Agile Requirements Engineering's Challenges

Ezeldin Sherif, Waleed Helmy and Galal Hassan
Faculty of Computers & AI, Cairo University, Cairo, Egypt

Keywords: Scrum, Requirements Engineering Challenges, Agile.

Abstract: Agile methodology focuses on delivering working software as soon as possible, whilst having the customer involved from start to end. There are some challenges in requirements engineering in Agile. For example, non-functional requirements are being ignored and not treated as first-class artifacts during the development lifecycle. This causes a lot of problems such as customer dissatisfaction and a lot of rework; which affects time and cost. This paper explains the different challenges of requirements engineering in Agile for the past decade.

1 INTRODUCTION

Agile Software Development (ASD) aims at maximizing productivity, effectiveness, and speeding delivery; through minimizing documentation and removal of unnecessary practices within time constraints. ASD is a hotspot of the software development field where Agility is flexible; it is a state of dynamic and self-improvement. Its main purpose is to provide usable software that adds value to clients by reducing the amount of generated intermediate software products, models, documentation, and other comparable artifacts. ASD has numerous advantages. However, there are still some challenges. The paper answers the research question which is what are the current challenges in Agile requirements engineering? The challenges of Agile requirements engineering are discussed in this paper. It is organized as the following: Section two presents the research methodology, Section three gives an overview of ASD, Section four presents requirements engineering challenges in Agile for the past decade and Section five provides the conclusion.

2 RESEARCH METHODOLOGY

This paper is a research to study existing challenges in requirements engineering in Agile software development. The research methodology runs into four phases. The first phase screened the existing related articles through significant search engines such as Google Scholar, ACM, IEEE Xplore Digital Library, Springer, Science Direct, and Wiley Online Library. The screening phase used a keyword-based search on the terms “challenges in Agile”, “challenges in requirements engineering of Agile”, “challenges in Scrum”, “requirement engineering in Agile”, “requirement engineering and Agile”, and “requirement engineering or Agile” and “requirements in Agile”. The second phase focused on filtering the collected articles for choosing the major key works that would be reviewed and analyzed. All relevant articles were categorized according to their origin, whether scientific or practice-oriented and according to the publication site, whether in journals or conference proceedings. The third phase was building a descriptive-analytical study of the filtered works to reveal the challenges of requirements engineering in Agile. Finally, the fourth phase has drawn the conclusion from the study.

3 AGILE SOFTWARE DEVELOPMENT BACKGROUND

The term Agile implies agility, adaptability, and flexibility (Jarzębowicz & Weichbroth, 2021). One large software project is split into several executable subprojects in Agile software development. First, the development team focuses on the most significant features that users have requested. For all selected functionalities, the team employs an iterative incremental development technique, with each iteration result-

Sherif, E., Helmy, W. and Hassan, G.
Agile Requirements Engineering's Challenges.
DOI: 10.5220/0011309900003266
In *Proceedings of the 17th International Conference on Software Technologies (ICSOFT 2022)*, pages 221-225
ISBN: 978-989-758-588-3; ISSN: 2184-2833

Table 1: Challenges in Agile Requirements Engineering.

No	Challenge	(Schön et al., 2017)	(Behutiye et al., 2017)	(Alsaqaf et al., 2017)	(Gaikwad & Joeg, 2017)	(Elghariani & Kama, 2016)	(Alam et al., 2017)	(Käpyaho & Kauppinen, 2015)	(Inayat, Moraes, et al., 2015)	(Inayat, Salim, et al., 2015)	(Sunner & Bajaj, 2016)	(Saleh et al., 2018)	(Batra & Bhatnagar, 2019)	(Telesko, 2018)	(Ramesh et al., 2010)	(Saleh et al., 2021)
1	Neglect of NFR	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	-	Yes	Yes	Yes
2	Minimal documentation	Yes	Yes	-	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
3	Inadequate/inappropriate architecture	Yes	-	Yes	-	Yes	-	-	-	Yes	Yes	Yes	Yes	Yes	Yes	Yes
4	Prioritization of NFR	Yes	-	-	-	-	-	-	Yes	-	-	Yes	-	-	Yes	Yes
5	NFR infeasibility	-	-	Yes	-	-	Yes	-	-	-	-	-	-	-	-	-
6	Teams interaction	-	-	Yes	-	-	-	-	-	-	-	-	Yes	-	-	-
7	Inadequate NFR verification	-	-	Yes	Yes	-	-	-	-	-	-	-	Yes	-	Yes	-
8	NFRs identification	-	-	Yes	-	-	Yes	-	-	-	-	Yes	-	-	-	-
9	Customer unavailability	-	-	-	Yes	Yes	-	-	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
10	Project budget and time estimation	-	-	-	-	Yes	Yes	-	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
11	Change of requirements	-	-	-	-	Yes	Yes	-	-	Yes	Yes	Yes	-	-	-	Yes
12	Problems related to prototyping	-	-	-	-	-	-	Yes	-	-	-	-	-	-	-	-
13	Not understanding the big picture	-	-	-	-	-	-	Yes	-	-	-	-	-	-	-	-
14	Lack of requirements traceability	-	-	-	-	-	-	-	Yes	-	-	-	-	-	-	-
15	Contractual limitations	-	-	-	-	-	-	-	Yes	Yes	Yes	-	-	-	-	Yes
16	Customer's knowledge and agreement	-	-	-	-	-	-	-	-	Yes	Yes	Yes	-	-	-	-

