Indian Journal of Science and Technology, *Vol 9(20), DOI: 10.17485/ijst/2016/v9i20/86869, May 2016*
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Software Documentation Management Issues and Practices: a Survey

C. J. Satish* and M. Anand

School of Computer Science and Engineering, VIT University, Near Katpadi Road, Vellore – 632014, Tamilnadu, India; satish.cj@vit.ac.in, manand@vit.ac.in

Abstract

Background/Objectives: Software Documentation plays a significant role in any project. This paper is mainly focused on summarizing the various problems with documentation and the practices proposed to address them**. Findings**: The analysis establishes the fact that documentation needs are specific for each project. The documentation management issues relate to either neglecting or overdoing documentation. **Application/Improvements:** A study of all the issues and practices for documentation management revealed that documentation processes should be tailor made for every project according to the various factors that determine the documentation needs for a project. Implementation of documentation practices should be project specific rather than organization specific.

Keywords: Artifacts, Maintenance, Software Documentation, Software Engineering

1. Introduction

Software documentation is created even before the development of the software. Software artifacts like requirements, design, and test case documentation play a significant role in capturing the in-depth knowledge about a software project. Documentation should not be only used for system description but also detail the creation, updates made to the system[1]. Such artifacts play an important role in knowledge transition and thereby helps reduce the time taken for program comprehension during impact analysis[2].

Documentation also plays a crucial role when the software hand over happens from one organization to another organization. Only documentation can provide the knowledge required for maintenance engineers from several perspectives. Software documentation also increases the functional correctness of changes for complex tasks[3]. Majority of the time is spent on understanding the impact of changes on the system during maintenance[4]. In the absence of software documentation, engineers should spend more time gathering the required information about the system before making the necessary changes[5].

Research has proved that good quality documents improve the performance of engineers working on the project[6,7]. It has been found that even documents that are not up to date may still be useful. The value of documentation is judged based on its capability to impart knowledge to its users even when it's not up to date[8].

Though documentation is proven to be having great impact on software projects, still a lot of issues surround the development and maintenance of documents. These issues are being continuously reported in various research papers from the time software engineering came in to place. There are many issues like documentation not up-to-date, documents not being traceable reported repeatedly in many papers. Such issues have been addressed using different techniques like introduction of tools[9-12] and processes[13-15].

Though several strategies have been proposed to address the issues with documentation, the problems still persist and an effective solution seems elusive. It is the need

**Author for correspondence*

Table 1. Software Documentation Issues and Practices

Author	Year	Problem Reported	Proposed Solution
Timothy C. Lethbridge[16]	2003	• Documentation is out of date • Documentation is not complete • It is a challenging task to search and find useful contents in software documentation	• Should bring in simple and powerful tools for documentation • Should enforce discipline among software engineers so that they will update documents frequently
Ahmad Salman Khan[2]	2012	• Inadequate documentation and difficulties in tracking changes are considered to be core Hand Over Problems	• Documentation should not contain unnecessary information such that searching becomes more difficult • Common System Repository should be used to track all changes to the system
Golara Garousi[13]	2013	• Quality Factors that affect documentation quality are documentation up-to-date-ness, accuracy and completeness	• Opportunistic and situational documentation • Document should be developed in such a way that it caters to the needs of different users and one size fits all documentation strategy should not be followed • Retirement of least accessed documents should be done.
Sergio Cozzetti B. de Souza[17]	2005	As the documentation is not up to date, it loses its trustworthiness. Henceforth maintenance engineers understand the system using only source code. • Documentation is prone to the following issues • Documentation is missing or it is not of good standards • A lot of documents get created without any objective and documentation is not frequently updated • It is difficult to access documentation that are of different formats and saved on different systems. • Programmers are not motivated to document • It is difficult to establish documentation standards	• Documentation that are frequently used like requirements specification and data models should be updated • Comments in source code also acts as an important guideline for maintenance engineers
Sumita Das[14]	2007	• The location of documentation is a critical issue. • The solution to a problem may exist in a document that the engineers may never find	• Documentation structure should enable easy navigation to information (structural properties like acronyms ,glossaries, FAQs ,diagrams, examples, table of contents indices and appendix should be used) • Readability, simplicity and completeness, accuracy and brevity should be maintained for all documents
Andrew Forward[8]	2002	• Documentation is rarely updated by engineers who are working even in high quality projects • Changes to documents are not traceable to source code • Documents are frequently used even when they are outdate (requirements specification is an example)	• Extraction of system documentation directly from source code using tools • Tools to support traceability among documents and source code

