used in many phases of software development process[24]. They specify requirements and quality attributes for planned software systems and thus constitute a method to agree with the customer what quality means [24, 25].

A framework that gives structure to the discovery, reporting, and measurement of defects and software problems which are found by the defect finding activities and primary problem and also integrates them. On the bases of this layout, measureable common attributes are identified and organized to these activities. The attributes are used with checklists on the basis of observation how it is using and supporting forms to converse the definition and specification for defect measurement and problem. An illustration of checklist and supporting forms can be used to decrease the misreading of measurement results and can be applied to address the information needs of different users.

Just it is as clear that the quality of the software is inversely proportional to the number and frequency of problems and defects associated with a software product. The direct measurements of software processes are the Software problems and defects. These quantitative measures portray the change direction abruptly in defects and problem discovery, reparation, response to customers and process and product imperfection. Many significant software qualities attribute factors, and criteria- reliability, correctness, completeness, efficiency, and usability can be quantified among each other's by using problem and defect measurement. [15]

The Aspect Oriented Software Development (AOSD) is another framework of representing early specification of non-functional requirements. Yet, a regular and similar vision of the AOSD jargon is at a standstill missing. Given work goal is to incorporate AOSD concepts, the novel standard of ISO/IEC 25030 on quality requirements of software, and typical requirements engineering design.[14]

The basic results of this study are requirements, Aspects, Software Quality (REASQ) and conceptual model. The REASQ model and ontology is a useful tool at early stages of development. The UML is used to state the core result of this study. Modeling concepts

formalized into three relative ontologisms, representing the scope of aspect-orientation, software quality and requirements engineering. In aspect oriented engineering process these ontology's can play the role of umbrella to specify the quality requirement.[14]

A quality model suitable for such a purpose, through the relative evaluation of existing quality models and their relevant support for Software Quality Engineering. A quality model caters the structure towards a definition of quality. A quality model that is to be used as the foundation for the definition of quality requirements should help in both the specification of quality requirements and the evaluation of software quality. In other words, it should be usable from the top of the development process to the bottom and from the bottom to the top. [10]

3. Critical Evaluation

The most important thing which primarily affects the quality of software is requirement gathering process. Means Software quality is dependent on quality requirement specification. Requirements are gathered through many techniques and tools but quality requirements can only be gathered when quality attributes are clearer to client, software engineer and other stakeholders.

The important thing one which secondarily plays an efficient role in the success of software is the selection of an appropriate SDLC model. SDLC model is also dependent on requirement because model selection is based on the nature of requirements As higher rate of project failure is common to software development industry. There are many reasons and one is improper selection of SDLC model. [18]

As the entire field of software quality model is diverse and fuzzy in nature. To cure it, we accounts and collects critique of existing models by analyzing purposes and usage scenarios of these models which are based on literature and author's experiences [8]

Best quality software model can be developed by selecting appropriate or best suitable model within budget and time. Best suitable model selection can be gained by studying the considerable characteristics of the project proposed by different researchers. If the developer is sure about the requirements of project and their suitability to SDLC model then model selection is very easy [18]

Most of project failure in the field of software engineering is due to less concentration on software requirement specification (SRS) [1] [11]. So we critically evaluated that Quality software is a multidimensional field and needs improvement in all the sections especially the one dimension which is the stakeholders. Just it is as clear that the quality of the software is inversely proportional to the number and frequency of problems and defects associated with a software product [15]

