7/20/2020

The Contributions of Technical Communicators to Developer Portals and the Developer Experience

Robert A. Bowman and Sonia Verma
GRADUATE STUDENTS, TCO 685, MSTCM, MERCER UNIVERSITY

Abstract

Purpose

To learn how technical communicators = are contributing to the developer experience by trying to understand how they are assessing the value of their contributions, whether they are contributing to other aspects of the developer experience outside of API documentation, and if they can do a better job of articulating those contributions.

Methods

A mixed methods approach, with a survey delivered as a questionnaire. The survey had 19 total questions with a mix of open-ended, single-select, and multi-select options. The survey was posted to the Write the Docs Slack community, in a channel specific to API documentation.

Results

Most technical communicators who answered the survey feel that the developer portals they maintain provide most of the information a developer would need to use the API. They also feel that they are contributing to many aspects of developer portals outside of just API documentation, such as user advocacy, usability testing, analytics, product management of the developer portal.

Most feel valued by their organizations, meaning that they are articulating their value, but need to improve on this to be better seen as part of the developer experience team. However, many said that they wished to contribute more to the developer experience but are either overloaded with work or lacked programming skills or technical knowledge.

Conclusion

Technical communicators are making contributions to the developer experience with documentation and other aspects such as product management and user advocacy. They feel valued by their organizations, but wished they had additional technical skills and could contribute more.

Keywords

API documentation, Developer Portal, Developer Experience, SDK documentation, API Writers

Introduction

Developer Experience, often initialized as DX, can be defined as anything that makes it easier for a developer to use an (*7 Mantras For Quality Developer Experience | Nordic APIs |*, 2019), and consists of the experiences related to the artifacts and activities encountered as part of software development (Fagerholm & Münch, 2012)

Everyone resonates with products and services that can be used with maximum ease. For a developer visiting a developer portal, which can be defined as a platform that houses all the information a developer might need to use an API (Application Programming Interface), the API itself is the product. A good developer experience encourages developers to use the API and reduces training time and support costs, while a bad developer experience discourages developers from using the API and increases the support and training costs associated with the API (van Tomme, 2018).

Developer portals play multiple roles in developer experience, such as articulating the organization's unique value proposition, clearly stating the use case for each API including the types of problems the API solves and establishing trust in the API provider as a business. Developer portals can also be used to build confidence in the security and reliability of the API, along with providing developers all the tools needed to quickly determine if the API is suited for their purpose and how to start using the API (Santos, 2019).

Multiple studies, as discussed in the literature review, have focused on the importance of API documentation and how bad documentation can pose a threat to the adoption of the API. For example, Robillard (2009) found in his research with developers at Microsoft that 50 out of 74 respondents cited unclear or incomplete API documentation as one of the main obstacles to learning.

However, a developer portal encompasses more than the API documentation. Following are the key components of an ideal developer portal:

- Overview pages – Landing page and pages that have high-level information on the APIs. These pages attempt to answer 'what APIs are available and what can I do with an API' questions.
- Guides and tutorials - Getting started and other tutorials that answer 'how do I solve my problem with the API' questions.
- Example code - Collection of use-case based examples that focus on potential issues that the API can solve. Note that the examples should also be provided in commonly used languages.
- SDKs - Information on Software Developer Kits (SDKs) that a developer can drop in their app and get started with minimal integration effort.
- API references - A reference of all endpoints and parameters with their description and limitations.
- Test environment – Sandboxes and API explorer to test and experiment with the APIs.
- Support resources - Community pages, contact details of the API support team, blogs, etc.

Technical communicators (TCs) are often an important part of the DX, which in turn plays a role in how developers decide on things like which API to use for a project (Myllärniemi et al., 2018). However, there is little data on how TCs are contributing to the overall DX, as opposed to issues directly related to API documentation.

Our research assumes that most TCs currently working on developer portals are aware of the elements of good API documentation. We assume that along with basic technical communication skills, they also possess the other skills of a good API writer such as the ability to read code in at least one language, ability to test the sample APIs that they document (Society for Technical Communication, 2014), and how to use popular docs-as-code tools (Gentle et al., 2017), etc. However, there is little information available on how TCs contribute to the DX. One known contribution is the API documentation, but there is not much knowledge about TC's contribution to other aspects of developer experience or if TCs can contribute to other aspects. The quality of a TC's contribution is dependent not only on the skills and experience of the TC, but also on the team's perspective and value of the TC's contribution. We also wanted to study how involved TCs are in the process of DX and the challenges they face.