Erik Arisholm[6]	2006	• UML Diagrams were very large and difficult to update • Modification of UML diagrams consume lot of effort	• Current tools need further modifications to support changes to models and consistency checking
Laura Moreno[18]	2014	• Documentation is often missing and outdated.	• Continuously updating comments on source code • Automatic generation of natural language summaries from source code using tools
Ahmad Salman Khan[19]	2012	• Documentation is not consistent with the system	• A repository for system documentation should be established • The services provided by the repository should be defined • The repository should be subjected to software configuration management • Documentation standards should be defined.
Ana M. Fernández-Sáez[15]	2015	• Documentation is not updated • Documentation does not exist for some cases • Documents are very long • For old systems documentation may not explain the current situation • Lack of synchronization between models and source code	• Documentation usage should be based on team size. Larger teams use UML documents more frequently. • Documentation maintenance should be given additional weightage based on team size
Christoph Johann Stettina[20]	2014	• Documentation is not up to date • Systems have documentation that are not properly written using well defined standards	• Instead of the development team alone being involved in the documentation process all teams which use the documentation should be made part of it • Project initiator should be pro-active and be part of the process
William L. Miller[21]	2013	• During software hand over around 500 documents were handed over to support and understand the software • The handed over documents were not current, accurate or of the correct version • Documents were delivered in different formats	• A process should be in place to ensure that every document is accurate and current • All documents should be in an editable electronic version
Mira kajko-Mattsson[1]	2005	• Documentation that is out of date is not useful to a maintenance engineer who is fixing a complex task. • Documentation that is of poor quality is a major reason for defects during the maintenance phase. • Documents are not updated along with the source code changes • Traceability of changes is not achieved	• Organizations should provide detailed guidelines on documentation • Engineers should be motivated enough for writing quality documentation
Khaled Jaber[22]	2013	• Specification and design documents are not updated to reflect the source code changes • Company documentation and coding standards often change but the old code is not made up to date	• Implementation of tools that can establish traceability between the changes to the source code and documentation • Documentation can be done before the making changes to the code
Steven P. Reiss[10]	2005	• The lack of tool support makes software artifacts evolve independently and become inconsistent over time	• Implementation of a tool (CLIME) responsible for detection of inconsistencies between documentation and source code.

Sofia Sherman[23]	2012	• Software Engineers do not update documents frequently • Architecture specification documents are rarely updated	• Architecture Specification documentation(ASD) process proposed for maintenance of ASD Documentation
Coen J. Burki[24]	2014	• Documentation is not complete and people who have the knowledge about the system retire without adequate knowledge transition • The absence of original requirements documentation nullifies the option of rebuilding the system.	• Supporting redocumentation by automatically generating technical and functional documentation using tools (Omnext is the tool discussed in this paper)
Christoph Treude[20]	2015	• It's difficult to search for information in technical documentation • Most of the times engineers did not know where to look for documentation	• Automatic extraction of development tasks from documents to enhance navigation support for software engineers
Maria Alaranta[11]	2012	• Poor documentation is a key source of ambiguity and uncertainties	• Implementation of a Transactive Memory System (combining individuals or groups transactive memories and communication)
Lionel C. Briand[25]	2003	• Documentation is obsolete and incomplete. • Documentation is expensive to maintain and engineers find it difficult to update documents due to time pressures in the software industry	• The use of Object Control Language during object oriented analysis is proposed.
Golara Garousi[26]	2015	• Documentation is not up to date, it is inconsistent and incomplete • Documentation activity involves lot of cost and it is tough to maintain documentation in many projects	• Code comments should be given • Design documents should be of high quality and it should be available on time • Documentation is highly useful to intermediate developers than senior developers. Therefore documentation should cater to the needs of the intermediate developers. • Readability of Documents should be increased • Avoid Extra Large Documents
Vandana Singh[27]	2013	• Software documentation is not consistent or complete. • There is little documentation describing what is needed from the system	• Effective information retrieval techniques are needed • Tools will helps users to a) minimize the jargon b) delete out of date information
Wei Ding[28]	2014	• Absence of architectural information is problematic for changing software. • Software Architecture documentation is time consuming to create and it is often not up to date.	• Need to understand the needs of Open source software developers with respect to Software Architecture Documentation • There is a need to investigate the quality of architecture documentation in open source projects
R. Plosch[29]	2014	• It is difficult to mention quality requirements precisely using natural language. It is not possible to measure the quality of a requirement given in natural language.	• Stucturedness understandability, accuracy and clarity should be given importance with respect to documentation