4. A multidimensional view of Quality Software through diagram

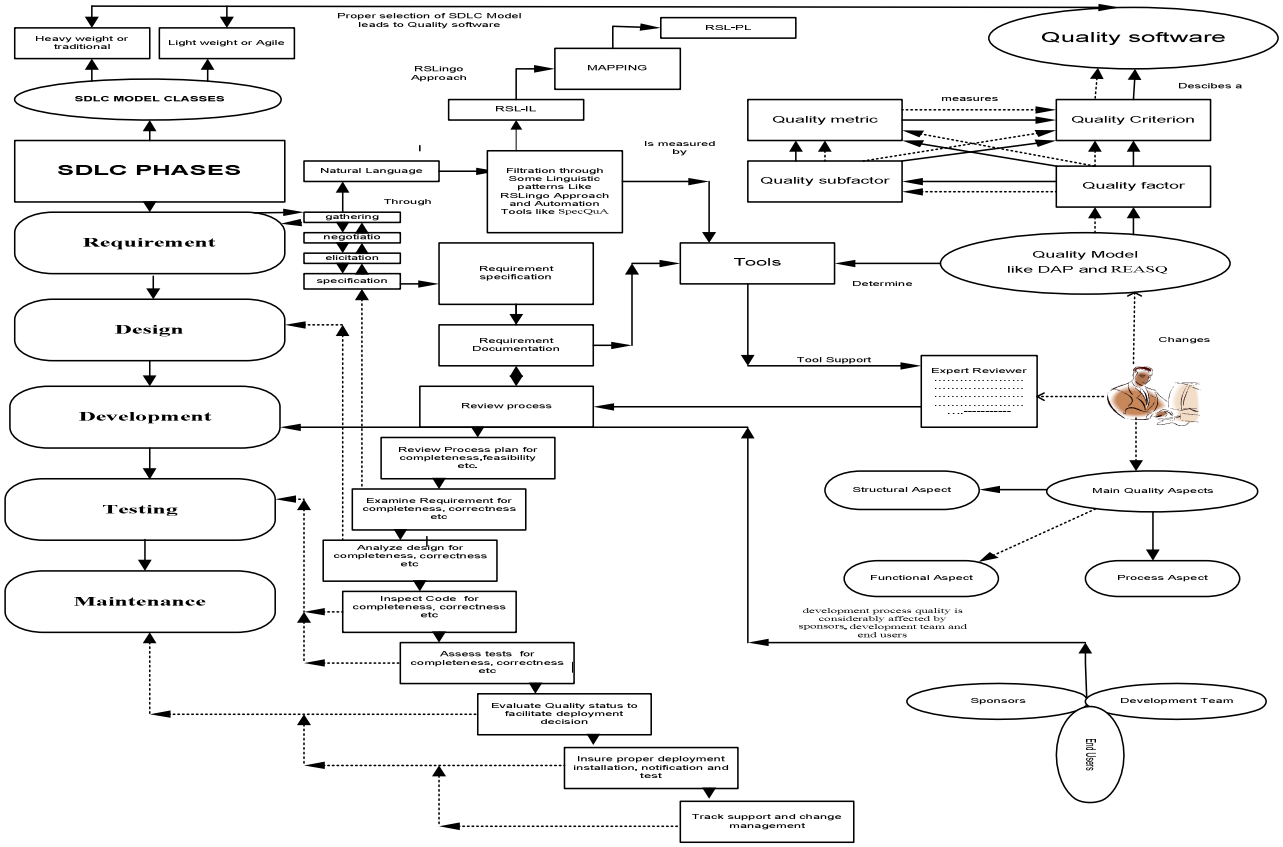

Figure1.3

Figure 1.3 describes a multidimensional view of quality software. SDLC, Quality SRS, Quality models and stakeholders are the different dimensions which affects the quality of software. When quality software is developed the first thing which affects the quality of software are the requirements and the selection of SDLC model. Secondly Stakeholders also affect the development phase of SDLC model. Quality SRS can be documented after the filtration of requirements manually or through tools and for this purpose many automated techniques like RSLingo approaches are used. An expert reviewer takes its reviews with the help of quality models and by following review process. Quality models also decided the suitable tools and these models are tested through the quality metrics. Reviewer also tests it with the help of different quality aspects. These all dimensions are interconnected and their selection is dependent on the previous selection of each dimension. So each dimension can be tested and traced in quality software.

