J. Comp. Sci. Comm., 1(2)12-31, 2015

ISSN 2356-8844
Journal of Computer Sciences and Communication
Www.Textroad.Com

A Survey of Knowledge Management Techniques in SDLC to improve Software Quality

Shumaila Ghafoor, Rizwana Khan, Mamoona Humayun

University Institute of Information Technology, PMAS-Arid Agriculture University Rawalpindi, Pakistan

ABSTRACT

Software Quality is to be assured in all phases of the software development life cycle. Knowledge management is tested in many processes and phases of software development methodologies to improve the quality of software processes. Rather than proposing a complete knowledge management based software development process or model directly, it is important to synthesize the existing work first to clear the research directions. For this purpose, we have presented a simple survey of literature of commonly used existing tools and techniques related to knowledge management, which are applied in different phases of software development life cycle. In results, we have found many tools and various approaches for each phase of SDLC.

KEYWORDS: Software Quality Assurance, Software Development Life Cycle, Knowledge Management.

1- INTRODUCTION

1-1- SOFTWARE QUALITY ASSURANCE

Software Quality refers to "A set of activities whose purpose is to demonstrate that an entity meets all quality requirements" [1]. Software quality assurance refers to assuring the customers as well as other stakeholders (Developer, manager, etc.) that, software is free from errors. Quality assurance is a set of activities which ensures quality in terms of whole software development life cycle like requirement, designing and implementation. [2]

There are many models and various approaches available in literature to ensure the software quality. Standards include CMM [3], CMMI [4], ISO [5] and IEEE [6] etc. ensures the quality of the software process. Whereas, to ensures the quality of software product we have specific and generic methods and standards. In CMMI and other standards, quality assurance techniques can be applied in all phases of the software requirement process which includes phases: stakeholder identification, requirement identification, requirement evaluation, requirements agreement, requirement recording. [7]

1-2- KNOWLEDGE MANAGEMENT

Knowledge management is managing the organizational's knowledge. Knowledge is combined in terms of symbols, data, Information and then comes knowledge. [8] Knowledge management can be applied in every field of engineering and science to gather, store, and share the knowledge to the stakeholders. For accessing knowledge

*Corresponding Author: Shumaila Ghafoor, University Institute of Information Technology, PMAS-Arid Agriculture University Rawalpindi, Pakistan.

organizations use case studies, past research papers and historical database and repositories which includes the same data and information. [9] And for the purpose of the storage purpose, a knowledge repository and experience factory [10] [11] can be used. And similarly, for the sharing knowledge instant messages, calls and other related tools can be used in knowledge management environment. [12]
In all SDLC phases knowledge management is being applied by some of the practitioners. For example, in Requirement engineering phase, knowledge management is applied using SECI (Socialization, Externalization, Combination and Internalization) Knowledge management model. SECI collects Explicit and Tacit knowledge. [13]. Similarly, in Architecture and design phase [14], it is an early set of design and decisions. Risks are identified in early stages of design and development. In the development phase, developers have storage of time and lack of knowledge so they have to use knowledge management. Testing is usually expensive, especially manual testing relies on automated testing tools. KM supports testing teams to capture, store and share testing knowledge [15]. As deployment is post development activity and it involves installation and configuration is complex activities, for this purpose KM technique can be applied, so the organization delivers software product and to remove errors using KM tools and ontologies are being used. [16]

1-3- SURVEY PROCEDURE

To conduct this survey to intent of finding the knowledge management techniques in each phase of the software development life cycle, we have used following libraries which includes: ACM Digital Library, IEEE Explore, Springers, Science Direct, Google Scholar and Others. And the keywords are given below in **Table 1**:

Table 1: Keywords to Search Survey Results

Software Engineering Keywords	**Knowledge Management Keywords**
Software Development Life Cycle	Knowledge Management
Software Development Processes	Knowledge Repositories
Software Quality Assurance	Experience Factory
Software Quality Attributes	Knowledge Management in Software Engineering
	Knowledge Management in Software Testing
	Knowledge Management in Software Requirements Management
	Knowledge Management in Software Development
	Knowledge Management in Software Maintenance

The remainder of this paper is structured as follows: Section 2 summarizes some of the related work. Section 3 is actual synthesized data related to knowledge management techniques in a all phases of the software development life cycle in terms of quality factors of each technique. Section 4 is analysis and discussion based summary phase of work. Section 5 presents the future work and also conclude the work.