Through our research we are trying to find answers to following research questions:

- How do technical communicators assess the influence and usefulness of their contributions to the developer experience?
- Are technical communicators contributing to other aspects of developer experience?
- Can technical communicators do a better job of articulating their influence or usefulness to the developer experience?

This research has been approved by the Mercer University IRB.

To better understand the current state and the challenges, we first reviewed the existing research done by academics and industry practitioners on this topic. In the literature review, we give an insight to the findings from the previous research and other articles written on API documentation and related topics. We then explain our data collection and analysis methodology. We share the findings through different data presentation modes and key highlights from participant quotes. We end the report with our conclusions, research limitations, and the possibilities of future research on this topic.

Literature Review

Overview

The literature review was conducted using a search of peer-reviewed articles and conference papers available through the Mercer University online library, and through online searches for relevant keywords to find websites, blog posts, and other online content. Note that physical library access, both university and public, was unavailable at the time the research was conducted (May - July 2020) due to the closure of facilities during the COVID-19 pandemic.

Search keywords and methods

A university library database and journal search for “developer experience” or “software developer experience,'' found a few peer-reviewed journal articles and conference papers available, but these articles only mention documentation as an artifact that is required for the developer experience (DX). There was no specific mention of technical communicators (TCs).

Searching for related topics, such as “developer portals” and “API documentation,” which can be part of the components of a DX, found more peer-reviewed articles that placed TCs within the overall DX framework. However, the number of articles and the amount of research currently available did not seem to be enough to answer the research questions.

For this reason, searches were also made for websites, blog posts, industry articles, and other content available on the Internet related to the concepts of developer experience and technical communicators, using the same keywords as the database searches. However, because there is no standardized way to vet websites, blog posts, or other Internet content for credibility, determinations on which content to use had to be based on other factors. For example, we used Alexa rankings showing how many other sites linked to the site being considered, the age of the website, the overall reputation of the author (for blogs or articles), the subjective quality of the content available. We also used the fact that as TCs the online content we referenced are among the websites we frequent in our own day-to-day professional work.

Concepts and themes discovered

The concepts surrounding developer experience and its relationship to TCs are new. There are few peer-reviewed journal articles and conference papers available on the subject. For example, we found only conference papers on developer experience from Fagerholm and Münch (2012) and Fontão et al. (2018) and an article from Myllärniemi et al. (2018) on how the developer experience impacts the adoption of software frameworks.

Other articles related specifically to DX were about the concept of experience as related to skill-level (Eyolfson et al., 2011; Latorre, 2014), or about the experience of using a specific software tool (Kuusinen, 2016). The article by Fagerholm and Münch does suggest that both DX as a concept and the experience, or skill-level, of the developer are interlinked (2012), the only article to do so.

Developer portals and API documentation, which can be part of a developer experience, have more peer-reviewed articles available that placed the contributions of TCs within the overall developer experience framework (Meng et al., 2018; Myllärniemi et al., 2018). Several of these articles acknowledged that inadequate documentation is a major challenge for software developers in learning an API (Meng et al., 2018; Watson, 2012) and that quality documentation shortens the learning curve for learning an API (Robillard & DeLine, 2011).

However, the general theme is that these contributions are a given, with little consideration as to the process and methods used by TCs or how those contributions are improving the overall developer experience.

Themes related to the research questions

For this reason, our research focused on understanding how TCs are determining the influence and usefulness of their contributions to the DX and how well they are articulating this influence. We also looked at what contributions TCs are making to other aspects of the developer experience outside of things like API documentation.

Looking at the available peer-reviewed literature for developer experience, we found that the available research only mentions documentation as an artifact, with little consideration or discussion as to how that documentation was created or the process that led to its creation.

Literature about API documentation and developer portals does acknowledge the work of TCs and places them within the overall developer experience (Meng et al., 2018; Robillard & DeLine, 2011), but without consideration as to how TCs can contribute or are contributing to the developer experience.

None of the peer-reviewed articles acknowledged the challenges faced by TCs in their crucial role of creating, curating, and maintaining the information required for a developer portal.