5. Summary of all sections A, B and C in tabular form

Section A. Table 1.1

	Author	Purpose	Method	characteristics	limitation	Advantage	Tool
	Rodrigues Silva et al [1]	to increase quality of "Requirement Specifications".	RSLingo Approach , use of a tool support	a tool support to increase the quality of SRSs, productivity and validation	Requirement gathering again human-intensive, time consuming and error prone. Here no guarantee of quality of requirement specification	Tool support increases the quality requirement specification	SpecQuA software
	Silva, Rodrigues et al [2]	to increase quality of "Requirement Specifications	Extension of RSLingo approach with proper toolset SpecQuA (specific ation quality analyzer) software	requirements are represented in RSL-IL (Requirement specification language an intermediate language) automatically extracted from natural language specification	by using Tool and RSL-IL we can alleviate the quality problems but cannot be sure to illuminate these	RSLingo approach (conversion informal requirements to formal) by considering that the requirements are represented in RSL-IL (Requirement specification language an intermediate language) automatically extracted from natural language specification or authored directly by user.	SpecQuA (specificatio n quality analyzer)
	Ferreira , David, Rodrigues and Silva et al [3]	To automate different manual performed tasks.	An approach RSLingo is an information extraction approach based on linguistic patterns, which follows a multi-language strategy which is based on two languages and the mapping between them	RSLingo enables the automation of some verification tasks that prevent common requirements quality problems	Here no guarantee of quality of requirement specification	Contrarily to other approaches, this decoupling allows one to deal with requirements as "white-box "items, enabling a deeper understanding at a semantic level.	SpecQuA (specificatio n quality analyzer)
	Ferreira , David, Rodrigues and Silva et al [4]	to define linguistic patterns that enable one to deal with the linguistic concerns of the textual representations of NL requirements that improves the quality of requirements specifications and productivity of requirements engineers	RSL-PL language in defining new linguistic patterns and coping with different requirements writing styles, allowing it to be adapted and reused in different application scenarios	RSL-PL is useful flexible allowing a pattern to be adapted and reused in different application scenarios	It is dependent on tool. Still critical and not gives high quality because it is extracted from natural language that have many quality problems and	RSL-PL is more better way for quality requirement specification as compare to traditional method like natural language.	

	Ferreira , David, Rodrigues and Silva et al [5]	RSL-IL Requirement specification language is an Intermediate language which was proposed and designed for the back end control mechanism of RSLing approach.	RSL-IL Requirement specification language is an Intermediate language represents requirements in a stable and formal form tractable by a computer with the help of a tool.	Improves the quality of requirements also predict future research paths of usability concerns of RSL-IL in cognitive alignment and in concrete syntax. Further research is in making rules and heuristics to automate the quality of requirement specifications.	If minimal constructs internally organized as viewpoints not write/ extract properly the required quality can't be obtained	RSL-IL is consisted on minimal constructs internally organized as viewpoints give a way to tackle the requirement formalization problem	
	Ferreira , David, Rodrigues and Silva et al [6]	to increase quality of "Requirement Specifications".	RSLingo technique is the combination of two sub languages RSL-IL and RSL-PL and mapping between them.	RSLingo allows one to automatically verify some quality criteria and generate approving representation. RSLingo gives a deeper view of system as compare to other specification techniques.	Dependent on tool required skill and experience as compare to traditional method.	However it has more potentiality for quality requirement specifications. Future work can be done in a controlled environment like Laboratory case studies for the validation of RSLingo approach to gain the accurate requirements from users.	
	Ferreira , David, Rodrigues and Silva et al [7]	to support the functional requirement specifications which are written in controlled natural language	The ProjectIT it emphasizes **on** tool support for RE process which is usually performed manually produces error prone and less cost effective results for modern complex systems.	For the validation and generation of early design draft from requirement specification suitable tool support is provided by this approach	Without tool it will be difficult to integrate RE with the Model-Driven engineering prototype	As requirements are the foundation for each project and the activities performed here surly benefit from consistency checking of the requirement specifications	ProjectIT's CASE tools
	Ferreir, David, Rodrigues and Silva et al [9]	The basic objective is to facilitate RE view of Project IT approach to integrate RE with the MDE processes, to give improved interaction. CSCW ease the communication, cooperation and negotiation of stakeholders	Introduces a CSCW (Wiki based) platform. Its purpose is to give improved interaction and flouting the natural limitations between stakeholders by obliging different background and individual skills	. The benefits of the model and platform can be combined with natural language processing techniques, which are already experienced by Project IT's CASE tools [7] to power to act effectively for quality and rigor of requirement engineering workflows	It is dependent on tool	Another feature of supreme importance is to discover traceability mechanisms at a high quality granularity level.	CSCW Wik based tool
	Japenga, Robert et [12].	Purpose is writing a good and quality SRS that is the key feature for the success of a project.	The IEEE standards (www.ieee.org) are an excellent source for definitions of System and Software Specifications	Today systems are complex and small in size so requirement specification is big challenge. To face and compete these challenges IEEE is an excellent source for definition of system and software specification.	But the problem is disobedience the defined definition so we should improve SRS not only by making good documentation but also in field work. Today systems are complex and small in size so requirement specification is big challenge.	A lot of standards are available for different kinds of systems	