2- RELATED WORK

Improving and ensuring the quality of software is not an easy task, it involves a proper sequence of steps, methods and proper guidelines to apply in each phase of the

software development life cycle. There are many protocols and standards are also available for ensuring the quality. Traditional approaches of software development lacks the software quality, and nowadays, new approaches, where agile based software methodologies are very common, focus on the quality of software. Similarly, knowledge management is also considered very helpful in software engineering paradigms. Here is the review of some of the literature in which authors define different strategies and ways by using knowledge management and other related techniques to ensure the overall quality of software development process.

Shanmuganathan et.al,. (2015) [17] presents a survey of knowledge management techniques in SDLC phases. Author(s) have enlisted some of the tools and technologies related to the knowledge management in phases of SDLC, but the scope is limited and no focus on quality attributes in actual. And the terms are enlisted in a general way.

It can also be observed from the study given in [18], in which Infonea tool is discussed, which is used as knowledge management tool and it uses visual search to extract any kind of stored knowledge from the knowledge repository and can improve the overall software quality through the complete knowledge life cycle. It ensures the paperless environment, a web based knowledge repository which makes the system a platform independent and considered as software engineering knowledge portal for any software organization.

It can also be observer from the study [19] related to use of knowledge management in Lean software development methodology in which author(s) proposes a Continuous Quality Improvement (CQI) framework based on knowledge management, which is embedded in software development life cycle to ensure the continuous quality. PDCA (Plan, Do, Check, Act) has some limitations in software enterprises like lots of memory which causes a lot of rework, loss of technology, hiring new employees and many software enterprises is weak and unconscious to share knowledge everywhere. So a CQI knowledge based model should develop.

Alan Dearle et.al, [20] examines the dimensions of the software deployment past , present and future that are influenced. In this paper six case studies are used using different techniques and tools. A set of standard techniques and terminologies are used javabeans, .net linuxtools. These are designed to simplify deployment,development and maintenance of the project and highlighted the deployment issues that are faced by the developers. Similarly, Ron Basu et.al, [21] establishes the main role of quality is time, cost and it is implemented by the concerned people. The methodology used in this paper is a case study, semi structured interviews and survey followed by conceptual model and validated by the SPL model. This paper provides best practices for quality.

Dong et.al, [22] describes the structures which are perceived by knowledge management in terms of software quality. Knowledge sharing is the most important factor that affects the quality the most. Survey methodology is used for the collection of data. Some theoretical aspects of the knowledge sharing are unfolded in this paper. This paper describes how knowledge sharing improves the quality by overcoming the challenges of quality. Similarly, in another study of Andreas et al.,[23] applied knowledge management based techniques are used for quality management. In this paper author compare the process of four companies and finds out that which company's product is the best and the worst. Finds the difference b/w the companies which uses the KM based framework and which not used KM. This paper describes that With KM approaches improve the quality of the product.

3- KNOWLEDGE MANAGEMENT IN SOFTWARE DEVELOPMENT LIFE CYCLE PHASES

Software development life cycle [24] is divided into six phases. These are designed in such a way that an output of one phase is considered as an input of the next phase.

3-1- KM IN SOFTWARE REQUIREMENT ENGINEERING PHASE

Knowledge is highly depended on RE and is divided into classes explicit and tacit, tacit(embodied and yet not embodied) is about need that is shared by stakeholder with the developer [25] explicit knowledge is shared and stored via documents.requirements are the needs of the users and the clients provided by the software system such as features,functions and properties.the aim of the requirement engineering is to analyse extract, manage , combine and evaluate the software system's requirement.eventualy most of the developers have to work with the domain they have a little knowledge and may get the aid of domain experts with tacit knowledge. There are many requirements elicitation techniques JAD[26], RAD[27]and storyboarding[28] ,human interfaces, brain storming,data models, use cases, IS plans and prototyping. About knowledge there is one thing you have to know more than describe it so tacit knowledge is helping to laydown the negative effect on RE[25].

Table 2: List of Knowledge Management based techniques to improve the Software Quality in Software Requirement Phase

Name of Technique	Improve Software Quality Through	References
JAD	Cost, Reliability, efficiency,	[25][26][13]
RAD	Flexibility,correctness reusability, integrity	[25][27][13]
Computer Aided engineering	Interoperability, portability, flexibility, usability, integrity, efficiency.	[17] [30]
Storyboarding	Flexibility, portability, interoperability.	[25][28][17]
REP (requirement elicitation process)	Integrity, efficiency, Flexibility, portability, interoperability.	[29]
Brain storming	Usability, efficiency.	[31]
Requirements workshop	flexibility, usability, reliability.	[73]
Prototyping prototypes	Reliability, Testability, Portability, Flexibility.	[32][33]
Domain analysis	Validation, Maintainability, Efficiency,reliability.	[34][35][36]
Repository grids	Interoperability, portability	[37][38][39]

