



OPEN ACCESS

EDITED BY

Huai Liu,
Swinburne University of Technology, Australia

REVIEWED BY

Sandra Sanchez-Gordon,
Escuela Politécnica Nacional, Ecuador
Feliënbe Hermans,
VU Amsterdam, Netherlands

*CORRESPONDENCE

Artem Kruglov
✉ a.kruglov@innopolis.ru

RECEIVED 10 March 2023

ACCEPTED 29 August 2023

PUBLISHED 25 September 2023


CITATION

Bakare A, Ciancarini P, Farina M, Kruglov A,
Okonicha O, Smirnova M and Succi G (2023)
Learning from West African storytellers.
Front. Comput. Sci. 5:1183602.
doi: 10.3389/fcomp.2023.1183602

COPYRIGHT

© 2023 Bakare, Ciancarini, Farina, Kruglov,
Okonicha, Smirnova and Succi. This is an
open-access article distributed under the terms
of the [Creative Commons Attribution License
\(CC BY\)](https://creativecommons.org/licenses/by/4.0/). The use, distribution or reproduction
in other forums is permitted, provided the
original author(s) and the copyright owner(s)
are credited and that the original publication in
this journal is cited, in accordance with
accepted academic practice. No use,
distribution or reproduction is permitted which
does not comply with these terms.

Learning from West African storytellers

Ayomide Bakare¹, Paolo Ciancarini², Mirko Farina ^{3,4},
Artem Kruglov^{1*}, Ozioma Okonicha¹, Marina Smirnova¹ and
Giancarlo Succi²

¹Lab of Industrializing Software Production, Faculty of Computer Science and Engineering, Innopolis University, Innopolis, Russia, ²Department of Computer Science and Engineering, University of Bologna, Bologna, Italy, ³Philosophy and Computer Science, Faculty of Humanities and Social Sciences, Head of Human Machine Interaction Lab (HMI Lab), Institute for Digital Economy & Artificial Systems (IDEAS), Xiamen University and Lomonosov Moscow State University, Xiamen, Fujian Province, China, ⁴Visiting Research Fellow University of Religions and Denominations, Qom, Iran

Several research works propose non-textual alternatives to textual documentation and similar forms of representing information in software development. This is because of the problems that stem from writing these documents, which range from incomprehensible requirements to ambiguous user stories. The various proposals of researchers often contain some trace of oral or visual communication. In this paper, we study the implications of eliminating textual communication and substituting unnecessary writing by extracting the values of West African oral storytellers. Traditional West African communities did not make use of writing for thousands of years and yet their legends, customs, beliefs, and knowledge were effectively transmitted across several generations. How did they manage to accomplish this? What can we learn from their storytellers? How can these lessons be applied to software products? These are all questions that this paper attempts to answer. Perhaps if we fully understand how they operated, then we can target our written communication to the activities where it is needed instead of spreading writing across plenty of tasks as it is currently. To achieve this, we performed an analysis of the two domains: West African oral storytelling and software development and found similarities, then selected some key elements from oral storytelling and explained how they can have relevance in software development. The theme directly encompasses diversity and inclusion by bringing into software engineering a perspective of a region where its literacy research is scarcely being explored. The study found that traditional oral storytelling can provide insights into effective communication and audience engagement, and identified four ways in which software development can be compared to oral storytelling. The study also found that certain elements of storytelling, such as audience relationship, story structure, parables and proverbs, and community relaxation and support, can be applied to writing tasks in software development.

KEYWORDS

oral storytelling, storytelling in software development, empirical methods, software as an art, agile software development

1. Introduction

We cannot list the countless scientific and technical papers emphasizing the importance of a properly written software documentation. There are several reasons why it is a good idea to write a software specification document: it is a contract among the stakeholders, it is a guideline for the developers, a reference for the testers, etc. Still, after almost a century of software development, most software knowledge is still passed via oral transmission. Here, we are not referring to the overall vision of software systems, which are well-described with the “official” technical documentation; rather we concentrate our effort on the team meetings conducted for review, reflection, discussion, and feedback.

During the early days of software development, software analysis and design were mostly perceived to be a purely technical process. Nowadays, researchers and developers realize that the process of building some software presents a strong affinity to conversations (Chilana et al., 2016) and/or writing stories (Lloyd, 2000). Taking this interpretation of software development into consideration, alongside the initiative arose to dive into the history of West African storytellers and attempt to learn a thing or two from them. Why West African storytellers? This region stands out because its population had a late start in writing (Kelly, 2019). This makes us wonder about how they were able to preserve their stories and pass them down from generation to generation without writing for such a long time. It is the same thing we are trying to achieve in software development throughout the life cycle of the software project, as well as through the turnover of developers and employees who oversee the project, and we want to be able to preserve and transfer the essence of the information using writing only when necessary. This led to the research question “What elements can be adopted from West African oral storytelling to substitute the unnecessary writing part of software development?”.