Todd Waits[30]	2014	• Maintenance of System documentation using binary files from multiple contributors is a tedious task • Collaborative work using long email chains or using network file shares will lead to synchronization problems and loss of information. The information will not be trustworthy. It will be difficult to track the latest trustworthy information.	• Integration of Documents with the build process • Traceability can be achieved by binding a specific application build to a specific documentation build
Vikas S. Chomal[9]	2015	• The important reasons behind defects in the development and maintenance phase are the lack of complete and consistent documentation that are up to date.	• A tool was implemented for evaluating the quality of Software Documentations. The tool used a template for evaluation of the documents.
Gerard o Canfora[31]	2011	• Software comprehension isa time consuming task because of lack of up to date documentation. The documents are not update due to time or resource constraints during the development phase.	• The high level documentation that has design views like class, sequence and state chart should be updated using source code. • Artifacts should be updated along with source code. • Artifacts can be reverse engineered from source code using tools
Andrew Forward[12]	2002	• The current perception of documentation isthat it is outdated, irrelevant and incomplete.	• Document aura metrics to predict the usefulness, referential decay and authority of a document. • Implementation of Document Aura Calculator tool that works on DAC(Document Aura Calculator) metrics
Christoph Johann Stettina[32]	2013	• Too much documentation is available for systems that are not updated and trustworthy • It is a tedious task to update software design documentation iteratively.	• The maintenance engineers should be included in the early stages of development.
Eirik Tryggeseth[3]	1997	• Software maintenance problems have given high priority to the problems of lacking or inadequate documentation	• Documentation is an asset in software maintenance and should be precisely taken care of. • Experiment was carried out to prove the effectiveness of documentation on maintenance activities. • Good documentation practices lead to reduction of impact analysis time

of the hour to understand why software documentation is facing so many issues and why solutions proposed in various research articles is not solving this problem.

As part of this paper, we have summarized the issues reported with Software Documentation and the various practices proposed to address them. We have primarily focused only on documentation issues reported in these papers.

We have summarized the reported issues and proposed practices in Table 1. Finally we have summarized our analysis of the review and proposed effective documentation guidelines under Section 2.

2. Conclusion

The analyses of the research papers reveal that documentation issues seem persistent throughout the life time of the software. The most common issues reported are

- Documentation being out of date
- Documentation not being traceable to the changes made in code
- Documentation lacking standards and a lot of effort needed with respect to updating of documents
- Software Engineers showing less enthusiasm with documentation

- Lack of Tools for documentation maintenance and change tracking

Though documentation is plagued with many such issues it still acts as a performance booster in projects. Even documents that are out of date are considered to be useful according to research. Research also proves that documentation increases the performance of engineers with respect to impact analysis and understanding the system[15]. Based on the analysis we have listed the following recommendations on Software Documentation.
The significance of software documentation is determined as follows

- The size of the project
- The experience levels of the persons who will be using the documents
- The geographic distribution of teams
- Standards enforced on the documentation content
- Processes that govern the evolution of documents
- Tools that enhance navigation, searching and traceability of documents

Though software engineering lays certain standards on documentation governance the same is not applicable for all projects. Projects that are managed using smaller teams with experienced engineers who are not geographically distributed can work well with minimum documentation. Documentation can be boon for projects of larger scale with geographically distributed teams.

Documentation change management tools or traceability enhancement tools will also be applicable only for projects on a large scale with budget to accommodate the training and purchase of such tools. Enforcing higher documentation standards and stringent reviews on documentation content and structure can solve much of the problems with documentation even without the tool support.

Rather than following a standard process for documentation for all projects, the documentation process should be tailor made for every project based on the points mentioned above. Documentation can effectively increase productivity in projects where the documentation process is defined as per the needs of the project.

Application of common documentation practices for all projects can sometimes lead to draining of resources and demotivation for engineers who are passionate about technical work. Our analysis of the research papers lead to a conclusion that documentation processes should be a project specific choice and common documentation practices across the organization may complicate and lead to further issues rather than providing quality solutions.

3. References

1. Kajko-Mattsson Mira. A survey of documentation practice within corrective maintenance. Empirical Software Engineering 10.1. 2005; p. 31–55.
2. Lethbridge Timothy C, Janice Singer and Andrew Forward. How software engineers use documentation: The state of the practice. Software iEEE 20.6. 2003; p. 35–39.
3. Tryggeseth Eirik. Report from an experiment: Impact of documentation on maintenance. Empirical Software Engineering 2.2. 1997; p. 201–07.
4. Sousa Castro Maria Joao and Moreira Helena Mendes. A survey on the software maintenance process. 1998 IEEE, Proceedings International Conference on Software Maintenance. 1998.
5. Leotta Maurizio et al. A pilot experiment to quantify the effect of documentation accuracy on maintenance tasks. 2013 IEEE International Conference on Software Maintenance. IEEE, 2013.
6. Arisholm Erik et al. The impact of UML documentation on software maintenance: An experimental evaluation. IEEE Transactions on Software Engineering 32.6. 2006; p. 365–81.
7. Fernandez-Saez Ana M et al. On the use of UML documentation in software maintenance: Results from a survey in industry. Model Driven Engineering Languages and Systems (MODELS). 2015 ACM/IEEE 18th International Conference on. IEEE, 2015.
8. Forward Andrew and Timothy C Lethbridge. The relevance of software documentation, tools and technologies: a survey. Proceedings of the 2002 ACM symposium on Document engineering. ACM, 2002.
9. Chomal Vikas S and Jatinderkumar R Saini. Software Template for Evaluating and Scoring Software Project Documentations. International Journal of Computer Applications 116.1. 2015.
10. Reiss Steven P. Incremental maintenance of software artifacts. IEEE Transactions on Software Engineering 32.9. 2006; p. 682–97.
11. Alaranta Maria and Stefanie Betz. Knowledge Problems in Corrective Software Maintenance-A Case Study. IEEE, 2012 45th Hawaii International Conference on System Science (HICSS). 2012.
12. Forward Andrew. University of Ottawa: Software Documentation–Building and Maintaining Artefacts of Communication. Dissertation. 2002.