Articles and websites from industry are more likely to recognize the value of the contributions that TCs make, since those contributions can have a direct impact on the success of the product or service being sold (*API Developer Experience (DX) Resources*, 2020). The work of TCs outside their traditional roles is also acknowledged (Johnson, 2020), which speaks about the influence that TCs have on the developer experience as a whole outside of the writing function.

Most of the articles found for the literature review are from outside the technical communicator's point of view and are from the viewpoint of someone who only sees the documentation as an assumed artifact, even when it is assumed that the documentation is necessary for success.

Articles written by TCs articulate their usefulness in the overall developer experience, by showing how they help with developer onboarding via usability (Society for Technical Communication, 2014, p. 18), by creating code samples for API documentation (Society for Technical Communication, 2014, p. 20), and by ensuring that their documentation is adding value beyond what can be gathered just from reading code and code comments (Society for Technical Communication, 2014, p. 24). However, these articles mainly appear in professional and non-peer-reviewed journals.

The literature review shows that while the influence and contributions of TCs are acknowledged by those outside the profession, more systematic research must be done by TCs to better showcase and articulate their influence on the developer experience.

Methodology

For our research, we looked at the role of technical communicators (TCs) related to the developer experience (DX), which can be defined as a way to understand how software developers think and feel about their activities in their working environments (Fagerholm & Münch, 2012). We focused our research on the following questions:

- How do technical communicators assess the influence and usefulness of their contributions to the developer experience?
- Are technical communicators contributing to other aspects of the developer experience?
- Can technical communicators do a better job of articulating their influence or usefulness to the developer experience?

Research methods

Our research used a mixed methods approach, with a mix of quantitative and qualitative methodology. We used an online survey administered as a questionnaire to collect data from TCs who work on APIs. By using a survey, we were able to gather more responses in the allotted time available for data collection, and this also allowed us to get responses from TCs around the world who were able to answer anonymously at their own convenience.

Sample selection

For our study, the population consisted of TCs who work on API documentation and who are a part of the Write the Docs (WtD) #documenting-apis Slack channel. This channel has approximately 1,500 members, out of the approximately 9,500 total members of the Write the Docs Slack workspace (https://www.writethedocs.org/slack/).

Data collection

The survey was designed using Google Forms. This method was chosen because the researchers were using Google Drive to collaborate, so having all the data collected in one location made it easier to analyze and review.

Before the actual survey began, we piloted our survey with two TCs who worked on developer documentation and were known to one of the researchers. The response and feedback from these two TCs helped the researchers to make small edits to two questions which were not clear.

The survey was posted to the WtD Slack channel on June 24, 2020. Reminders were sent on June 29 and July 3, and the survey closed on July 5, 2020. When the survey closed, we had received 34 responses.

The survey had 19 questions (see Appendix A), of which 13 were multiple choice, two used a Likert-scale rating of 1-5, and four were open-ended. All questions had the option for the respondents to enter in their own free-form answers in case they wanted to share more information about that question or a related process. The survey concluded with an open-ended question to allow the respondents to share any related or relevant information about their experience that was not asked for in the survey.

Data analysis

Our survey had 19 questions which can be categorized in three different types: questions that allowed free-form answers, questions where respondent could select multiple options, and questions where respondent could select only one option. These questions were related to the three research questions and can be categorized into two main categories: questions focused on eliciting information on the documentation process and questions focused on eliciting information on the technical communicators' contribution to developer experience in their organization.

- For the multiple-choice questions where the respondent selected from the answers given, we looked at the number of times that the response was chosen, and at the frequency that response was chosen.
- For the questions where the respondent gave a free-form answer, we used an open-coding approach to understand the general categories and themes present in the data.
- For the questions where the respondent could select only one option, we used a frequency distribution.

Throughout the data collection and analysis process, only the researchers had access to the data. We did not collect any personally identifiable information (PII); however, to maintain the confidentiality of the respondents we assigned the respondents a code with a mix of alphabet and number characters such as R1, R2, etc. Participant quotes in the Findings section of the report use these codes to refer to the participants.

We received 34 responses in total, but 3 respondents do not work on developer documentation and 1 respondent worked as a consultant. Therefore, our actual respondents were 30. After we stopped accepting responses to the survey, we exported them to an Excel Sheet. We then used this sheet to assign codes to the free-form answers (qualitative data) and group the codes to create themes that link to the categories and subcategories listed below. We also used quantitative data to find relationships between the questions that influenced each other.