This paper explores the oral-centric method of telling stories used by West Africans *griots* and maps out a comparison with current software development methods. Furthermore, it highlights the elements of the storytellers that look promising and finally proposes a new approach on how the oral storytelling methods of West Africans can improve the methods software developers use. This incorporates diversity and inclusion into software engineering field by drawing inspiration from and exploring a region where research of its literature has not been explored in depth.

The paper has the following structure: Section 2 provides the background of our work and discusses a comparison of storytelling and software development; Section 3 presents the motivations of our research and describes our approach; Section 4 outlines our results; Section 5 discusses the limitations of this work; finally, Section 6 draws our conclusions and describes our plans to continue this research.

2. Background

To start off with, what is oral storytelling? Oral storytelling is simply the art of telling a story verbally to an audience. It could be in the form of singing, chanting, dancing, or even poems. For centuries, the history, beliefs, and folklore of West African communities have been preserved through the tradition of singing and storytelling. A key feature of oral storytelling is that there is an audience and that the interaction between the storyteller and the audience is like a two-way street. The listeners are giving feedback just as much as the storyteller is narrating the tale.

On the other hand, what is software development? According to IBM (2023), “Software development refers to a set of computer science activities dedicated to the process of creating, designing, deploying, and supporting software”. The field of software engineering as a creative activity is not completely novel. Some works have already begun addressing and revealing the ways in which the activities of writing and programming, while different, can be noticed to have the same fundamental intersection.

One of such research (Hermans and Aldewereld, 2017) tries to draw a comparison between writing and programming by comparing their goals and challenges. While the result observed some differences between writing and programming, it also listed seven steps of the two processes which coincide with each other (Burke, 2012). On the other hand investigates to understand if there is any intersection between programming and writing through the storytelling motif using a formal educational setting. The result of this research shows the effectiveness of aligning writing with computer science, as there is an interconnected composition process.

There are also researchers like Bussell and Taylor (2006), Ciancarini et al. (2020), and Lloyd (2000) who aimed to change the viewpoint of people to see how these two processes can learn from each other. Ciancarini et al. (2020) reaches the major finding that the advance of software engineering can be inspired by the literature. In Bussell and Taylor (2006), the authors shed light on a side of software development that holds similar properties to some collaborative writing projects for example making movies. Lloyd (2000) focused on general engineering design, but pointed out that engineering design itself can be seen from three perspective of experience. Individual experience, social experience, and organizational experience, and that storytelling is an important way to conduct and package social experiences.

All of these authors have shed light and sparked the conversation on the compatibility of programming and narratives: this paper narrows it down to West African literature. The region of West Africa was chosen over other regions like North East Africa (precisely Egypt) because West African written literature started a lot more recent than other regions (Kelly, 2019).

In West Africa, while oral storytelling today is not as prominent as it once was since people began learning to read and write, it is still a major source of information sharing. This storytelling method is a shared communal event where people gather, “listening, and participating in accounts and stories of past deeds, beliefs, wisdom, counsel, morals, taboos, and myths” (Utley, 2008). In a similar way, software development is also a communal process where developers congregate together, discussing, designing, planning, programming, testing, and participating in accounts of a process. While this seems to be the most recognizable similarity between the two, Table 1 shows more.

To elaborate, in West African oral storytelling, the narration usually occurs after everyone has returned from their farms and had their meal, then everyone gathers and participates in the storytelling ceremony (Utley, 2008). In other cases, there are special storytellers called “griots” who are summoned and have to entertain the king and his court. In each of these storytelling ceremonies, there is always an audience present, either the village or the king’s court. Likewise, in software development, in particular the agile methodology, the team members gather at the end of each iteration to reflect or give feedback. This is important as it improves the team’s productivity and allows them to grow and improve. In these meetings, there is always a group of people involved; the team (Lamoreux, 2005).

Next, the oral nature of West African storytelling generally serves the purpose of entertaining. This is the reason why singing, dancing, musical instruments, and chanting are also part of the storytelling process (Utley, 2008). Similarly, a recent paradigm,

TABLE 1 West African storytelling vs. software development.