13. Garousi Golara et al. Evaluating usage and quality of technical software documentation: an empirical study. Proceedings of the 17th International Conference on Evaluation and Assessment in Software Engineering. ACM, 2013.
14. Das Sumita, Wayne G Lutters and Carolyn B Seaman. Understanding documentation value in software maintenance. Proceedings of the 2007 Symposium on Computer human interaction for the management of information technology. ACM, 2007.
15. Fernandez-Saez Ana M et al. On the use of UML documentation in software maintenance: Results from a survey in industry. 2015 ACM/IEEE 18th International Conference on Model Driven Engineering Languages and Systems (MODELS). IEEE, 2015.
16. Lethbridge Timothy C, Janice Singer and Andrew Forward. How software engineers use documentation: The state of the practice. Software, iEEE 20.6. 2003; p. 35–39.
17. de Souza Sergio Cozzetti B, Nicolas Anquetil and Kathia M de Oliveira. A study of the documentation essential to software maintenance. Proceedings of the 23rd annual international conference on Design of communication: documenting & designing for pervasive information. ACM, 2005.
18. Moreno Laura. Summarization of complex software artifacts. Companion Proceedings of the 36th International Conference on Software Engineering. ACM, 2014.
19. Khan Ahmad Salman and Mira Kajko Mattsson. Management of documentation and maintainability in the context of software handover. 2012 8th International Conference on Computing Technology and Information Management (ICCM). IEEE, 2012; 1.
20. Treude Christoph, Martin P Robillard and Barthelemy Dagenais. Extracting development tasks to navigate software documentation. IEEE Transactions on Software Engineering 41.6. 2015; p. 565–81.
21. Miller William L, Lawrence B Compton and Bruce L Woodmansee. Assuming Software Maintenance of a Large, Embedded Legacy System from the Original Developer. 2013 IEEE International Conference on Software Maintenance. IEEE, 2013.
22. Jaber Khaled, Bonita Sharif and Chang Liu. A study on the effect of traceability links in software maintenance. Access, IEEE 1. 2013; p. 726–41.
23. Sherman Sofia and Irit Hadar. Identifying the need for a sustainable architecture maintenance process. 2012 IEEE, 5th International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE). 2012.
24. Coen J Burki, Harald H Vogt. How to save on software maintenance costs An Omnext white paper on software quality. 2014.
25. Briand Lionel C. Software documentation: how much is enough? 2003 IEEE, Proceedings Seventh European Conference on Software Maintenance and Reengineering. 2003.
26. Garousi Golara et al. Usage and usefulness of technical software documentation: An industrial case study. Information and Software Technology 57. 2015; p. 664–82.
27. Singh Vandana and Lila Holt. Online Technical Support: How It Works and Why It Fails? 2013 IEEE, 46th Hawaii International Conference on System Sciences (HICSS). 2013.
28. Ding Wei et al. How do open source communities document software architecture: An exploratory survey. 2014 IEEE, 19th International Conference on Engineering of Complex Computer Systems (ICECCS). 2014.
29. Plosch Reinhold, Andreas Dautovic and Matthias Saft. The Value of Software Documentation Quality. 2014 IEEE, 14th International Conference on Quality Software (QSIC). 2014.
30. Waits Todd and Joseph Yankel. Continuous system and user documentation integration. 2014 IEEE International Professional Communication Conference (IPCC). IEEE, 2014.
31. Canfora Gerardo, Massimiliano Di Penta and Luigi Cerulo. Achievements and challenges in software reverse engineering. Communications of the ACM 54.4. 2011: p. 142–51.
32. Stettina Christoph Johann and Egbert Kroon. Is there an agile handover? An empirical study of documentation and project handover practices across agile software teams. 2013 International Conference on Engineering, Technology and Innovation (ICE) & IEEE International Technology Management Conference. IEEE, 2013.