ing in a functioning system. The development team concentrates on responding rapidly to changing requirements. When a change in requirements is requested, the development team quickly adjusts the plan. Agile software development has become increasingly popular in the past decade due to its benefits: early and frequent delivery, transparency, flexibility, lower costs, better productivity, better software quality, allow for changes, focus on business values, better customer engagement, better customer satisfaction and lower process overhead (Jarzębowicz & Weichbroth, 2021)(Käpyaho & Kauppinen, 2015)(Matharu et al., 2015)(Achrak & Chkouri, 2020)(Alhazmi & Huang, 2020). One of the most essential qualities of ASD is the ability to handle unstable requirements throughout the development lifecycle and, second, it can deliver products in shorter timeframes (Huo et al., 2004). Nowadays, there are several different Agile methods available; for example Scrum, Crystal, Agile Modeling, and XP. The Agile methods are concentrated on several various aspects of the software development life cycle. Although they are different in detail, they share the same core values and principles. Agile methods deal with unpredictably changing requirements by adhering to a set of principles that include simple planning, short iterations, earlier releases, and frequent user feedback. Acceptance testing, pair programming, continuous integration, and refactoring are just a few of the approaches that have quality assurance potential (Huo et al., 2004). In 2001, seventeen software developers met to discuss a lightweight and effective development method (Beck et al., n.d.)(Aldave et al., 2019) (Husaria & Guerreiro, 2020). They published a document called "Manifesto for Agile Software Development". Using a collective knowledge of software development and seeing a need to change from heavyweight process models such as the Waterfall, they wrote the Agile manifesto which included twelve principles and four values. There are two main objectives of those principles. The first objective is to increase awareness about Agile methods. The second objective is to assist project teams in determining whether or not they are adopting an Agile method. This manifesto, together with all of its principles and values, represents the concept behind Agile methods, and should ideally be included in all of the Agile methods' practices (Fernandes & Almeida, 2010). Those principles are not enough to accommodate the importance of the non-functional requirements (NFRs) (Ana Silva et al., 2017).

4 REQUIREMENTS ENGINEERING CHALLENGES IN AGILE

As demonstrated in table 1, there are numerous challenges in Agile requirements engineering. It shows 15 different research papers along with 16 challenges. The challenge labeled with "Yes" means that this challenge is mentioned in the given paper. The challenge labeled with "-"means that this challenge is not mentioned in the given paper. As noticed in table 1, there are 16 different challenges in requirements engineering in Agile. We can categorize those challenges into three categories. The first category contains challenges that are most repeated across different research papers. In this category, two challenges fit this criterion, which is: "minimal documentation" and "neglect of non-functional requirements". These two challenges are stated in 14 different research papers. This gives an indicator that there is a major pain in non-functional requirements and documentation in Agile which requires further research. The second category contains challenges that are repeated in almost half of the research papers. In this category, two challenges fit this criterion, which is "inadequate/inappropriate architecture", "customer unavailability" and "project budget and time estimation". The third category contains challenges that are repeated in a few research papers. This category, contains the rest of the challenges which contain 11 challenges. The following is the list of challenges that are stated in the above table with further discussion of each one:

1. Neglect of Non-functional requirements:

   Ignoring non-functional requirements (NFRs) is one of the main challenges of ASD in the (Behutiye et al., 2020). Problems in neglecting non-functional requirements include insufficient documentation of non-functional requirements, techniques inadequate to deal with NFRs in Agile development, and excessive focus on functionality. According to Cao and Ramesh's empirical investigation (Behutiye et al., 2017) (Alsaqaf et al., 2017), found that non-functional requirements are given lesser importance in the early stages of ASD; as customer focuses on core functions. Furthermore, NFRs are frequently overlooked in the testing of ASD for a variety of causes, including a lack of understanding, inexperience, and time and expense constraints. (Cruzes et al., 2019). Microsoft recommends capturing both non-functional and functional requirements like (Jarzębowicz & Weichbroth, 2021). Additionally, Oracle also believes that stakeholders must have a common knowledge of application requirements. for software development to be effective (Jarzębowicz & Weichbroth, 2021). Non-functional needs may be disregarded during the initial stages of the development of software, which can lead to:

   (a) Software of poor quality (Behutiye et al., 2017),

   (b) Larger maintenance time and cost (Behutiye et al., 2017) (Andreia Silva et al., 2016),

   (c) Software failure (Younas et al., 2020).

2. Minimal documentation:

   Existing requirements engineering practices in Agile software development fail shortly as regards the documentation of non-functional requirements. For example, non-functional needs are difficult to specify and document with user stories (Behutiye et al., 2017). When non-functional needs are not written, requirements traceability becomes harder, the risk of forgetting non-functional requirements rises, and negative effects such as low user acceptability may occur (Behutiye et al., 2017) (Memmel et al., 2007) (Hussain et al., 2009).

3. Inadequate/inappropriate architecture:

   One of the most important purposes of software architecture is to create a software design that addresses non-functional requirements (Kassab, 2017). The achievement of non-functional needs is linked to software architecture. Changes made to non-functional requirements at any point in the development phase may result in costly changes in the software architecture; (Alsaqaf et al., 2017). Most non-functional requirements should be known before development, because they can affect the choices of architecture, programming language, database, or operating system (Eberlein, 2003). Therefore, changing the architecture is extremely difficult after the development has started (Saito et al., 2012).

4. Prioritization of NFRs:

   The requirements are prioritized based on functional requirements rather than non-functional requirements. As a result, it is based on a single dimension, because the fundamental criterion for prioritizing those functional requirements is business value. As a result, non-functional requirements aren't actually taken into account while prioritizing requirements.

5. Non-functional requirements infeasibility:

   Finding out that the non-functional requirements are infeasible at a later point of the development stage may cause refactoring of the architecture and re-implement the delivered features (Alsaqaf et al., 2017).

6. Team interactions:

   Different teams might be working on the same code and non-functional requirements could be implemented across the entire system. (Alsaqaf et al., 2017).

7. Inadequate NFRs verification:

   Finding and developing acceptance tests for non-functional needs can be tough due to their difficulty in the modeling (Alsaqaf et al., 2017).

8. Non-functional identification:

   To gather all non-functional requirements, all stakeholders should be identified to represent all different viewpoints of the system. Overlooking any stakeholder may lead to missing requirements and therefore may cause major problems (Alsaqaf et al., 2017).

9. Problems related to prototyping:

   Customers may have unrealistic expectations as a result of issues such as early UI prototypes. (Käpyaho & Kauppinen, 2015).

10. Not understanding the big picture:

    Because there is little pre-planning, not having a clear image of the requirements at the start of the development might result in a lot of rework and irritation. (Käpyaho & Kauppinen, 2015).

11. Lack of requirements traceability:
    The absence of requirement traceability is a concern with ASD. The proposed is provide methods to enable adequate requirements traceability (Inayat, Moraes, et al., 2015).
12. Contractual limitations:
    Fixed-price contracts don't allow changes during the software's development cycle (Inayat, Salim, et al., 2015).
13. Customer's knowledge and agreement:
    The challenge is that customer has incomplete knowledge of the domain and therefore incomplete consensus among customer groups (Sunner & Bajaj, 2016).

5 CONCLUSION AND FUTURE WORKS

There are many different challenges in requirements engineering in Agile software development. Each challenge comes with its impact. In conclusion, neglecting non-functional requirements and minimal documentation are two main challenges in Agile. This requires further research to minimize the impact of those two challenges without breaking the values and principles of the Agile Manifesto.