Once we completed our data analysis, we used the following categories and subcategories to group our findings and discussion section:

- Documentation process
    - Content creation
    - Content testing and review
    - Content publication
    - Content management and maintenance
    - Interaction with another teams
- Technical communicator contribution
    - Current contribution and involvement
    - Current valuation and importance
    - Challenges
    - Opportunities

Results and Discussion

We gathered data from 34 respondents, out of which three did not work on developer documentation and one worked as independent consultant. So, we had 30 respondents including two from the pilot survey. None of our survey questions (Appendix A) were mandatory. Consequently, some respondents did not answer all the questions. Therefore, the total number of respondents (N) varies for each question.

Respondent characteristics

Before we present our findings, we have narrated our respondents' characteristics. Understanding these characteristics is important because these characteristics influence the responses that have been selected. As a result, the findings and discussions are likely to be skewed towards the writers who have similar characteristics.

The job title of 'technical writer' is held by 22 out of 30 respondents. Even though four respondents' official job title was documentation manager, they were also partly responsible for writing documentation along with other managerial tasks. Nine respondents also worked on other related aspects such as providing support to the support engineers; coordinating other documentation tasks such as meetings, release plans; troubleshooting writing tools issues; managing wiki; peer reviews, and so on.

Table 1: Respondent's API documentation experience vs. term at current company

Term at current company (years)	**API documentation experience (years)**			
	0-2	**3-5**	**6-10**	**10+**
0-2	10	7	-	-
3-5	2	4	-	1
6-10	1	-	1	1
10+	-	1	-	1

Nine respondents were lone writers and 11 worked in a small team of writers, i.e. 2-9 writers. Six respondents worked in a team of 10 - 20 writers, and four respondents worked in a team of 40 or more writers.

Most respondents (15) who contributed to other aspects of the developer portal were also either the only writer or were a part of a very small team of writers (a team of 1-5). We infer from this data that TCs who work either alone or in a small team are working for small to medium companies where they get an opportunity to contribute to other areas as well as develop their skill sets in other areas.

However, many of such respondents also selected work overload as one of their challenges. TCs who worked in bigger teams work in large setups that might have predefined documentation processes and content structure.

Since 25 respondents have 0-5 years of experience and 19 have spent only 0-2 years in their current company (see Table 1), the findings of this report might be skewed more towards the TC population with similar characteristics.

Almost half of the respondents (13) are new to API documentation while 12 respondents have 3-5 years of experience in API documentation. Four respondents have been in their current company for a longer period than the experience they have in API documentation which also indicates that these writers were only involved in UI-based or product interface documentation at the beginning of their time at their company.

Research question 1

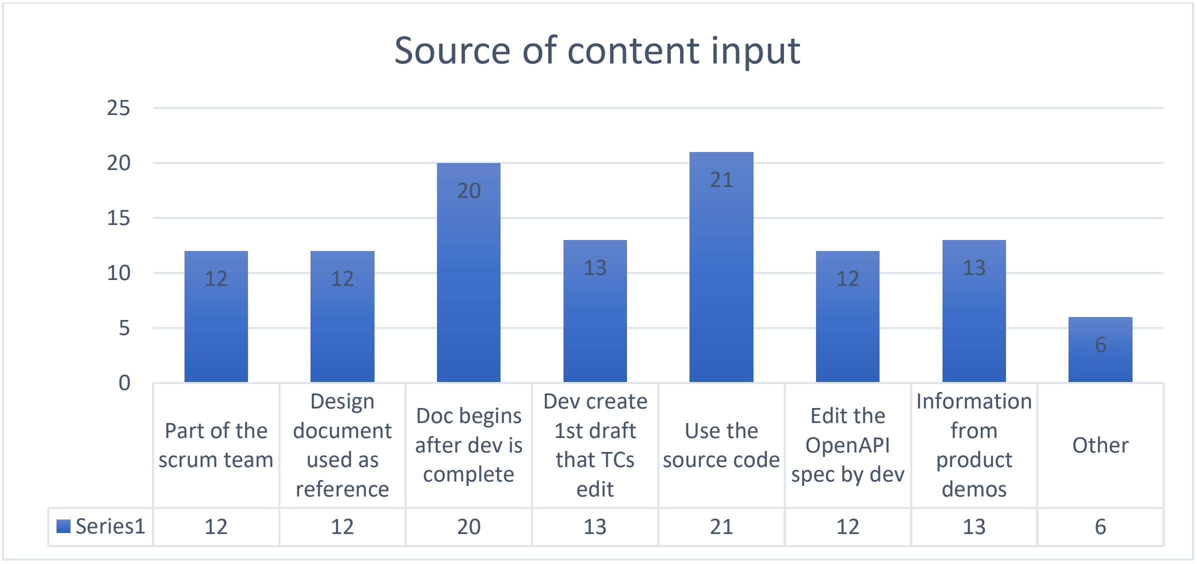
We have divided our findings and discussions based on our research questions. Our first research question was "How do technical communicators assess the influence and usefulness of their contributions to the developer experience?"