	Goldman, James Abraham, and Song al [13].	to present an approach to prepare SRS with Use Cases. The Use Case approach has become a de-facto Standard for capturing functional requirements.	Classification technique is used to employ it for the identification and management of the Use Cases	The prescribed method provides an additional facility for the analyzers in the preparation of a standards amenable SRS document by averting unnecessary specification effort and reduction of cognitive load through reduction	no concrete techniques to identify and link use cases to sections of the SRS.	Functional requirements can be gathered by using an approach of use cases, that has become a de-facto standard. The standard of IEEE Std.8301998 provides a template for software requirements documentation	

Section B. Table 1.2

s. no	Auther	Purpose	Method	characteristics	limitation	Advantage
11	Rajnish, Ranjana, and VYAS et al [11]	Purpose of paper is to aware the stakeholders when they specify the requirement must understand past ambiguous requirement to avoid defects in upcoming time. The aim is to focus on the relation between requirements quality and project outcomes.	Various statistical analysis Techniques were applied over the SRS quality data and project outcomes.	Some quality SRS characteristics are Unambiguous, Correct, Complete, Consistent, Modifiable Traceable and Verifiable Requirements are important and provide many opportunities to further strengthen and improve what we are doing	what happens when requirements are not Managed?	
12	Seema, Sona et al [17]	According to the problem we use different types of model of SDLC	Use different types of model of SDLC like water fall, iterative, spiral etc. And have following stages	Provides a sequence of activities for system designers and developers to follow for developing software	Each model has its own limitations on different stages	The stages give a proper layout to each model
13	Clara, V. Therese et al [18]	Best quality model software can be developed by selecting appropriate or best suitable model within budget and time.	Best suitable model selection can be gained by studying the considerable characteristics of the project proposed by different researchers	If the developer is sure about the requirements of project and their suitability to SDLC model then model selection is very easy.	The one trade-off in model selection is the tie of two conditions as requirement suggests one model and theoretically it may suggest another model.	Proper selection of SDLC model leads to quality model and lower the rate of project failure.
14	Seema, Sona et al[19]	A view on classes of SDLC model describing traditional methodologies and benefits limitation of Feature Driven Development Agile methodologies	A comparison between SDLC classes Heavyweight and lightweight			
15	Massey, Vishwas. Satao et al [20]	A deep study of various tools, techniques and methodologies is given that extract the most simple and easy method for the comparison of SDLC method	Some methodology analysis for successful comparison of SDLC models.	SDLC provide a layout for software development and a proper employment of SDLC permit the management to normalize the complete development plan of the software.	Comparison between SDLC models is a complex task because these models give a theoretical layout and there is no mathematical proof or any device to do it. It's entirely dependent on developer end and client end constraints.	By this study a best suited SDLC model can be chosen
16	Ragunath, P. K., et al.[21]	To resolve the issues which are ignored by traditional SDLC	To resolve this hypothetical model is introduced	a hypothetical model, explains the interaction of users and developer that is converted into a three dimensional model as user, owner and the developer in a usual SDLC model	The limitation o f addressing these issues under the project management are the discussion of these at surface level not on ground level.	Hypothetical model the novel approach attempts to remove the defects of conventional SDLC "one size fits all".