REFERENCES

Achrak, E. M., & Chkouri, M. Y. (2020). Integrate and Apply the Recommendation System of Agile Methods. In Advances in Intelligent Systems and Computing: Vol. 1105 AISC. Springer International Publishing. https://doi.org/10.1007/978-3-030-36674-2_30

Alam, S., Nazir, S., Asim, S., & Amr, D. (2017). Impact and Challenges of Requirement Engineering in Agile Methodologies: A Systematic Review. International Journal of Advanced Computer Science and Applications, 8(4), 411–420. https://doi.org/10.14569/ijacsa.2017.080455

Aldave, A., Vara, J. M., Granada, D., & Marcos, E. (2019). Leveraging creativity in requirements elicitation within agile software development: A systematic literature review. Journal of Systems and Software, 157. https://doi.org/10.1016/j.jss.2019.110396

Alhazmi, A., & Huang, S. (2020). Survey on Differences of Requirements Engineering for Traditional and Agile Development Processes. Conference Proceedings - IEEE SOUTHEASTCON, 2020-Janua. https://doi.org/10.1109/SoutheastCon-44009.2020.9397492.

Alsaqaf, W., Daneva, M., & Wieringa, R. (2017). Agile Quality Requirements Engineering Challenges: First Results from a Case Study. International Symposium on Empirical Software Engineering and Measurement, 2017-Novem, 454–459. https://doi.org/10.1109/ESEM.2017.61

Batra, M., & Bhatnagar, A. (2019). A Research Study on Critical Challenges in Agile Requirements Engineering. June, 1214–1219.

Beck, K., Beedle, M., Bennekum, A. van, Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R. C., Mellor, S., Schwaber, K., Sutherland, J., & Thomas, D. (n.d.). Manifesto for Agile Software Development. Retrieved March 27, 2020, from http://agilemanifesto.org/

Behutiye, W., Karhapää, P., Costal, D., Oivo, M., & Franch, X. (2017). Non-functional requirements documentation in agile software development: Challenges and solution proposal. ArXiv, 515–522. https://doi.org/10.1007/978-3-319-69926-4

Behutiye, W., Karhapää, P., López, L., Burgués, X., Martínez-Fernández, S., Vollmer, A. M., Rodríguez, P., Franch, X., & Oivo, M. (2020). Management of quality requirements in agile and rapid software development: A systematic mapping study. Information and Software Technology, 123, 106225. https://doi.org/10.1016/j.infsof.2019.106225

Cruzes, D. S., Felderer, M., Oyetoyan, T. D., Gander, M., & Pekaric, I. (2019). How is security testing done in agile teams? A cross-case analysis of four software teams. Lecture Notes in Informatics (LNI), Proceedings - Series of the Gesellschaft Fur Informatik (GI), P-292, 133–134. https://doi.org/10.18420/se2019-40

Eberlein, A. (2003). Requirements Engineering and Agile Software Development, University of Calgary. WET ICE 2003. Proceedings. Twelfth IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, 2003., 1–6.

Elghariani, K., & Kama, N. (2016). Review on Agile requirements engineering challenges. 2016 3rd International Conference on Computer and Information Sciences, ICCOINS 2016 - Proceedings, 507–512. https://doi.org/10.1109/ICCOINS.2016.7783267

Fernandes, J. M., & Almeida, M. (2010). Classification and comparison of agile methods. Proceedings - 7th International Conference on the Quality of Information and Communications Technology, QUATIC 2010, 391–396. https://doi.org/10.1109/QUATIC.2010.71

Gaikwad, V., & Joeg, P. (2017). A case study in requirements engineering in context of agile. International Journal of Applied Engineering Research, 12(8), 1697–1702.

Huo, M., Verner, J., Zhu, L., & Babar, M. A. (2004). Software quality and agile methods. Proceedings - International Computer Software and Applications Conference, 1, 520–525. https://doi.org/10.1109/cmpsac.2004.1342889

Husaria, A., & Guerreiro, S. (2020). Requirement engineering and the role of design thinking. ICEIS 2020 - Proceedings of the 22nd International Conference on

Enterprise Information Systems, 2(Iceis), 353–359. https://doi.org/10.5220/0009489303530359

Hussain, Z., Slany, W., & Holzinger, A. (2009). Current state of agile user-centered design: A survey. Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 5889 LNCS, 416–427. https://doi.org/10.1007/978-3-642-10308-7_30

Inayat, I., Moraes, L., Daneva, M., & Salim, S. S. (2015). A reflection on agile requirements engineering: Solutions brought and challenges posed. ACM International Conference Proceeding Series, 25-29-May-. https://doi.org/10.1145/2764979.2764985