3-2- KM IN SOTWARE ARCHITECTURE AND DESIGN PHASE

Software architecture is set of early design and decision. Architecture is very early stage of software system any error in design make a system failur. we have critical

systems, security systems and web systems we can't directly change in code that is very costly and difficult. Change in requirements changes in software architecture not change in code directly. We cator software architecture through a tool UML2.0. KM in software architecture is to support. So the software development, evaluation and deployment are determined by software design [40]. Most organizations do not document the software architecture and architecture is made by an architect according to the customer and user needs and requirements.Making a good design is a challenge in engineering. A model [48] is proposed to recover software architecture knowledge from existing documents for the help to the developers. Improving their knowledge through community networking by sharing, transferring explicit and tacit knowledge that can promote sharing knowledge , quality control and management support [48].

Table 3: List of Knowledge Management based techniques to improve the Software Quality in Software Architecture & Design Phase

Name of Technique	Improve Software Quality Through	References
ADDSS(Architecture Design Decision Support System)	Flexibility, reliability	[40] [41]
Archium	Maintainability, traceability	[42] [43]
AREL(Architecture Rationale and Element Linkage)	Reliability	[44]
PAKME (process based architectural knowledge management environment)	Efficiency, correctness	[45]
The knowledge architect	Flexibility, maintainability	[46]
Question Option Criteria (QOC)		[47]
Computer Aided Design (CAD)	Reliability, reduce time cost, Portability.	[17][30]
SysML	Reusability, reduce time, cost, scalability, correctness, usability, virtualization.	[49]
Simulation Techniques	Reusability, reduce time cost, scalability, correctness, usability, virtualization.	[50]
Geometric dimensioning and tolerancing (GD&T).	Interoperability, efficiency	[51]

3-3- KM IN SOFTWARE DEVELOPMENT PHASE

Software development is the stage from where the design is implemented and
A great pressure on developers to make such software which is error free and the

chances of no failures the error or fault in the development stage may cause the failure of a great project. Causes of failure in the development stage are due to lack of knowledge of software engineers, So the availability of the knowledge can be integrated through knowledge management such as sharing, storing and transferring right person at the right time improving the quality of the development [9].
CMM [3] reduce the cost of implementation through eliminating the inconsistencies in software projects. Most of the development companies adopt SPI frameworks for improving the quality of the software product in development stage [3].
When CMM are integrated [4] with KM they are more effective and efficient in the sense of storing, capturing , sharing, transforming and improving for batter documentation and project is handover to the customer without any flaw or delay.

Table 4: List of Knowledge Management based techniques to improve the Software Quality in Software Development Phase

Name of Technique	Improve Software Quality Through	References
CMM / CMMI	Better documentation, cost reduction, reliability	[3][4]
Software process improvement (SPI)	Better documentation, maintenance,	[52] [53]
VSEs	Maintenance, correctness	[54] [17]

3-4- KM IN SOFTWARE TESTING PHASE

Software testing process is to consider that system is correct and working properly and giving the expected output. Software testing process is to consider that system is correct and working properly and giving the expected output. For insuring that system is working according to the user specification or this purpose COP (Community of Practice) is working, in COP software programmers, designers, developers and the user include testing the system [55]. They are all working together to avoid the software from failure of the testing process so there is a need of a tool for this and the tool is called KMS for best practices.
There are many problems that are faced with testing to reuse of testing knowledge,low rate of use and barriers in software testing and poor sharing environment for software testing knowledge [55] [56]. From software specification, we derived test cases, testing skills and reusing program modules are the knowledge involves in the software testing environment. All data kept in the explicit and tacit knowledge in experience or library from where COP can access so that is the opportunity that KM provides through managing knowledge in software testing phase.
In this section we have discussed some of the tools and techniques, based on the knowledge management and can support the testing process. So by using KMS [17] model in managing knowledge in software testing phase and this helps the testers to reduce errors and to improve the quality of the software. The KMS model gives the best quality of the software in testing phase.

Table 5: List of Knowledge Management based techniques to improve the Software Quality in Software Testing Phase

Name of Technique	Improve Software Quality Through	References
KMS	Reduce errors, flexibility correctness, managing data.	[17] [57]
KMM	Cost reduction, reliability	[58]
Fault Injection Model	Efficiency, reliability, flexibility.	[59]
Selective mutation	Efficiency, reliability, scalability.	[60]
Semantic mutation	Efficiency, Portability.	[61]

3-5- KM IN SOFTWARE DEPLOYMENT PHASE

Software engineering is the combination of IT skills, integration of ideas that creates the value of the software development system. Deployment is the activity after the testing phase when the product is ready for handover to the customer it is the phase of configuration, installation and activation takes place.