No.	WA oral storytelling	Software development
1	Storytelling involves a community or an audience	Software development involves a team
2	It aims to entertain its audience	It aims to satisfy its customers
3	The story depends on the morals of the tribe	The product depends on the structure of the organization (Conway's law)
4	A story is made up of verses, stanzas	A program is made up of lines of code

such as Lean software development, focuses on customer-centered products; that is, products that ensure that the customer receives value from the product developed (Ebert et al., 2012). Entertainment can be one way to enhance the customer experience and create a positive emotional connection with the product.

Furthermore, in storytelling, the kinds of stories are often distinguishable from tribe to tribe because each tribe has its own elements from their values that make it unique (Utley, 2008). Similarly, according to Conway's law (Herbsleb and Grinter, 1999), the products and systems developed by organizations are a reflection of the organization structure and communication.

The last similarity is pretty straightforward. Just like a book containing a written story can be divided into chapters, an oral story can be divided into verses or stanzas, and a program can be divided into lines of code or modules. The smallest construct of the organization metrics in the two processes are words.

In the remaining sections, first we discuss the research problem and its motivation, the approach we take is elaborated on, and, finally, we explain the results and draw conclusions.

3. Research problem and motivation

In software development, in fact, a great deal of writing is involved. This ranges from writing software architecture descriptions to user and security requirements to writing clean code and comprehensible comments. In all of these aspects of writing, there is still a problem in ensuring that the architecture description is clear or that the requirements are unambiguous. Chaabane et al. (2017) is directed at the problem of describing SoS software architecture, Osman and Zaharin (2018) recaps the problem of requirements in software engineering.

Regarding user stories, Gilson et al. (2020) proposes a way that can visualize user stories, as understanding, analyzing, and maintaining the backlog of written user stories is quite difficult. Of numerous papers that emphasize the importance of clean code, Ljung (2021) is a recent paper that performs a literature review to understand developers' perception of clean code, as there can be a negative effect on businesses if developers write difficult-to-maintain code. Finally, for accurate and reliable documentation, Aghajani et al. (2020) says that it is "an invaluable asset in any software project" but developers still have a problem with documenting properly and hence documents have useless and inadequate information.

All of these are to stress the importance and problems of writing in software development. A lot of research has been going on in these different facets of software and researchers are investigating ways to produce more useful write-ups. Any

ambiguities could potentially lead to the wrong interpretations of the requirements and consequently affect the project:

- Using vague language. Using words like "sometimes" or "usually" can leave the reader unsure of what to expect.
- Overusing passive voice can make sentences sound awkward and unclear. For example, "The data was analyzed" is less clear than "We analyzed the data."
- Writing excessively long sentences. Some software documents can be wordy, with long sentences and complex phrasing. For example, "In order to ensure proper functionality, it is imperative that the user take the necessary steps to configure the system correctly" instead of "Configure the system properly to ensure it works correctly."
- Using technical jargon without explanation. For example, "The system implements a stateful protocol using a RESTful API" instead of "The system uses a specific type of communication to exchange data." While jargon can be helpful in communicating precise technical concepts, it can be confusing for non-technical readers.
- Using acronyms. Acronyms are frequently used in software documents, such as "API" for "Application Programming Interface" or "SQL" for "Structured Query Language." However, it's important to remember that not everyone may be familiar with these acronyms.

From our point of view, most of the writing problems related to software development mentioned above are often resolved by including some form of verbal communication or non-textual technique, such as displaying visually (Gilson et al., 2020).

Therefore, the motivation of this paper is to question the extent to which writing is absolutely necessary in software development as a whole. Currently, writing is involved in some form in almost every step of the software development life cycle. Perhaps if the software development community relied on oral communication like in the West African communities, we can pass down information, structures, and values just like they were able to do for years upon years. Of course, it is definitely not feasible to completely eliminate writing and communicate only orally, but we can start up the conversation to understand in which areas exactly writing is crucial and in which areas we can substitute it for other activities.

To understand and answer the research question that arises from the problem, we conducted a study that examines the oral storytelling-based lifestyle of ancient West Africans. With the results of this examination, we plan to transfer and apply them to software development activities. The study is guided by the following goals:

- G₁:** To analyze the degree to which writing is meaningful and effective in the activities that involve it in software development.
- G₂:** To discover the most significant elements of oral storytelling adopted by West Africans.
- G₃:** To determine how these elements can be molded and adapted to fit software development activities.

4. Results

This section is divided into three subsections to support each the goals specified in the previous Section 3.

