A Guidebook for Technical Communicators in Agile Scrum Environments

Danijela Krstic

WRIT 8505

University of Minnesota

Spring 2020

Abstract

Research problem: The lack of understanding and industry consensus on the role of technical communicators in the agile scrum environment. In this study, I completed a literature review of the agile scrum framework and a technical writer's current challenges working within it. I conducted interviews with 10 employees at Sovos, a computer software company, and coded their responses to determine how technical writers currently work within agile scrum, what is working well, and what isn't. I used the findings of this study to develop a guidebook for technical communicators working in agile scrum environments. It can be found in the Appendix of this paper.

Keywords: agile, scrum, technical communication, technical writing, computer software

Introduction

As a technical writer beginning a new job at Sovos, a tax compliance and regulatory reporting software company, I was immediately introduced to phrases and procedures that I'd never heard before. While my graduate studies in technical communication have been robust and covered a wide variety of topics, it is impossible to cover everything. Technical communication emcompasses numerous possible career paths, including technical writing, information architecture, visual design, and many more (STC, 2020). Technical communicators can also have careers in many different fields, such as computer software, engineering, and IT technology, to name a few (Write a Writing, 2011). With all of these possible careers and fields to go into, it is not surprising that I started my technical writing career with some gaps in knowledge. With this study, I hope to fill some of those gaps and provide guidance for technical communicators working in computer software.

One phrase that was new to me on my first day at Sovos was the agile scrum process. Agile scrum was described to me as a procedure used by product and development teams to continuously push software updates out to users every two weeks (Atlassian, 2020). A fuller explanation of agile scrum will follow later in this paper. In brief, agile is "an iterative and incremental based development [process] … meant for short term projects with an effort of team work that follows the software development life cycle" (Sharma et al., 2012, p. 892-893). In agile, there is a high priority placed on customer needs and satisfaction. Scrum is a type of agile development process that focuses on working together as a team to deliver the project "within time and with minimal cost" (Mahalakshmi & Sundararajan, 2013, p. 192). As a member of the product team, I had many questions about this process. How do I fit into this environment? Will I be involved in every scrum meeting and sprint? If so, how am I supposed to get any work done?

These questions and more led me to want to research more about how technical communicators can best fit into this agile scrum environment.

Through initial research surveying online resources, I've found a lack of industry consensus around the issue. In the below graphic (Figure 1) provided by Sovos to explain scrum, the technical writer is noticeably missing. Other scrum graphic variations, shown in

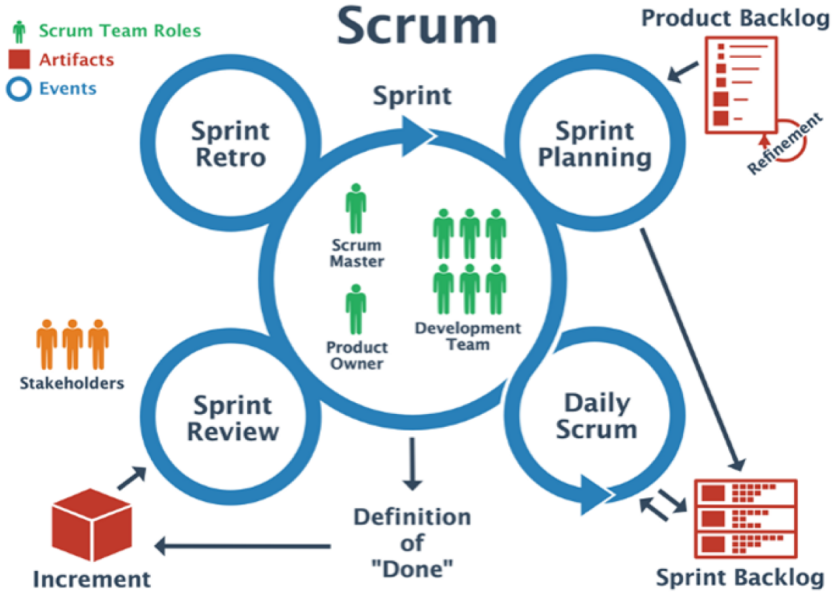

Figure 1. Agile Scrum Roles & Process. Retrieved from Sovos.