Inayat, I., Salim, S. S., Marczak, S., Daneva, M., & Shamshirband, S. (2015). A systematic literature review on agile requirements engineering practices and challenges. Computers in Human Behavior, 51, 915–929. https://doi.org/10.1016/j.chb.2014.10.046

Jarzębowicz, A., & Weichbroth, P. (2021). A Systematic Literature Review on Implementing Non-functional Requirements in Agile Software Development: Issues and Facilitating Practices. In Lecture Notes in Business Information Processing (Vol. 408). Springer International Publishing. https://doi.org/10.1007/978-3-030-67084-9_6

Käpyaho, M., & Kauppinen, M. (2015). Agile requirements engineering with prototyping: A case study. 2015 IEEE 23rd International Requirements Engineering Conference, RE 2015 - Proceedings, 334–343. https://doi.org/10.1109/RE.2015.7320450

Kassab, M. (2017). A contemporary view on software quality requirements in agile and software architecture practices. Proceedings - 2017 IEEE 25th International Requirements Engineering Conference Workshops, REW 2017, 260–267. https://doi.org/10.1109/REW.2017.60

Matharu, G. S., Mishra, A., Singh, H., & Upadhyay, P. (2015). Empirical Study of Agile Software Development Methodologies. ACM SIGSOFT Software Engineering Notes, 40(1), 1–6. https://doi.org/10.1145/2693208.2693233

Memmel, T., Reiterer, H., & Holzinger, A. (2007). Agile methods and visual specification in software development: A chance to ensure universal access. Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 4554 LNCS(PART 1), 453–462. https://doi.org/10.1007/978-3-540-73279-2_51

Ramesh, B., Cao, L., & Baskerville, R. (2010). Agile requirements engineering practices and challenges: an empirical study. Information Systems Journal, 20(5), 449–480. https://doi.org/10.1111/j.1365-2575.2007.00259.x

Saito, Y., Monden, A., & Matsumoto, K. (2012). Evaluation of non functional requirements in a request for proposal (RFP). Proceedings of the 2012 Joint Conf. of the 22nd Int. Workshop on Software Measurement and the 2012 7th Int. Conf. on Software Process and Product Measurement, IWSM-MENSURA 2012, 106–111. https://doi.org/10.1109/IWSM-MENSURA.2012.23

Saleh, M., Baharom, F., Farvin, S., Mohamed, P., & Ahmad, M. (2018). A Systematic Literature Review of Challenges and Critical Success Factors in Agile Requirement Engineering. 9th Knowledge Management International Conference (KMICe), July, 248–254. http://www.kmice.cms.net.my/248

Saleh, M., Baharom, F., & Mohamed, S. F. P. (2021). Critical success factors and challenges in agile requirements engineering. Turkish Journal of Computer and Mathematics Education, 12(3), 1670–1682. https://doi.org/10.17762/turcomat.v12i3.989

Schön, E. M., Winter, D., Escalona, M. J., & Thomaschewski, J. (2017). Key challenges in agile requirements engineering. Lecture Notes in Business Information Processing, 283, 37–51. https://doi.org/10.1007/978-3-319-57633-6_3

Silva, Ana, Araújo, T., Nunes, J., Perkusich, M., Dilorenzo, E., Almeida, H., & Perkusich, A. (2017). A systematic review on the use of Definition of Done on agile software development projects. ACM International Conference Proceeding Series, Part F1286, 364–373. https://doi.org/10.1145/3084226.3084262

Silva, Andreia, Pinheiro, P., Albuquerque, A., & Barroso, J. (2016). A process for creating the elicitation guide of non-functional requirements. Advances in Intelligent Systems and Computing, 465, 293–302. https://doi.org/10.1007/978-3-319-33622-0_27

Sunner, D., & Bajaj, H. (2016). Classification of Functional and Non-functional Requirements in Agile by Cluster Neuro-Genetic Approach. International Journal of Software Engineering and Its Applications, 10(10), 129–138. https://doi.org/10.14257/ijseia.2016.10.10.13

Telesko, R. (2018). Road to agile requirements engineering: Lessons learned from a web app project. Studies in Systems, Decision and Control, 141, 65–78. https://doi.org/10.1007/978-3-319-74322-6_5

Younas, M., Jawawi, D. N. A., Ghani, I., & Shah, M. A. (2020). Extraction of non-functional requirement using semantic similarity distance. Neural Computing and Applications, 32(11), 7383–7397. https://doi.org/10.1007/s00521-019-04226-5