4.1. G₁: Analysis of the importance of writing in software development activities

We have already established that software development cannot operate without any writing at all. [Stavely \(2011\)](#) emphasizes that the skill of writing cannot be underestimated. His book touches on the areas where writing is used in software development, such as document design, code documentation, review write-up, requirements, and proposal of the project. However, we formulate the hypothesis that “some writing activities done by software programmers are unnecessary.” Taking the definition of software waste from [Womack and Jones \(1996\)](#) as “any activity that consumes resources but creates no value,” we can cross-examine the processes that involve writing and evaluate if they actually contribute to the value of the customer. Using five of [Sedano et al.'s \(2017\)](#) waste classification approach, some writing activities fall into various categories of waste as seen in [Table 2](#).

Of course, it is important to note that these examples are of the current problematic ways of writing that researchers are trying to solve. Nevertheless, the process of figuring out the best techniques or models and training the developers also poses some waste. On that account, this classification supports the hypothesis that not all writing activities done by software developers are necessary.

4.2. G₂: Discovering the most significant elements of oral storytelling adopted by West Africans

Just like one can distinguish a good software developer from a bad one, there are also good and bad storytellers. Good West African oral storytellers did not just rely on their words to relay the story to their listeners. They were able to introduce facial and physical gestures, character imitations or even songs and dances in their narrative. The best storytellers used the following elements: drama, careful timing, appropriate voices, and sustenance of a dynamic relationship with the audience ([Utley, 2008](#)). In addition, experienced storytellers incorporated repetition, rhythm, imagery, proverbs, riddles, and similes ([Utley, 2008](#)) into their stories. All of these elements play a role in the communal storytelling experience:

- Drama fosters intrigue,
- Repetition increases understanding and retentive memory,
- Gesture captures attention,
- Song and dance ensures audience interaction, and
- Maintaining a relationship tightens the bond.

4.3. G₃: Adapting elements of storytelling to software development activities

After highlighting the principles of the oral storytelling and realizing that some of the writing activities in software development can be regarded as waste, we move on to adapt the storytelling components to replace some writing tasks. To create a focus group of writing activities to apply these elements to, we use the same writing activities from [Table 2](#).

4.3.1. Mismanaging the backlog

[Barbosa et al. \(2016\)](#) stated the problem that in large projects stories could be duplicated as a result of lack of communication between team members. As we can see from West African storytelling, the members of the community are just as vital to the storytelling process as the storyteller themselves. When the storyteller gives a prompt, everyone is to **respond** and this is done a couple of times so that even if someone in the audience is participating for the first time, by the third group chant they already join in.

4.3.2. Rework

[Koznov \(2012\)](#) points out the issue of poor technical documentation and how there are approaches to make the teaching of proper documentation easier. Regarding one of such approach he states that “changing design is faster than rewriting the text.” Agreeing with it, we can use the West African storytelling element for this purpose. Storytellers always have an order of events before every storytelling session; this demonstrates how important the **structure** of the oral narrative is ([Tuwe, 2016](#)). From introducing all the characters at the beginning to concluding with a moral lesson, the storyteller does not stray from the order and narrates in a vibrant manner accompanied by gestures, songs, and dances. In order to reduce re-writing any form of documentation, a structure has to be maintained.

4.3.3. Unnecessarily complex solutions

While there are currently architecture description languages, specially for helping to deal with and describe complex software systems, we can also take a look at the use of proverbs and parables in West African storytelling. The “goal of using proverbs and parables is to achieve harmony and wisdom...” ([Tuwe, 2016](#)). The proverbs and parables represent the collection of knowledge and emotions of the people. When there is harmony and wisdom, developers can be on the same page and examine each feature and question its **necessity** before including it in the design. The equivalent of proverbs and parables in software would be when team members share anecdotes of the lessons from previous projects and experiences they overcome together.

TABLE 2 Some writing activities as waste.

Waste	Description	Observed causes
Mismanaging the backlog	The cost of duplicating work, expediting lower value user features, or delaying necessary bug fixes	Lots of inadequate user stories in the backlog makes it difficult to manage
Rework	The cost of altering delivered work that should have been done correctly but was not	Re-writing ambiguous stories
Unnecessarily complex solutions	The cost of creating a more complicated solution than necessary, a missed opportunity to simplify features, user interface, or code	Detailed and precise software architecture description
Extraneous cognitive load	The costs of unneeded expenditure of mental energy	Iterative ambiguous requirements
Psychological distress	The costs of burdening the team with unhelpful stress	Additional long documents

TABLE 3 Summary of results.