To answer this question, we categorized our findings into following subcategories: content creation process, content testing and review, content publication, content management and maintenance, and interaction with other teams.

Note that for each chart the numbers (N) may not add to 30 (total number of responses) since respondents were able to select multiple answers or no answer.

Content creation process

To gather inputs or to create the first draft, writers used a mixed approach than instead of relying on just one source (Figure 1). One of the sources of information by 21 respondents is the source code. Twelve respondents used the API design doc that the architect or a developer creates or 13 used the product demos as a reference point for their documentation. 13 respondents also mentioned that in their teams, developers created the first draft and writers edited the draft. Similarly, 12 respondents edit the OpenAPI spec originally created by the SMEs for field descriptions etc.



	Part of the scrum team	Design document used as reference	Doc begins after dev is complete	Dev create 1st draft that TCs edit	Use the source code	Edit the OpenAPI spec by dev	Information from product demos	Other
Series1	12	12	20	13	21	12	13	6

Figure 1: Source on doc input & content creation process (N=30)

When respondents refer to the source code, we interpret that they use the REST API requests and responses to draft the reference information because 15 respondents also used tools such as Postman to test their content. Respondents who use the API design document or the product demo as a source of information might be using these more for writing the conceptual information. In teams where developers wrote the first draft and TCs edited it, the role of the TC is minimized compared to other teams, but it also means that developers are equally involved in the API documentation in such teams.

> Quote from respondent R17: "*The developers frequently consult me on terminology. I can freely edit comments in the API spec via pull requests. The developers make me feel like one of the team*." (Note: a "pull request" is a Git source control feature that allows someone to make an update and for the maintainer to "pull" that "request" into the code or documentation if approved).

Twelve respondents were part of the scrum team and got regular inputs from the SMEs, but 20 began their documentation only after the development was complete or nearly complete. This points more towards the overall project management philosophy of the company that is, whether the company uses the Agile vs Waterfall project management method. TCs who start documentation after the development is complete either use the product demo or the first draft that the developers create. Eight respondents mentioned that they are not considered a part of the team. Therefore, we can infer that they write documentation only after the development is complete or they work with teams who still follow the Waterfall project management method.

Content testing and review

As shown in Figure 2, content review by SMEs (development and QA) is the most commonly used method by 26 respondents. However, 15 respondents also use a tool such as Postman to test their content or get it peer reviewed by another writer (18). Fewer writers (5) run usability tests on their content.

SME review is the main way documentation is tested. This is in contrast with Q17 - primary challenges faced by TCs, where 11 respondents said that lack of input from SMEs was their biggest challenge to contribute to API documentation. Along with SME review and peer review, use of tools such as Postman is common among TCs. It shows that TCs are not completely dependent on SMEs for their review rather they are also testing the content independently before they publish the content or send the content for SME review.