Figures 2 and 3, also fail to include technical writers as part of the process. While technical communicators are frequently employed at computer software companies that utilize the agile scrum framework, there are no scrum graphics that include them. In one article, found in the Society of Technical Communication's Knowledge Base, technical communicators are told to report to the "documentation manager" and that membership on too many teams can lead to confusion about which team's work to prioritize (TCBOK, 2019). Another article stresses the importance of technical writers attending all meetings within sprints and implies that the tech writer will only be a part of one development team (Spielman, 2017). One source says to gather information from developers and produce content as quickly as possible (Birgit, 2016), while

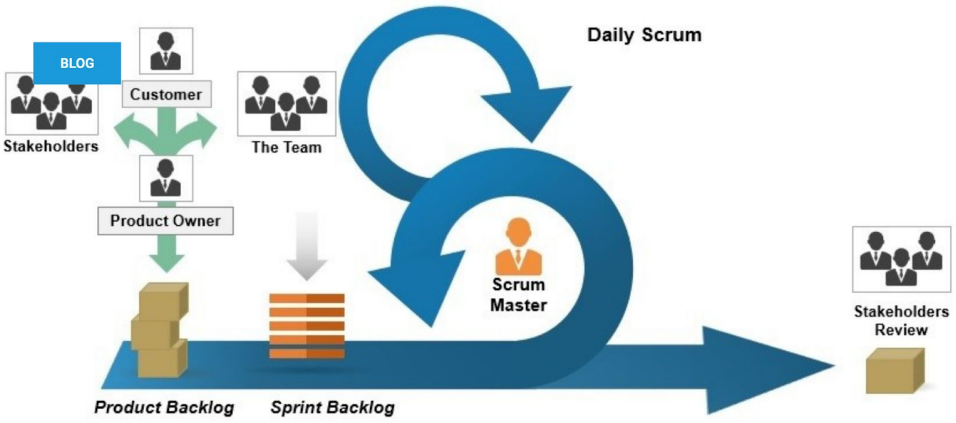

Figure 2. Agile SCRUM. Retrieved from https://hangoutagile.com/agile-scrum/

another says it's important to stay connected to sales and solution engineers to get the user's point of view (Diamond, 2019). These differing perspectives contribute to a lack of understanding on how technical communicator's can best fit into the agile scrum environment. The purpose of this research is to develop a guidebook for technical communicators working in these environments and to provide clarity and guidance for them in their agile scrum roles.

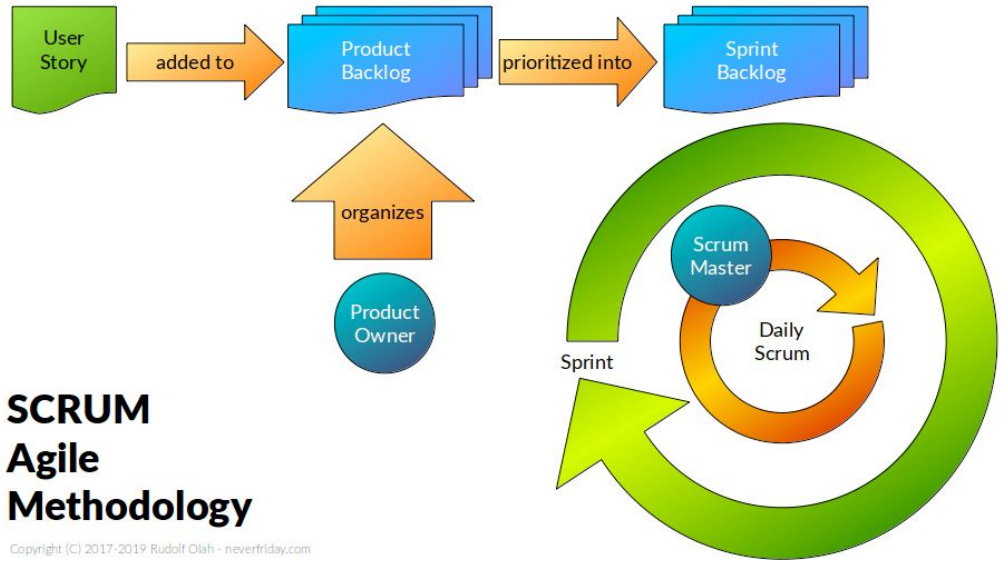

Figure 3. SCRUM Agile Methodology. Retrieved from https://neverfriday.com/2017/11/12/scrum-methodology-diagram/

As a means to provide clarity and guidance, I conducted a literature review of the agile scrum methodology as well as a technical communicator's challenges within this framework. I also interviewed agile scrum professionals, including product owners, product managers, software developers, and technical writers, and present the themes identified from these participants. Lastly, I developed a guidebook based on this research (Appendix).