Table 6: List of Knowledge Management based techniques to improve the Software Quality in Software Deployment Phase

Name of Technique	Improve Software Quality Through	References
Group Storytelling techniques	Maintenance, understandability	[62] [17]
Model driven architecture	Integrity, reusability, portability, code generation.	[63] [64]
Cobra component model	Flexibility, integrity, validity, Reusability, reduce time, cost, scalability, correctness, usability, virtualization.	[65] [66]
IOC		[67] [68]
PXE		[69] [70]
GUID	Interoperability	[69] [71]
Pre-boot execution environment		[69] [70] [72]
URI	Portability,reusability	[69]

3-6- KM IN SOFTWARE MAINTENANCE PHASE

After deployment phase when software delivered to the customer there are still some errors that are undetected , these errors must be removed. This phase is very important in this phase the maintenance engineer [17]

Table 7: List of Knowledge Management based techniques to improve the Software Quality in Software Maintenance Phase

Name of Technique	Improve Software Quality Through	References
Stabilizing requirements and specifications	Correctness, Usability, Efficiency, Testability, Flexibility.	[73] [74]
Develop prototypes and have the requirements reviewed by the client	Flexibility, maintenance, efficiency, reliability	[75] [76]
Reusable source code And interface methods	Reliability, flexibility, Reusability,testability, Integrity	[77]
Reusable test plans and test cases	Reliability, flexibility, Reusability,testability, Integrity	[78] [79]
Utilizing change Control board	Efficiency, Reliability, Automation	[80]

4- SUMMARY AND ANALYSIS ON FINDINGS:

Knowledge management and its tools are using exponentially in software organizations in many ways. Drupal and other CMS tools are under consideration of academia and practitioners. We have found many techniques and tools related to knowledge management for each phase of the software development life cycle and explained separately in previous sections. Now it is important to analyze them and summarize them at one place. In this paper, the main objective is to analyze the knowledge management attributes in such a way, which can be helped to improve the quality of any phase of software development. The summary is given in tabular form below in **Table 8**

Table 8: Summary of Findings the Techniques of Knowledge Management in each phase of SDLC

Requirement phase	Design and Architecture phase	Development phase	Testing phase	Deployment phase	Maintenance phase
JAD	ADDSS (Architecture Design Decision Support System)	CMM	KMS	Group Storytelling techniques	Stabilizing requirements and specifications
+RAD	Archium	Software process improvement (SPI)	KMM	Model driven architecture	Develop prototypes and have the requirements reviewed by the client
Computer Aided engineering	AREL(Architecture Rationale and Element Linkage)	VSEs	Fault Injection Model	Cobra component model	Implementing communication plan

Requirement phase	Design and Architecture phase	Development phase	Testing phase	Deployment phase	Maintenance phase
Storyboarding	PAKME(process based architectural knowledge management environment)	Pair programming	Selective mutation	IOC	Reusable source code And interface methods
REP (requirement elicitation process)	The knowledge architect	**5- -**	Semantic mutation	PXE	Reusable test plans and test cases
Brain storming	Question Option Criteria (QOC)	**6- -**	**7- -**	GUID	Utilizing change Control board
Requirements workshop	Computer Aided Design (CAD)	**8- -**	**9- -**	Pre-boot execution environment	-
Prototyping prototypes	SysML	**10- -**	**11- -**	URI	-
Domain analysis	Simulation TEchniques	**12- -**	**13- -**	**14- -**	**15- -**
Repertory grids	Geometric dimensioning and tolerancing (GD&T).	**16- -**	**17- -**	**18- -**	**19- -**

19-1- ANALYSIS OF RESULTS FOR QUALITY PERSPECTIVE

To achieve the software quality in any methodology or process, practitioners and software engineers set some parameters and standards to achieve software quality. In our findings, we have analyzed each tool and techniques of knowledge management in all the phases of the software development life cycle, in terms of software quality. To gather the data and enlist in terms of software quality, we have presented our findings in tabular form. Below mentioned tables will present the quality achieved by each knowledge management technique. **Table 9** presents the list of techniques to improve the quality of software from requirements phase. **Table 10** presents the techniques related to Design and architecture phase, **Table 11** is for software implementation phase, **Table 12** is for software testing phase, **Table 13** is for software deployment phase and finally **Table 14** presents the results related to the software maintenance phase to improve software quality through knowledge management techniques.