No.	Goal	Outcome
1	Analyze the degree to which writing is meaningful and effective in the activities that involve it in software development	Five waste categories that writing can be filed under
2	Discover the most significant elements of oral storytelling adopted by West Africans	Interaction, drama, timing, relationship, repetition, rhythm, imagery, proverbs, riddles and similes, gesture, song, and dance
3	Determine how these elements can be molded and adapted to fit software development activities	Mismanaging the backlog can be solved by using the listeners interaction element
		Rework can be solved by using the structure element
		Unnecessarily complex solutions can be solved by using proverbs and parables
		Extraneous cognitive load and Psychological distress can be solved by appropriate and fixed rest times

4.3.4. Extraneous cognitive load and psychological distress

These two are a group because the same element can be adopted. Developers spend their mental and physical energy on understanding any documentation, but there is a threshold where it obstructs productivity. In some communities, working was taboo when it is time for storytelling; **rest** and support are essential. No person should be left behind, if someone is stranded, the rest of the community come to their aid and go on together to listen to stories. Developers can replace writing when stressed with mandatory breaks and the team members should be supportive.

5. Limitations

The idea that software is the result of a conversation (Chilana et al., 2016) is especially related to agile models of software development and the Conway law. The idea that storytelling, in particular, gives substance to the conversation is especially influenced by Extreme Programming practices, which include requirements represented by user stories (Beck, 2000) and architectures described by “metaphores” (Herbsleb et al., 2003).

While the focus of this study is to examine West African storytellers, a couple of references used were about general African story telling. This is because:

- Non-gray literature on research into West African storytelling is narrow,

- Storytelling is one common characteristic that spans across the continent of Africa.

Additionally, this approach is new and requires a deeper investigation into the substantial benefits for software developers.

6. Conclusion

In conclusion, this paper started by pointing out that a lot of research has gone into recognizing the similarity between software development and narratives and how they can possibly gain something from each other. Then it moved on to express that regardless of knowing the value of well-written and comprehensible documents, most of the documentation written by software developers are still obsolete. Eventually, most of the core information shared among developers is passed on through gatherings, meetings, and giving feedback to each other in person. This gave rise to the aim of the paper to acknowledge the places where writing is necessary and places where it is not, in order to reduce or get rid of the redundant writing.

Following this purpose, we first reviewed the ways and importance of traditional oral storytelling to a West African community. These storytelling communities did not use written narratives for quite some time and yet for many generations they successfully passed down information. Then we conducted an analysis of ways in which software development is comparable to West African oral storytelling where we came up with four ways that they relate to one another.

Furthermore, we gave some examples of research done in diverse parts of software engineering to understand and write better documents. With the problem of written documentation, the paper studied waste from Lean software development paradigm and separated the necessary writing from the unnecessary, uncovered the essential elements of West African storytellers and proposed a way to apply these elements in the software production.

As seen in Table 3, the results showed that the elements of audience relationship, story structure, interwoven parables and proverbs, and community relaxation and support can be applied to the parts of the manuscript targeted. Even though this study focused on only five aspects of writing in software development. Future work can include and explore other writing tasks required in software development and possibly view how the other elements discovered can influence as well.

Data availability statement

The original contributions presented in the study are included in the article/supplementary material, further inquiries can be directed to the corresponding author.

Author contributions

PC, MF, and GS contributed to conception and design of the study. AB, AK, OO, and MS conducted literature review. AB and

OO performed the survey. AK and MS analyzed results of the survey. AB, AK, and MS wrote the first draft of the manuscript. All authors contributed to manuscript revision, read, and approved the submitted version.

Acknowledgments

We thank Innopolis University for generously supporting this research.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