Agile Scrum

In order to understand a technical communicator's role in an agile scrum environment, we first need to understand the agile scrum environment itself. The agile software process is defined as "an iterative and incremental based development, where requirements are changeable according to customer needs" (Sharma, Sarkar, & Gupta, 2012, p. 892). It places customer satisfaction as the highest priority, divides tasks into small increments, and "is meant for short term projects with an effort of team work that follows the software development life cycle" (pp. 892-893). The general agile cycle is shown in Figure 4 and includes these six steps: requirements, design, development, testing, deployment, and review.

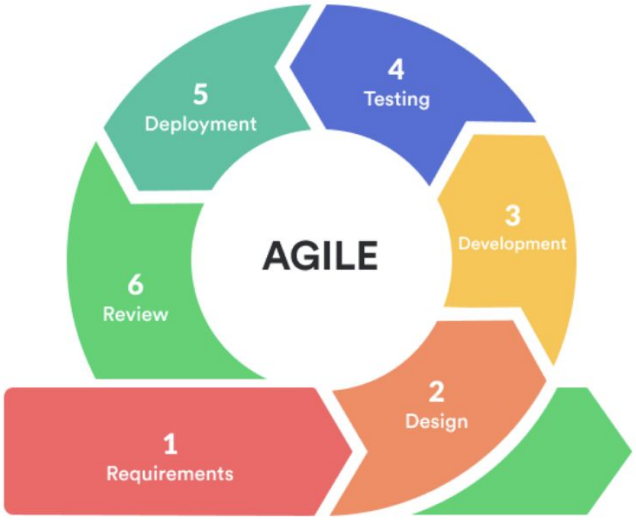

Figure 4. Agile Methodology. Retrieved from https://hackernoon.com/a-case-study-type-insight-into-agile-methodologies-for-software-development-cd5932c6

The agile scrum process can be implemented through multiple methods; scrum is one of the most popular methods. In “Traditional SDLC Vs. Scrum Methodology - A Comparative Study,” Mahalakshmi and Sundararajan (2013) explain that:

> SCRUM was initiated by Ken Swaber in 1995. It was included in agile methodology since it contains the same concepts of agile. A SCRUM is a team pack, where everyone in the team acts together. It delivers the project within time and with minimal cost. (p. 192)

In practice, as shown in Figures 1, 2, and 3, the agile scrum process generally begins with user stories being added to the product backlog. The stories in the backlog could be about features that need to be added to the software, bug fixes, user interface changes, or other requests. These can be initiated by customers/users, product stakeholders, the scrum team, or the product owner. From there, the product owner organizes the backlog and meets with the scrum team to “groom” the backlog and decide which stories they will take on in the upcoming sprint. At Sovos, this meeting is called “sprint planning.” Once the sprint is planned, the development team begins working on stories. During the sprint, the product owner, scrum master, and development team meet everyday to discuss who is completing what and work through any issues that the developers are facing, these are called “daily scrums.” Sprints can last anywhere from one week to four weeks. The Sovos best practice is to complete sprints every two weeks. At the end of each sprint, the team meets again in “sprint review” and “sprint retro” meetings to discuss what went well, what didn’t, and how to improve for the next sprint. Then the sprint process starts over again.

The roles included in a sprint as described above are the product owner, the scrum master, and the scrum team. Figure 5 is from Sovos training materials and describes the big picture idea behind each role. Product owners “build the right thing,” development teams “build

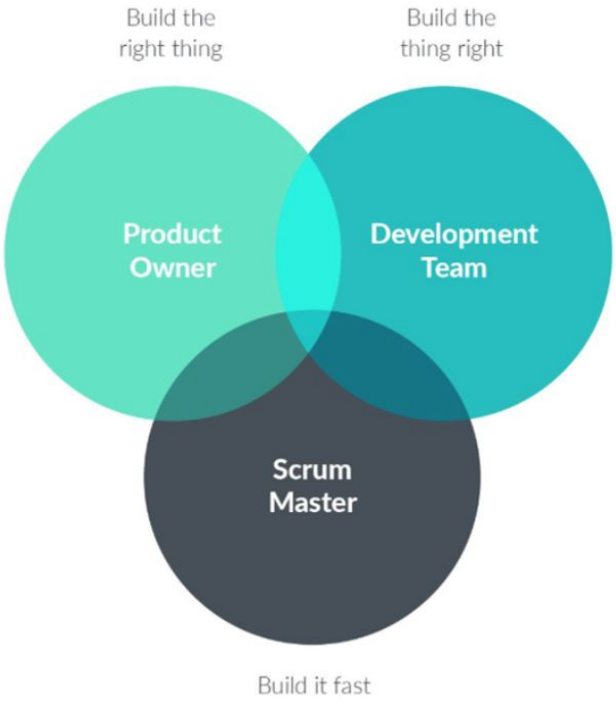

Figure 5. Scrum Roles. Retrieved from Sovos.