Name of Techniques	Quality attributes									
	Flexibility	Integrity	Testability	Reusability/ usability	Efficiency	Correctness	Relibility	Portability	Interoperability	Maintainabilit
JAD	✓									
RAD	✓	✓		✓		✓				
Computer Aided engineering	✓	✓		✓				✓	✓	✓
Storyboarding	✓							✓	✓	✓
REP(requirement elicitation process)		✓			✓			✓	✓	✓
Brain storming				✓	✓					
Requirements workshop	✓			✓			✓			
Prototyping prototype	✓		✓				✓	✓		
Domain analysis					✓	✓				
Repertory grids										

Table 9: List of Knowledge Management based Techniques on Requirements Phase along with achieving Quality Attribute by each technique

Name of Techniques	Quality attributes									
	Flexibility	Integrity	Testability	Reusabilty/ usability	Efficiency	Correctness	Relibility	Portability	Interopera-bility	Maintaina-bility
ADDSS(Architecture Design Decision Support System)	✓						✓			
Archium									✓	✓
AREL(Architecture Rationale and Element Linkage)							✓			
PAKME(process based architectural knowledge management enivornment)					✓	✓				
Question Opton Criteria (QOC)	✓		✓	✓		✓				
Computer Aided Design (CAD)					✓		✓	✓		
SysML	✓			✓		✓				

Table 10: List of Knowledge Management based Techniques of Design & Architecture Phase along with achieved Quality Attribute by each technique

Name of Techniques	Quality attributes									
	Flexibility	Integrity	Testability	Reusability	Efficiency	Correctness	Relibility	Portability	Interoprability	Maintenability
CMM					✓	✓				✓
Software process improvement (SPI)						✓				✓
VSEs						✓				✓

Table 11: List of Knowledge Management based Techniques of Implementation Phase along with achieved Quality Attribute by each technique

Name of Techniques	Quality attributes									
	Flexibility	**Integrity**	**Testibility**	**Reusability**	**Efficiency**	**Correctness**	**Reliability**	**Portability**	**Interoperability**	**Maintenability**
KMS	✓				✓	✓				✓
KMM						✓	✓			
Fault Injection Model	✓				✓		✓			
Semantic mutation					✓		✓			

Table 12: List of Knowledge Management based Techniques of Software Testing Phase along with achieving Quality Attribute by each technique.

Name of Techniques	Quality attributes									
	Flexibility	**Integrity**	**Testibility**	**Reusability**	**Efficiency**	**Correctness**	**Reliability**	**Portability**	**Interoperability**	**Maintenability**
Group Story telling techniques				✓						✓
Cobra component model										
Model driven architecture		✓		✓				✓		
PXE	✓			✓	✓	✓				✓

Table 13: List of Knowledge Management based Techniques of Software Deployment Phase along with achieved Quality Attribute by each technique

Name of Tech-niques	**Quality attributes**									
	Flexibility	**Integrity**	**Testibility**	**Reusability**	**Efficiency**	**Correctness**	**Reliability**	**Portability**	**Interoperability**	**Maintenability**
Stabilizing requirements and specifications	✓		✓	✓	✓	✓				
Develop prototyping and have the requirements reviewed by the client	✓				✓		✓			✓
Implementing communication plan										
Reusable source code and interface methods	✓	✓	✓	✓			✓			
Reusable test plans and test cases	✓	✓	✓	✓			✓			
Utilizing change control board					✓		✓			✓

Table 14: List of Knowledge Management based Techniques of Software Maintenance Phase along with achieved Quality Attribute by each technique

20- 5- CONCLUSION AND FUTURE WORK

In this survey paper, we focused on knowledge management tools and technologies for each phase of the software development life cycle, which may the quality of software development process or models. We have found many tools, techniques and approaches to improve the each phase of SDLC like Requirements, Design & Architecture, Development, Testing, Deployment and Maintenance.

Knowledge Management has mainly focused on acquisition of knowledge, storage, and sharing. Organizations are using many knowledge based tools like Drupal, Wordpress and other CMS are the best examples to store and share the organization's knowledge. Knowledge management is also integrated with existing processes in some way, intentionally or unintentionally. For example, we can see that in Pair Programming of the XP model, which belongs to the Agile family, is supporting knowledge management activities, because one person is dedicated to the development, design or document and another is to inspect the activities.

This survey paper is the base for our future work in designing knowledge management based complete software architecture to support software development methodologies first, and then designing and proposing a complete software development framework for the practitioners as well as software engineers from academia.

21- ACKNOWLEDGMENT

The authors wish to thank all the faculty members of University Institute of Information Technology – PMAS – UAAR especially Dr. Mamoona Humayun to supervise us in this research article.

22- REFERENCES

[1] El-Haik, Basem S., and Adnan Shaout, 2011. Software design for six sigma: A roadmap for excellence. John Wiley & Sons.

[2] Kitchenham, Barbara A, 1989 . "Software quality assurance." Microprocessors and microsystems 13.6: 373-381.