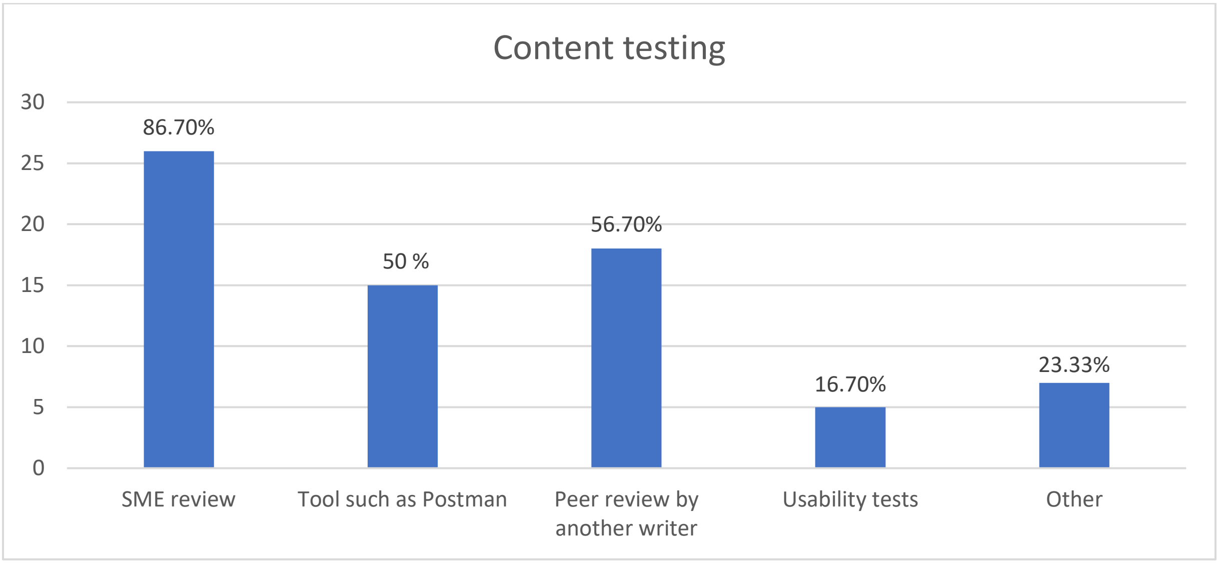
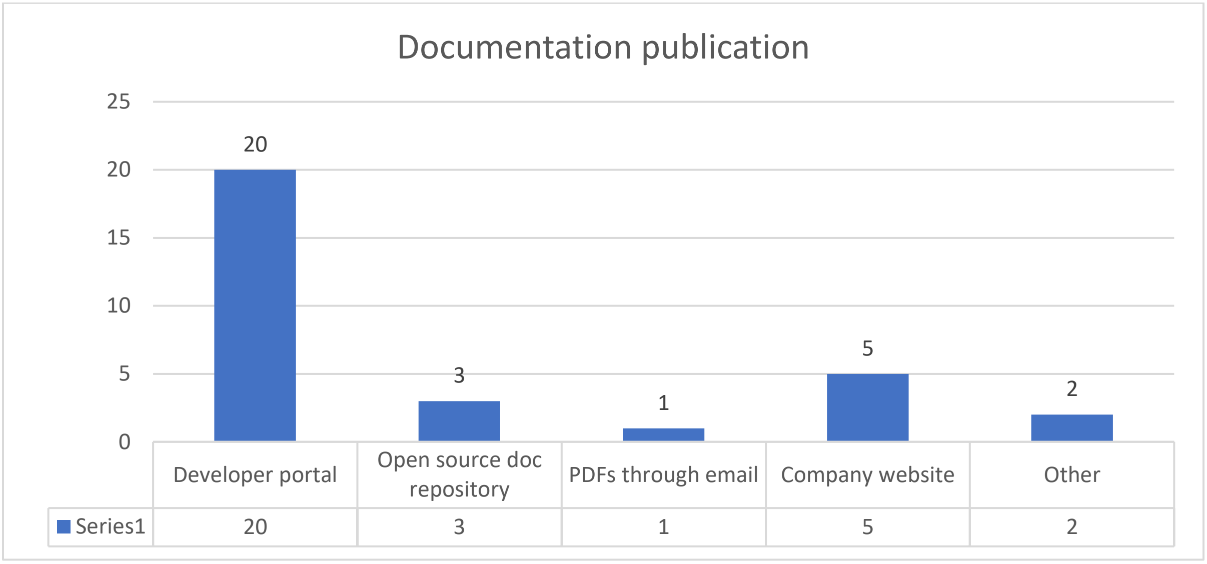


Figure 2: Content testing methods used by respondents (N=30)

Content publication

As shown in Figure 3, 20 respondents have their docs published on a developer portal, whereas five respondents reported that their docs are still published on the company website.