the thing right," and scrum masters "build it fast." In more detailed terms, the product owner acts as a customer representative and maintains the product backlog, determining which stories have priority over others. They define the features of the product and are responsible for final approval of the team's work. The scrum master occupies a leadership role over the scrum team members. They run the sprint planning, daily scrum, and sprint review/retro meetings and help team members overcome challenges during the sprint. The scrum team are the people who develop the product. Scrum teams are typically made up of five to ten members and each team member is responsible for their own work; team members can be programmers, testers, or user experience designers (Mahalakshmi & Sundararaja, 2013).

There are both advantages and disadvantages to using the agile scrum process. Mahalakshmi and Sundararajan (2013, p. 195) note the following advantages:

- Increase in customer satisfaction by optimizing turnaround and responsiveness to requests
- Increase in quality of work
- Provides better work estimates while spending less time creating them
- Provides a high level of control over the project schedule
- Changes are expected and accepted

The agile scrum framework also faces some issues and challenges. Complaints include the large number of meetings, leading to some feeling like a waste of time, and projects failing due to poor communication and/or collaboration by one or more team members (Cho, 2008; Mahalakshmi & Sundararajan, 2013). A huge disadvantage in the agile scrum environment involves documentation, specifically the lack of documentation. In “Issues and Challenges of Agile Software Development with Scrum,” Cho (2008) found that:

> The idea behind reducing documents in the agile method is to keep every team member equal by sharing skills and knowledge on the systems. In that way, if one person leaves, there is still a lot of shared knowledge that has gone around among other team members, so it is not a big deal. However, in reality, this is not feasible. (p. 193)

Part of the agile methodology involves reducing the amount of documentation for a product and stresses that the code itself should be a document (Cho, 2008). This ideology can work for current developers who are creating the product and know how to use it, but doesn’t consider new hires who could benefit from reviewing documentation as well as the users, who usually need additional guidance and support outside of the product itself.

Challenges for Technical Communicators in Agile Scrum

In the agile scrum environment, the only official "scrum roles" that are accounted for and included in graphic representations are product owners, scrum masters, and the scrum team. Technical communicators do not have a clear place in this framework and therefore, their role can differ depending on the company or team. Because of this, technical communicators face an uphill battle working in this environment and are faced with a lot of challenges to overcome. One of the more obvious challenges is the time sensitivity of documentation work in agile scrum environments. With the sprint process taking around two weeks, any documentation updates needed for that sprint must be completed within that time as well. The documentation and software updates should be released together so users have the most updated resources that match the product. As Dawson (2009) puts it: "with the developers choosing accessible goals and shipping them regularly, as a writer you feel that you're rushing." While it is necessary in most writing jobs to work quickly at times to meet deadlines, feeling rushed by others can be stressful. Technical writers in agile scrum environments currently need to go out of their way to communicate well with scrum teams in order to get quality documentation done on time without feeling overly rushed or pressured to publish something below their standards.

Technical communicators working in agile scrum environments also have to be aware of the agile documentation life cycle. Figure 6, below, displays an example of this cycle. The main steps it represents are planning, information gathering, execution, and quality analysis (Bidkar, 2011). The cycle estimate in Figure 6 assumes that the feature needing documentation will be completed in two sprints. The "Sprint Backlog" and "Estimate" boxes represent the planning portion of the cycle. "Interview" represents information gathering and "Write" represents execution. "Review" and "Test" are the quality analysis portions of the cycle. While having a cycle like this to base documentation work off of can be helpful, it also raises a lot of questions

and challenges. For one, estimating the effort of the documentation can be extremely challenging. Often, the sprint backlog will contain brief information about new features with few details, making estimation difficult for technical communicators (Bidkar, 2011). The “Interview” portion can also be troublesome when teams are overwhelmed with work and don’t view documentation as a priority. Scrum team members may be reluctant to take time out of their busy work days to talk to writers and walk them through the changes that are being made. Even if technical communicators manage to have productive interviews with adequate information gathering, some documentation requires the use of screenshots of the software interface. If the features are being developed during the sprint and won’t be finished until a day or two before