[3] Weber, Charles V., Bill Curtis, and Mary Beth Chrissis, 1994. The capability maturity model: Guidelines for improving the software process. Vol. 441. Reading, MA: Addison-wesley.

[4] Team, CMMI Product. (2002). "Capability Maturity Model® Integration (CMMI SM), Version 1.1." CMMI for Systems Engineering, Software Engineering, Integrated Product and Process Development, and Supplier Sourcing (CMMI-SE/SW/IPPD/SS, V1. 1)

[5] Ince, Darrel, 1994. ISO 9001 and software quality assurance. McGraw-Hill, Inc.

[6] Jung, Ho-Won, Seung-Gweon Kim, and Chang-Shin Chung 5 (2004). "Measuring software product quality: A survey of ISO/IEC 9126." IEEE software: 88-92.

[7] Pressman, Roger S, 2005. Software engineering: a practitioner's approach. Palgrave Macmillan.

[8] Tiwana, Amrit, 2000. The knowledge management toolkit: practical techniques for building a knowledge management system. Prentice Hall PTR.

[9] Humayoun, Mamoona, and Asad Masood Qazi (2015). "Towards Knowledge Management in RE Practices to Support Software Development." Journal of Software Engineering and Applications 8.08: 407.

[10] Basili, V.R., Caldiera, G., McGarry, F., Pajerski, R., Page, G., Waligora, S (1992). "The software engineering laboratory: an operational software experience facto-

ry". In: Proceedings of the International Conference on Software Engineering (ICSE). ACM, Melbourne.
[11] Basili, V.R., Caldiera, G., Rombach, H.D (1994). "The experience factory". In: Marciniak, J.J. (ed.) Encyclopedia of Software Engineering, vol. 1. Wiley, New York.
[12] Janes, Andrea, and Giancarlo Succi 2014. Lean Software Development in Action. Springer Berlin Heidelberg.
[13] Pilat, Lukas, and Hermann Kaindl 2011. "A knowledge management perspective of requirements engineering." Research Challenges in Information Science (RCIS), Fifth International Conference on. IEEE, 2011.
[14] Gomaa, Hassan, Larry Kerschberg, Vijayan Sugumaran, C. Bosch, I. Tavakoli, and L. O'Hara (1996). "A knowledge-based software engineering environment for reusable software requirements and architectures." Automated Software Engineering 3, no. 3-4: 285-307.
[15] Desai, A., and S. Shah 2011. "Knowledge management and software testing." Proceedings of the International Conference & Workshop on Emerging Trends in Technology. ACM.
[16] de Oliveira, Adriana Cristina, Renata Mendes de Araujo, and Marcos RS Borges 2007. "Telling Stories about System Use: Capturing Collective Tacit Knowledge for System Maintenance." SEKE.
[17] Vasanthapriyan, Shanmuganathan, Jing Tian, and Jianwen Xiang 2015. "A Survey on Knowledge Management in Software Engineering." Software Quality, Reliability and Security-Companion (QRS-C), IEEE International Conference on. IEEE, 2015.
[18] Baltus, Rob 2001. "Integrating Knowledge Management and Quality Management." Software Quality. Springer Berlin Heidelberg, 107-125.
[19] Basill, Victor R (1985). "Quantitative Evaluation of Software Methodology."
[20] Dearle, Alan 2007. "Software deployment, past, present and future." 2007 Future of Software Engineering. IEEE Computer Society,.
[21] Basu, Ron (2014). "Managing quality in projects: An empirical study." International Journal of Project Management 32.1: 178-187.
[22] Kyoon Yoo, Dong (2014). "Substructures of perceived knowledge quality and interactions with knowledge sharing and innovativeness: a sensemaking perspective." Journal of Knowledge Management 18.3: 523-537.
[23] Schneider, Linda, et al., 2013. "Knowledge Creation in Requirements Engineering-A Systematic Literature Review." Wirtschaftsinformatik.
[24] Davis, Alan M., Edward H. Bersoff, and Edward R. Comer (1988). "A strategy for comparing alternative software development life cycle models." Software Engineering, IEEE Transactions on 14.10: 1453-1461.
[25] Olmos Sanchez, Karla, and Jorge Entique Rodas Osollo 2013. "A Strategy to Requirements Engineering Based on Knowledge Management." Computer Science (ENC), Mexican International Conference on. IEEE, 2013.
[26] Duggan, Evan W., and Cherian S. Thachenkary (2004). "Integrating nominal group technique and joint application development for improved systems requirements determination." Information & Management 41.4: 399-411.
[27] Beynon-Davies, Paul, et al., (1999) "Rapid application development (RAD): an empirical review." European Journal of Information Systems 8.3: 211-223.
[28] Jones, Ian. (2008). "Storyboarding: A method for bootstrapping the design of