Section C. Table 1.3

Author	Purpose	Method	characteristics	Limitation	advantage
Deissen boeck, Florian et al [8]	To cure diverse and fuzzy field of software quality model by collecting critique of existing models by analyzing purposes and usage scenarios of these models which are based on literature and author's experiences.	A three level schema (DAP) is proposes for the classification of quality model w.r.t proposed purpose. DAP is the combination of three models (Definition, Assessment and Prediction model) which test a quality model based on model purpose.	General requirements are derived for quality model. On the bases of these requirements we can support existing quality models and can also evaluate for a given context. Guidance for Further development in quality models can also be deduced by these requirements.	DAP is the combination of three models (Definition, Assessment and Prediction model) which test a quality model based on model purpose. So a drawback is a separate quality model for each purpose.	Support existing quality models and can also evaluate for a given context. Guidance for Further development in quality models.
Ing, Côté M., and Georgia dou et al [10].	A quality model that is to be used as the foundation for the definition of quality requirements should help in both the specification of quality requirements and the evaluation of software quality.	Focused on analyzing the semantics of the different models with respect to the stated requirements.	A quality model caters the structure towards a definition of quality. A quality model used as the foundation for the definition of quality requirements should help in both the specification of quality requirements and the evaluation of software quality	The flaw in this approach [that you need a quality process to produce a quality product] is that the emphasis on process usually comes at the expense of constructing, refining, and using adequate product quality models."	usable from the top of the development process to the bottom and from the bottom to the top
Castillo, Isi, Losavio , Matteo, and Bøegh et al [14]	Goal is to incorporate AOSD concepts, the novel standard of ISO/IEC 25030 on quality requirements of software, and typical requirements engineering design.	The UML is used to state the core result of this study.	The REASQ model and ontology is a useful tool at early stages of development	However, a standard and homogenous vision of the AOSD terminology is still missing.	Aspect Oriented Software Development (AOSD) is another framework of representing early specification of non-functional requirements
Florac, William A. et al [15]	To describe and specify two software measures–software problems and defects–used to understand and predict	measureable common attributes are identified and organized to these activities, used with checklists on the basis of observation how it is using and supporting forms to converse the definition and specification for defect measurement and problem	An illustration of checklist and supporting forms can be used to decrease the misreading of measurement results and can be applied to address the information needs of different users.	Sometimes definition checklists cannot record all the information that must be conveyed to avoid ambiguities and misunderstandings.	Many significant software qualities attribute factors, and criteria- reliability, correctness, completeness, efficiency, and usability can be quantified among each other's by using problem and defect measurement
David Chappe ll et al [16]	There's no single best way to view software quality so Some conventional aspects of the software quality are discussed here.	Tools, group code reviews and effective management can also have a big impact on various aspects of software quality	The three main aspects of software quality are Structural, functional and process quality. Each one has its own value and detail.	There are trade-offs as well, where improving quality in one area can lower quality in another Given how important software has become to the world, it's hard to overemphasize the importance of software quality.	Many achievable aspects of software quality are functionality, Reliability, Reusability, portability, Maintainability and Efficiency.

VI. Conclusion

This paper presents the different dimensions of quality software and their dependencies on each other through a comprehensive figure 1.3. Quality software can be achieved by following the given sequence in the diagram. As discussed earlier that the main purpose of the paper is focusing quality of software in the context of SDLC and Requirement specification. So after taking literature review in three separate sections (A, B and C) which are Quality SRS, SDLC and Quality Software respectively its critical evaluation and summaries of all these section in tabular form which have the attributes of method, purpose, characteristics, limitation and advantage we evaluated that Software Quality is a multidimensional area of Software engineering. SDLC and quality SRS are the most common dimensions which affects the Quality of Software. However, some more dimensions like stakeholders also affect the quality of software

by affecting these areas (SDLC and SRS). A multidimensional diagram gives a comprehensive view of all these dimensions and their affection on software quality. Quality Software can be achieved through proper selection of SDLC and it is decided on the bases of requirement nature. Requirement nature is found by the customer's demand then its documentation and its validation which leads to quality SRS. It is also dependent on the development team. After deployment sponsors also affects software quality. So for future work there should be the more generalized ways through which quality of software can be gained. But it is a multidimensional area that cannot be easily achieved. Here the great need is to educate the stakeholders for quality achievement.

REFERENCES

[1] da Silva, Alberto Rodrigues. "Quality of Requirements Specifications: A Preliminary Overview of an Automatic Validation Approach." Proceedings of ACM SAC'2014 Conference. 2014.

[2] da Silva, Alberto Rodrigues. "QUALITY OF REQUIREMENTS SPECIFICATIONS."