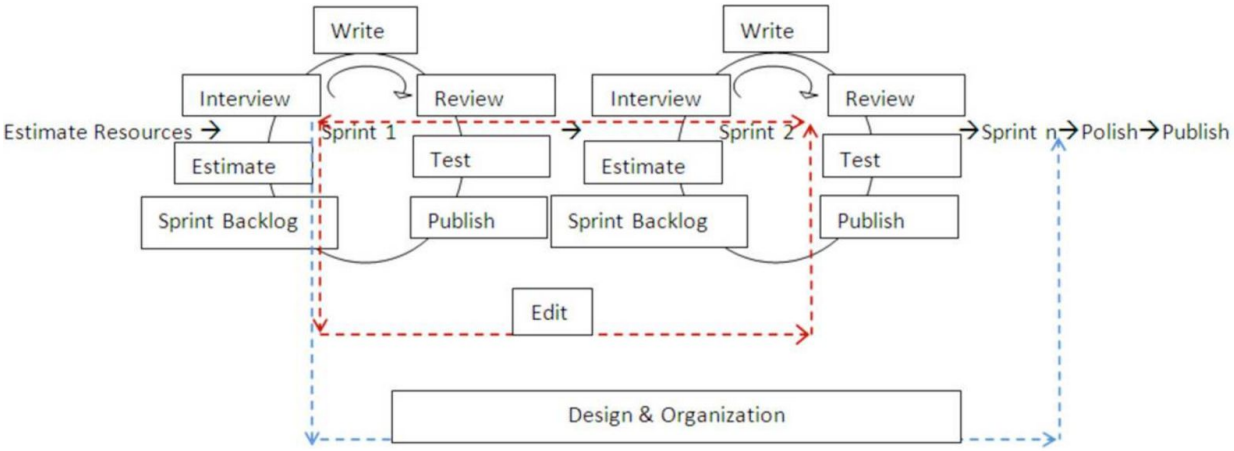

Figure 6. Agile Documentation Life Cycle. Retrieved from http://www.tcworld.info/e-magazine/technical-communication/article/adjusting-to-scrum/

the end of a sprint, that gives the technical communicator very little time to go in and get the required screenshots before everything needs to be completed. And if the writing is not able to be completed until right at the end of a sprint, the review and quality analysis portion of the cycle must be completed in a very short window of time, if it ends up being completed at all. As it stands, the agile documentation life cycle from Bidkar (2011) seems mostly aspirational, with

many problems to overcome before technical communicators are able to use it as a true documentation cycle guide.

Methodology

This IRB exempt study[1] explored the procedures and practices of technical writers in an agile scrum environment at Sovos.

Participants. For the study, I interviewed 10 participants. They are all Sovos employees; two are technical writers and the rest are product owners or software developers. Participants were recruited from my professional network at Sovos, eight work at the Minnetonka, MN office, one is out of the Boston, MA office, and the other is out of the Boulder, CO office. The group consisted of four females and six males ranging in age from 26 to 45. I chose these participants because they are either technical writers, or have a role in which they directly interact with technical writers in the agile scrum environment. I wanted to get different perspectives from people who have first-hand knowledge and experiences with agile scrum.

Data Collection. I collected data through participant interviews. The interviews were structured around four pre-determined and open-ended questions, designed to elicit conversation through thoughtful and detailed responses (see Table 1). Interviews occurred in February 2020 at a conference room at the Sovos office in Minnetonka, MN, with the eight local participants being interviewed in-person and the two out-of-state participants being interviewed virtually via Microsoft Teams. Interviews were audio recorded and transcribed using temi.com

[1] IRB STUDY00008868 exempt on Feb 18, 2020.