computer-based educational tasks." Computers & Education 51.3: 1353-1364.
[29] Hickey, Ann M., and Alan M. Davis 2003. "Requirements elicitation and elicitation technique selection: model for two knowledge-intensive software development processes." System Sciences,. Proceedings of the 36th Annual Hawaii International Conference on. IEEE, 2003.
[30] McMahon, Chris, Alistair Lowe, and Steve Culley (2004). "Knowledge management in engineering design: personalization and codification." Journal of Engineering Design 15.4: 307-325.
[31] Paetsch, Frauke, Armin Eberlein, and Frank Maurer 2003. "Requirements engineering and agile software development." null. IEEE, 2003.
[32] Pomberger, Gustav, and Günther Blaschek, 1996. Object-orientation and prototyping in software engineering. Prentice-Hall, Inc..
[33] Boehm, Barry W , 1981. Software engineering economics. Vol. 197. Englewood Cliffs (NJ): Prentice-hall.
[34] Prieto-Díaz, Rubén(1990). "Domain analysis: An introduction." ACM SIGSOFT Software Engineering Notes 15.2: 47-54.
[35] Hjørland, Birger(2002). "Domain analysis in information science: eleven approaches-traditional as well as innovative." Journal of documentation 58.4: 422-462.
[36] Alavi, Maryam, and Dorothy E. Leidner (2001). "Review: Knowledge management and knowledge management systems: Conceptual foundations and research issues." MIS quarterly: 107-136.
[37] Cannataro, Mario, and Domenico Talia (2003). "The knowledge grid." Communications of the ACM 46.1: 89-93.
[38] Moore, Reagan W 2001. "Knowledge-based grids." Mass Storage Systems and Technologies,. MSS'01. Eighteenth IEEE Symposium on. IEEE, 2001.
[39] Zhuge, Hai (2002). "A knowledge grid model and platform for global knowledge sharing." Expert Systems with Applications 22.4: 313-320.
[40] Capilla, Rafael, et al.2008. "ADDSS: Architecture Design Decision Support System Tool." Proceedings of the 2008 23rd IEEE/ACM International Conference on Automated Software Engineering. IEEE Computer Society, 2008.
[41] Capilla, Rafael, Francisco Nava, and Juan C. Duenas. "Modeling and documenting the evolution of architectural design decisions." Proceedings of the Second Workshop on SHAring and Reusing architectural Knowledge Architecture, Rationale, and Design Intent. IEEE Computer Society, 2007.
[42] Shahin, Mojtaba, Peng Liang, and Mohammad Reza Khayyambashi. "Improving understandability of architecture design through visualization of architectural design decision." Proceedings of the 2010 ICSE Workshop on Sharing and Reusing Architectural Knowledge. ACM, 2010.
[43] Jansen, Anton, et al. "Tool support for architectural decisions." Software Architecture, 2007. WICSA'07. The Working IEEE/IFIP Conference on. Ieee, 2007.
[44] Tang, Antony, Yan Jin, and Jun Han. "A rationale-based architecture model for design traceability and reasoning." Journal of Systems and Software 80.6 (2007): 918-934.
[45] Babar, Muhammad Ali, and Ian Gorton. "A tool for managing software architecture knowledge." Proceedings of the Second Workshop on SHAring and Reusing architectural Knowledge Architecture, Rationale, and Design Intent. IEEE Computer Society, 2007.