This publishing of docs on a developer portal shows that organizations recognize the value of a developer portal for their API docs and developer experience. Companies that have a developer portal value developer experience; however, how much they value developer experience also depends on the number of sections present in their developer portal as mentioned in Q8 (see Appendix A). For developer portals, TCs are working with several different tools, including static site generators and documentation repositories like Antora, GitHub Pages, GitLab Pages, and hosting platforms such as stoplight.io. This information shows that TCs must be willing to learn and adapt to tools and platforms that require more developer-centric skills like working with the command line and using text editors (as opposed to WYSIWYG editors), and source control applications like Git.

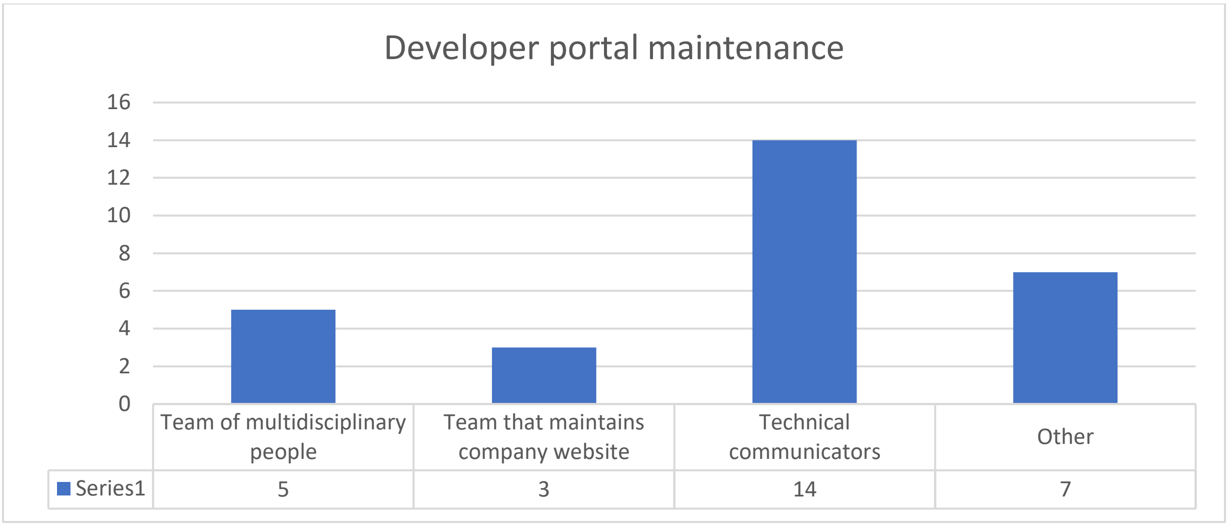


	Developer portal	Open source doc repository	PDFs through email	Company website	Other
■ Series1	20	3	1	5	2

Figure 3: Developer documentation location (N=30)

Content management and maintenance

Fourteen respondents mentioned that the writers maintain the developer portal, whereas five respondents said that their company had a dedicated team to maintain the portal (Figure 4). Three respondents, whose documentation is still housed in the company website, selected the option that the team that maintains the website also maintains the docs. Out of seven respondents who selected others, five of them work with development or other teams to maintain the portal.



	Team of multidisciplinary people	Team that maintains company website	Technical communicators	Other
■ Series1	5	3	14	7

Figure 4: Developer portal management & maintenance (N=30)

TCs are generally on their own for maintaining the developer portals. This independence may be why many also feel that they have the freedom to make any changes to the structure or content if needed (Q16 - how are writers placed to improve the content, see Appendix A). It is an encouraging sign that writers either maintain the developer portal or they are a part of the team that maintains the portal because it can also mean that writers can measure DX periodically and adapt the content and structure based on their findings.

Interaction with other teams

Technical communicators are required to collaborate across the organization in order to create and maintain the content for developer portals and developer experience. This was mainly disclosed in the "Other" section of several survey questions (Q10, Q14, Q16, Q17, Q19, see Appendix A) where 19 TCs discussed how they worked with other teams to maintain and update developer portals. TCs mentioned working with software developers, product management, customer experience, technical support, documentation tools teams, and system integrations teams as a regular part of their job.

Research question 2

Our second research question was “Are technical communicators contributing to other aspects of developer experience?”