Interview Questions. Based on Turner's *Qualitative Interview Design: A Practical Guide for Novice Investigator*s (2010), I developed interview questions following a standardized open-ended interview approach:

> The standardized open-ended interview is extremely structured in terms of the wording of the questions. Participants are always asked identical questions, but the questions are worded so that responses are open ended. This open-endedness allows the participants to contribute as much detailed information as they desire and it also allows the research to ask probing questions as a means of follow-up. (p. 756)

I developed these few open-ended questions as a means to allow the participants to fully express their viewpoints and experiences. Questions 2a and 2b act as follow-up questions based on the participants' response to question 2. They may be asked just 2a, just 2b, or both if their response to question 2 indicates both successes and challenges. With these interviews, I attempted to get Sovos employee's thoughts and opinions on how technical writers currently work within the agile scrum environment there, what works well, what doesn't, and what they think could improve the process. Participant's first-hand knowledge and experiences with agile scrum and technical writers at a computer software company will provide hugely constructive information. The themes I identified from these interviews acted as a valuable resource. Participant's opinions and ideas provided important guidance on a technical communicator's role in agile scrum as I developed my guidebook.

Table 1. Interview Questions.

Question Number	Question
1	Describe how technical communicators/writers currently work within the agile scrum environment at Sovos.
2	Are the current practices you just described successful? Why or why not?
2a	What makes them successful?
2b	What could be changed to improve these practices?

Analysis & Coding

I used an inductive qualitative analysis approach to analyze my interview data, specifically, a thematic content analysis. According to Canary (2019), with a thematic content analysis, "rather than approaching your data with a predetermined framework, identify common themes as you search the materials organically. Your goal is to find common patterns across the data set." After conducting all of my interviews, I carefully read through each transcript, annotating them as I went. I labeled relevant words, phrases, and sentences with codes relating to different concepts, opinions, and processes. I then created categories by grouping the codes I created. Next, I segmented my data by "positioning and connecting" my categories (Canary, 2019). This allowed me to draw connections between the categories and develop broader themes from the interviews that helped define key takeaways and ideas for my guidebook.

Findings & Discussion

Below, I present the findings of my qualitative data results, including the code/category for each interview question, as well as the overall themes found from the interviews. Specifically, I address the research problem of there being a lack of understanding and industry consensus around the role of technical communicators in the agile scrum environment.

Question One: Patterns of Inconsistency. The purpose of Question One was to find out how participant's see technical writer's current role at Sovos by prompting them to: "Describe how technical communicators/writers currently work within the agile scrum environment at Sovos." Participant responses varied, there was an inconsistency and confusion present in how they thought technical writers worked within agile scrum. Two participants said that at the beginning of each sprint, product owners create tasks in Jira, a software commonly used in agile project management to track the scrum teams work, for the technical writer. The writer then reaches out to the product owner or to developers if they need more information. They are expected to finish their work by the end of the two week sprint. Five other participants said that the product owner is supposed to email the technical writer when they have work for them. The technical writer then creates the task in Jira themselves and completes the work at their own pace or at another established due date, the work is not tied to a sprint cycle. The last three participants said that they are not exactly sure how technical writers currently work within agile scrum. They are under the impression that the technical writer's work is separate from normal developer sprint work but they know that Jira tasks are created to track their work. When further prompted, they did not know who was supposed to create the tasks in Jira.

Participants in my study have differing ideas of how technical writers work in agile scrum at Sovos. Eight of the ten participants work in the same department and these inconsistent responses illustrate a lack of understanding at the company surrounding tech writers and agile

scrum. These results must be taken into the context of my limited participant sample size. These results are indicative only of how technical writers work within agile scrum at Sovos. Future researchers exploring this subject should interview a larger participant sample from numerous companies in order to compare and verify results across the industry in general, as opposed to just at an individual company.

Question Two: Patterns of Ineffectiveness. The purpose of Question Two was to follow up on participant's responses to Question One and find out how well they think the technical writer agile scrum process is working by asking: "Are the current practices you just described successful? Why or why not?" Depending on their answer to this initial question, participants were asked either one or both of the following questions:

2a. What makes them successful?

2b. What could be changed to improve these practices?

The consensus from participants was that the current practices are not successful. As the responses to Question One display, participants are not on the same page about what the actual practices are for involving a tech writer in a sprint, leading to a confusing and ineffective process. Some specific examples were: Jira tasks not being created due to a lack of understanding about who should create them, product owners and developers not having enough time to adequately respond to questions from the tech writer, and tech writers either being too involved or not involved enough in daily sprint meetings.