[46] Liang, Peng, Anton Jansen, and Paris Avgeriou. "Knowledge Architect: A tool suite for managing software architecture knowledge." University of Groningen, Tech. Rep (2009).
[47] MacLean, Allan, et al. "Questions, options, and criteria: Elements of design space analysis." Human–computer interaction 6.3-4 (1991): 201-250.
[48] Lucia, Andrea De, et al. "Recovering traceability links in software artifact management systems using information retrieval methods." ACM Transactions on Software Engineering and Methodology (TOSEM) 16.4 (2007): 13.
[49] Friedenthal, Sanford, Alan Moore, and Rick Steiner. A practical guide to SysML: the systems modeling language. Morgan Kaufmann, 2014.
[50] Naylor, Thomas H. "Computer simulation techniques." (1966).
[51] Krulikowski, Alex. Geometric Dimensioning and Tolerancing. Cengage Learning, 1998.
[52] Zahran, Sami. Software process improvement: practical guidelines for business susccess. Addison-Wesley Longman Ltd., 1998.
[53] Carlsson, Sven A., and Mikael Schönström. "Software Process Improvement Through Knowledge Management." Proceedings of The Fourth European Conference on Organizational Knowledge, Learning, and Capabilities. 2003.
[54] Laporte, Claude Y., and Alain April. "Applying software engineering standards in small settings: Recent historical perspectives and initial achievements." Proceedings of the First International Research Workshop for Process Improvement in Small Settings. Software Engineering Institute, Carnegie Mellon University, CMU/SEI-2006-Special Report. 2006.
[55] Schlager, Mark, Judith Fusco, and Patricia Schank. "Evolution of an online education community of practice." Building virtual communities: Learning and change in cyberspace (2002): 129-158.
[56] Duguid, Paul. "“The art of knowing”: social and tacit dimensions of knowledge and the limits of the community of practice." The information society 21.2 (2005): 109-118.
[57] Alavi, Maryam, and Dorothy E. Leidner. "Review: Knowledge management and knowledge management systems: Conceptual foundations and research issues." MIS quarterly (2001): 107-136.
[58] Kochikar, V. P. "The knowledge management maturity model: a staged framework for leveraging knowledge." Proceedings of KM World (2000).
[59] da Silva, Jonny Carlos, et al. "A knowledge-based system approach for sensor fault modeling, detection and mitigation." Expert Systems with Applications 39.12 (2012): 10977-10989.
[60] Offutt, A. Jefferson, Gregg Rothermel, and Christian Zapf. "An experimental evaluation of selective mutation." Proceedings of the 15th international conference on Software Engineering. IEEE Computer Society Press, 1993.
[61] Clark, John A., Haitao Dan, and Robert M. Hierons. "Semantic mutation testing." Software Testing, Verification, and Validation Workshops (ICSTW), 2010 Third International Conference on. IEEE, 2010.
[62] Claypool, Kajal, and Mark Claypool. "Teaching software engineering through game design." ACM SIGCSE Bulletin. Vol. 37. No. 3. ACM, 2005.
[63] Stahl, Thomas, Markus Voelter, and Krzysztof Czarnecki. Model-driven software development: technology, engineering, management. John Wiley & Sons, 2006.
[64] Schmidt, Douglas C. "Guest editor's introduction: Model-driven engineering."

Computer 39.2 (2006): 0025-31.
[65] Wang, Nanbor, et al. "Towards a reflective middleware framework for QoS-enabled CORBA component model applications." IEEE Distributed Systems Online 2.5 (2001).
[66] Wang, Nanbor, Douglas C. Schmidt, and Carlos O'Ryan. "Overview of the corba component model." Component-Based Software Engineering. Addison-Wesley Longman Publishing Co., Inc., 2001.
[67] Boehm, Barry, and Li Guo Huang. "Value-based software engineering: A case study." Computer 36.3 (2003): 33-41.
[68] Boehm, Barry. "A view of 20th and 21st century software engineering." Proceedings of the 28th international conference on Software engineering. ACM, 2006.
[69] Dearle, Alan. "Software deployment, past, present and future." 2007 Future of Software Engineering. IEEE Computer Society, 2007.
[70] Maddux, Alvin, Glenn Baker, and Brian Donohoue. "Method and apparatus for advanced software deployment." U.S. Patent Application No. 09/929,832.
[71] Mishra, Debi P., et al. "Method and system for on-demand installation of software implementations." U.S. Patent No. 6,523,166. 18 Feb. 2003.
[72] Owens, Gary L., and David Labuda. "Automated software installation and operating environment configuration for a computer system based on classification rules." U.S. Patent No. 5,555,416. 10 Sep. 1996.
[73] Oasmaa, Anja, et al. "Norms and standards for pyrolysis liquids. End-user requirements and specifications." Energy & fuels 19.5 (2005): 2155-2163.
[74] Eades, James L., and Ralph E. Grim. "A quick test to determine lime requirements for lime stabilization." Highway research record 139 (1966).
[75] Mannio, Markus, and Uolevi Nikula. "Requirements elicitation using a combination of prototypes and scenarios." WER. 2001.
[76] Eliassen, Frank, et al. "Next generation middleware: Requirements, architecture, and prototypes." Distributed Computing Systems, 1999. Proceedings. 7th IEEE Workshop on Future Trends of. IEEE, 1999.
[77] Lemos, Otávio Augusto Lazzarini, et al. "CodeGenie: using test-cases to search and reuse source code." Proceedings of the twenty-second IEEE/ACM international conference on Automated software engineering. ACM, 2007.
[78] SHANG, Dong-juan, et al. "A Study of Test Cases and Reuse in Software Testing [J]." Computer Technology and Development 1 (2006): 021.
[79] Linzhang, Wang, et al. "Generating test cases from UML activity diagram based on gray-box method." Software Engineering Conference, 2004. 11th Asia-Pacific. IEEE, 2004.
[80] Conger, Sue A. The new software engineering. Course Technology Press, 1993.