[3]de Almeida Ferreira, David, and Alberto Rodrigues da Silva. "RSLingo: An information extraction approach toward formal requirements specifications."Model-Driven Requirements Engineering Workshop (MoDRE), 2012 IEEE. IEEE, 2012.

[4] de Almeida Ferreira, David, and Alberto Rodrigues da Silva. "RSL-PL: A linguistic pattern language for documenting software requirements."Requirements Patterns (RePa), 2013 IEEE Third International Workshop on. IEEE, 2013. 2 versions

[5] de Almeida Ferreira, David, and Alberto Rodrigues da Silva. "RSL-IL: An interlingua for formally documenting requirements." Model-Driven Requirements Engineering (MoDRE), 2013 International Workshop on. IEEE, 2013. 2 versions

[6] de Almeida Ferreira, David, and Alberto Rodrigues da Silva. "Obtaining Formal Requirements Representations with the RSLingo Approach."

[7] de Almeida Ferreira, David, and Alberto Rodrigues da Silva. "A controlled natural language approach for integrating requirements and model-driven engineering." Software Engineering Advances, 2009. ICSEA'09. Fourth International Conference on. IEEE, 2009.

[8] Deissenboeck, Florian, et al. "Software quality models: Purposes, usage scenarios and requirements." Software Quality, 2009. WOSQ'09. ICSE Workshop on. IEEE, 2009.

[9] de Almeida Ferreira, David, and Alberto Rodrigues da Silva. "Wiki-based tool for requirements engineering according to the ProjectIT approach." Software Engineering Advances, 2009. ICSEA'09. Fourth International Conference on. IEEE, 2009.

[10] Ing, Marc-Alexis Côté M., and Elli Georgiadou. "Software quality model requirements for software quality engineering." 14th international conference on the software quality requirement. 2003.

[11] Rajnish, Ranjana, and RAJNISH VYAS. "Writing quality requirements (SRS): an approach to manage requirements volatility." WRITING 1, no. 1 (2010): 28-37.

[12] Japenga, Robert. "How to write a software requirements specification." Micro Tools, Inc (2003).

[13] Goldman, James L., George Abraham, and IlYeol Song. "Generating Software Requirements Specification (IEEE Std. 830 1998) document with Use Cases."

[14] Castillo, Isi, Francisca Losavio, Alfredo Matteo, and Jørgen Bøegh. "Requirements, Aspects and Software Quality: the REASQ model." Journal of Object Technology 9, no. 4 (2010): 69-91.

[15] Florac, William A. Software quality measurement: A framework for counting problems and defects. No. CMU/SEI-92-TR-22. CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST, 1992.

[16] David Chappell THE THREE ASPECTS OF SOFTWARE QUALITY: FUNCTIONAL, STRUCTURAL, AND PROCESS Sponsored by Microsoft Corporation

[17] Seema, Sona. "Analysis and tabular comparison of popular SDLC models."International Journal of Advances in Computing and Information Technology(2012).

[18] Clara, V. Therese. "SDLC AND MODEL SELECTION: A STUDY." International Journal (2013).

[19] Seema, Sona. "Analysis and tabular comparison of popular SDLC models."International Journal of Advances in Computing and Information Technology(2012).

[20] Massey, Vishwas. "Prof. KJ Satao,"Comparing Various SDLC Models and The New Proposed Model On The Basis Of Available Methodology "." International Journal of Advanced Research in Computer Science and Software Engineering2.4 (2012).

[21] Ragunath, P. K., et al. "Evolving a new model (SDLC Model-2010) for software development life cycle (SDLC)." International Journal of Computer Science and Network Security 10.1 (2010): 112-119.

[22] Robertson, S. , Robertson, J.: Mastering the Requirements Process, 2nd edition. Addison-Wesley, 2006.

[23] Ribeiro, A., Silva, A. R.: XIS-Mobile: A DSL for Mobile Applications, Proceedings of SAC 2014 Conference, ACM, 2014.

[24] B. Kitchenham and S. L. Pfleeger. Software quality: The elusive target. IEEE Software, 13(1):12–21, 1996.

[25] S. Wagner and F. Deissenboeck. An integrated approach to quality modelling. In Proc. 5th Workshop on Software Quality (5-WoSQ). IEEE Computer Society Press, 2007.