Participants had many ideas about changes that could be made to improve these practices. They all wanted to establish specific guidelines about who is creating tasks in Jira and when to create them. Some thought the product owners should create them while others thought it should be the tech writers responsibility. Additionally, most thought that after these tasks were created, the tech writer, product owner, and developers should have set meetings,

outside of the daily sprint meetings, in order to figure out what exactly needs to be completed by what date and allow the tech writer to ask any content-related questions that they may have. This way, technical writers can still meet regularly with the scrum team but in a documentation-focused manner. There is no need for them to attend every scrum meeting since they usually end up wasting their time, these new meetings at the beginning of each sprint would solve this issue and allow the tech writer to work more quickly and effectively. If more information is needed or it's a longer project and check-ins are required, this meeting can also be used to organize those on future dates. Lastly, participants were interested in having a more formal review process of the tech writers work. They thought that this could include review by other tech writers for formatting, style, language, etc. as well as a review by the product owner or certain developers in order to check for specific content and wording.

As with Question One, these results are from a limited sample size of Sovos employees. Further research could be done to explore patterns throughout the larger software company industry utilizing the agile scrum framework.

Themes

Room for Improvement. All participants directly stated that the current practices used for technical writers in the agile scrum environment are not successful. Whether because of a lack of consensus around creating tasks or scrum team members not making time to answer tech writer's questions, there is great room for improvement in the process. The current practices are not effective and need to be reworked and improved in order for tech writers to be able to work more efficiently within the agile scrum framework.

Need for Effective Meetings. A suggestion by most participants was to have specific documentation meetings at the beginning of each sprint with the technical writer, product owner, and some developers. These meetings will allow everyone to get on the same page about the work that needs to be done and answer any questions that the tech writer might have. Previously, tech writers were invited to daily scrum meetings but were unsure about when to attend as most meetings did not involve the work they were doing. With a new specific documentation meeting, tech writers wouldn't have to waste valuable work time attending daily scrum meetings and can instead have a focused discussion on the documentation efforts needed for that sprint.

Importance of Review. All participants also noted the need for a specific and formal review process. While most had not had any major issues with tech writer's work, there was some confusion about who's approvals were needed for what work and they thought it would be beneficial to establish some set guidelines. Some participants stressed the importance of a review by other tech writers to check for general formatting, style, and language use. Others wanted a review by the product owner or a developer to check for content and specific technical terminology. I would suggest that reviews by both tech writers and subject matter experts would be extremely useful.

Conclusion

This research illustrates how technical communicators can be better utilized in agile scrum environments at computer software companies. By having specific documentation focused meetings and getting tech writers involved early on in each sprint, the entire scrum team benefits. Additionally, establishing a formal review process ensures that all documentation is accurate and can be published on time. I hope this research can act as a guide for technical communicators and anticipate that future research can build upon this and provide insights from other companies on how technical writers can best succeed in agile scrum environments.

References

Atlassian (2020). Scrum. Retrieved from https://www.atlassian.com/agile/scrum

Bidkar, P. (November, 2011). Adjusting to Scrum. Retrieved from http://www.tcworld.info/e-magazine/technical-communication/article/adjusting-to-scrum/

Birgit (2016, November 30). What It's Like to be an Agile Technical Writer. Retrieved from https://developer.epages.com/blog/methods-and-tools/what-its-like-to-be-an-agile-technical-writer/

Canary, A. (2019, October 10). How to Analyze Interview Transcripts in Qualitative Research. Retrieved from https://www.rev.com/blog/analyze-interview-transcripts-in-qualitative-research

Cho, J. (2008). Issues and Challenges of Agile Software Development with Scrum. *Issues in Information Systems 9*(2), 188-195. Retrieved from https://pdfs.semanticscholar.org/713c/41d6f352f97a94af1af94c56e0ba2584effd.pdf

Dawson, E. (2009, August 25). Technical Writing in Agile Software Development - Part 1. Retrieved from https://www.atlassian.com/blog/archives/technical_writing_in_agile_software_development_part_1

Diamond, A. (2019, January 25). 10 Tips for Technical Writers in an Agile Environment. Retrieved from https://medium.com/technical-writing-is-easy/10-tips-for-technical-writers-in-an-agile-environment-7f26ac468ff9

Mahalakshmi, M. & Sundararajan, M. (2013). Traditional SDLC Vs. Scrum Methodology - A Comparative Study. *International Journal of Emerging Technology and Advanced*

Engineering, 3(6), 192-196. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.413.2992&rep=rep1&type=pdf

Sharma, S., Sarkar, D., & Gupta, D. (2012). Agile Processes and Methodologies: A Conceptual Study. *International Journal on Computer Science and Engineering, 4*(5), 892-898. Retrieved from https://www.yashada.org/yash/egovcii/static_pgs/TC/IJCSE12-04-05-186.pdf