- Aghajani, E., Nagy, C., Linares-Vásquez, M., Moreno, L., Bavota, G., Lanza, M., et al. (2020). "Software documentation: The practitioners' perspective," in *Proc. ACM/IEEE 42nd International Conference on Software Engineering, ICSE '20* (New York, NY: Association for Computing Machinery), 590–601.
- Barbosa, R., Silva, A. E. A., and Moraes, R. (2016). "Use of similarity measure to suggest the existence of duplicate user stories in the scrum process," in *Proc. 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshop (DSN-W)*, 2–5.
- Beck, K. (2000). *Extreme Programming Explained: Embrace Change*. Addison-Wesley.
- Burke, Q. (2012). The markings of a new pencil: introducing programming-as-writing in the middle school classroom. *J. Media Lit. Educ.* 4, 121–135. doi: 10.23860/jmle-4-2-3
- Bussell, B., and Taylor, S. (2006). "Software development as a collaborative writing project," in *International Conference on Extreme Programming and Agile Processes in Software Engineering*, 21–31.
- Chaabane, M., Bouassida, L., and Jmaiel, M. (2017). "System of systems software architecture description using the ISO/IEC/IEEE 42010 standard," in *Proceedings of the Symposium on Applied Computing, SAC '17* (New York, NY: Association for Computing Machinery), 1793–1798.
- Chilana, P. K., Singh, R., and Guo, P. J. (2016). "Understanding conversational programmers: a perspective from the software industry," in *Proc. CHI Conference on Human Factors in Computing Systems*, 1462–1472.
- Ciancarini, P., Masyagin, S., and Succi, G. (2020). "Software design as story telling: reflecting on the work of Italo Calvino," in *Proc. ACM SIGPLAN Int Symp on New Ideas New Paradigms, and Reflections on Programming and Software - SPLASH (ACM)*, 195–208.
- Ebert, C., Abrahamsson, P., and Oza, N. (2012). Lean software development. *IEEE Comput. Arch. Lett.* 29, 22–25. doi: 10.1109/MS.2012.116
- Gilson, F., Galster, M., and Georis, F. (2020). "Generating use case scenarios from user stories," in *Proceedings of the International Conference on Software and System Processes, ICSSP '20* (New York, NY: Association for Computing Machinery), 31–40.
- Herbsleb, J., Root, D., and Tomayko, J. (2003). *The eXtreme Programming (XP) Metaphor and Software Architecture*. Technical Report CMU-CS-03-167, CMU.
- Herbsleb, J. D., and Grinter, R. E. (1999). Architectures, coordination, and distance: Conway's law and beyond. *IEEE Softw.* 16, 63–70.
- Hermans, F., and Aldewereld, M. (2017). "Programming is writing is programming," in *Proc. 1st International Conference on the Art, Science and Engineering of Programming, Programming '17* (Brussels: ACM).
- IBM (2023). *What is Software Development?* Available online at: <https://www.ibm.com/topics/software-development#:~:text=Software%20development%20refers%20to%20a,hardware%20and%20makes%20computers%20programmable.,> (accessed May 3, 2023).
- Kelly, P. (2019). "The invention, transmission and evolution of writing: insights from the new scripts of West Africa," in *Paths into Script Formation in the Ancient Mediterranean* eds Ferrara, S., and Valerio, M. (Rome: National Research Council), 189–209.
- Koznov, D. V. (2012). "Teaching to write software engineering documents with focus on document design by means of mind maps," in *Proce. IASTED International Conference on Computers and Advanced Technology in Education, CATE*, 112–118.
- Lamoreux, M. (2005). "Improving agile team learning by improving team reflections [agile software development]," in *Agile Development Conference (ADC'05)*, 139–144.
- Ljung, K. (2021). *Clean code in practice: developers' perception of clean code (Dissertation)*. Retrieved from: <https://urn.kb.se/resolve?urn=urn:nbn:se:bth-21443>
- Lloyd, P. (2000). Storytelling and the development of discourse in the engineering design process. *Design Stud.* 21, 357–373. doi: 10.1016/S0142-694X(00)0007-7

Osman, M. H., and Zaharin, M. F. (2018). "Ambiguous software requirement specification detection: An automated approach," in *Proceedings of the 5th International Workshop on Requirements Engineering and Testing, RET '18*, 33–40 (New York, NY: Association for Computing Machinery).

Sedano, T., Ralph, P., and Péraire, C. (2017). "Software development waste," in *Proceedings of the 39th International Conference on Software Engineering, ICSE '17* (IEEE Press), 130–140.

Stavely, A. M. (2011). *Writing in Software Development*. New Mexico Tech Press.

Tuwe, K. (2016). "The African oral tradition paradigm of storytelling as a methodological framework: employment experiences for African communities in New Zealand," in *African Studies Association of Australasia and the Pacific (AFSAAP) Proceedings of the 38th AFSAAP Conference: 21st century Tensions and Transformation in Africa*.

Utley, O. (2008). *Keeping the Tradition of African Storytelling Alive*. Yale-New Haven Teacher's Institute.

Womack, J., and Jones, D. (1996). Lean thinking: Banish waste and create wealth in your corporation. *J. Oper. Res. Soc.* 48.