Society for Technical Communication (STC) (2020). Defining Technical Communication. Retrieved from https://www.stc.org/about-stc/defining-technical-communication/

Spielman, R. (2017, November 29). Fitting Technical Writing into Agile Development. Retrieved from https://www.agileconnection.com/article/fitting-technical-writing-agile-development

TCBOK (2019). Agile Development. Retrieved from https://www.tcbok.org/wiki/information-management/agile-development/

Turner, D. W. (2010, May). Qualitative Interview Design: A Practical Guide for Novice Investigators. *The Qualitative Report, 15*(3), 754-760. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.674.8927&rep=rep1&type=pdf

Write a Writing (2011, May 4). Best Technical Writing Fields. Retrieved from https://www.writeawriting.com/technical/best-technical-writing-fields/

Appendix

A Guidebook for Technical Communicators in Agile Scrum Environments

As you begin working in agile scrum, keep the following Do's and Don'ts in mind.

Reference the modified scrum graphic, below, for further guidance and instruction.

Do	Don't
• Schedule specific documentation meetings at the beginning of each sprint. You, the product owner, and some software developers will use these meetings to align on the documentation efforts that are needed for that sprint and schedule other check-ins or demos as needed. • Review other tech writer's work and have tech writers and scrum team members review your work before publishing the final version. Getting different eyes on the documentation will allow for the most accurate work possible.	• Attend every sprint meeting. Most of these will not discuss documentation and will be a waste of your time. Focus your efforts on productive documentation meetings, completing your work, and reviewing others work as needed. • Create your own Jira tickets. Scrum team members should be responsible for notifying you when something they are working on requires documentation. Track these tickets as you complete work to keep team members updated on your progress.

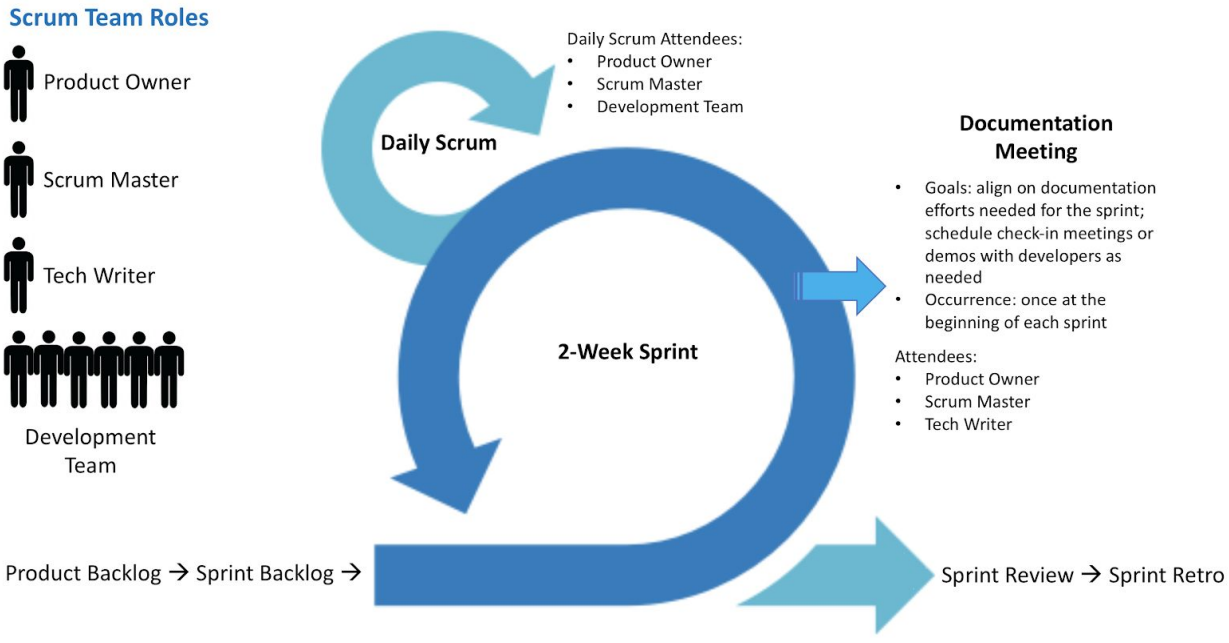

Figure 7. Scrum for Technical Communicators. Base Scrum Graphic retrieved from: https://www.pluralsight.com/paths/the-scrum-framework