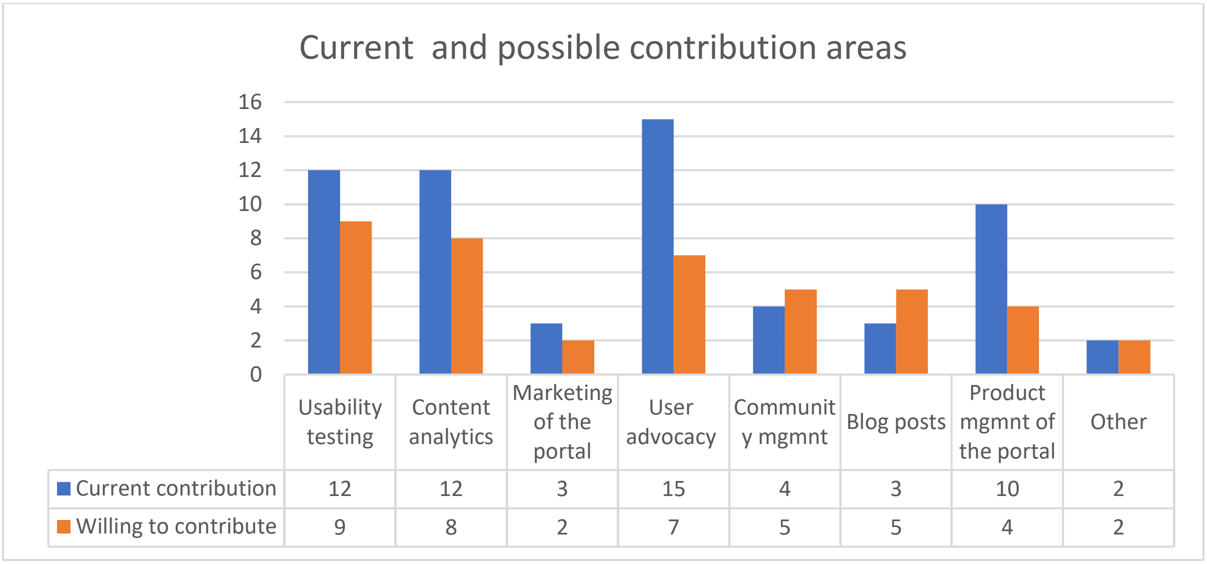
To answer this question, we categorized our findings into following subcategories: current contribution of TCs, current valuation of TCs, and developer experience.

Note that for each chart the numbers (N) may not add to 30 (total number of responses) since respondents were able to select multiple answers or no answer.

Current contribution of TCs

Technical communicators are involved in several other aspects of developer portals. Twelve respondents were contributing to usability testing, 12 to content analytics, and 15 respondents were contributing to user advocacy. Ten respondents were also involved in product management for the portal and three were involved with marketing of the portal. Community management was selected by four respondents and creating portal-related blog posts was mentioned by three respondents.

An interesting observation is that when looking at the responses to the questions of “Which areas of developer portal are you contributing (Q 12)” and “what areas do you want to contribute to (Q 13)” (Figure 5), technical communicators who were contributing in one area, such as user advocacy, but not contributing in another area, such as usability testing, wanted to contribute more to those other areas. This type of response was consistent across most respondents and shows that technical communicators generally try to contribute as much as they can across multiple areas.



	Usability testing	Content analytics	Marketing of the portal	User advocacy	Community mgmnt	Blog posts	Product mgmnt of the portal	Other
🟦 Current contribution	12	12	3	15	4	3	10	2
🟧 Willing to contribute	9	8	2	7	5	5	4	2

Figure 5: Areas where TCs contribute or want to contribute (Current N=27, Willing N=24)

The fact that 12 respondents contribute to content analytics and eight are willing to contribute shows that TCs consider analytics as an important metric to measure the value of their content. The fact that 10 respondents were involved with the product management of the developer portal, and four respondents wanted to add that to their contributions may suggest that TCs are adding or want to add product management skills to their existing skill set.

Only three respondents write blog posts related to their developer portals, and only four respondents said they would like to contribute by writing blog posts. However, only four respondents noted that a blog was available on their portal. The lack of blogs may mean that any blogging is done by another team and posted somewhere other than the portal, and the general lack of interest may mean that TCs see blog writing as a separate skill set from their normal duties and not one they are very interested in.

Current valuation of TCs

Twelve respondents said that TCs are mostly seen as necessary to the success of the API, and seven respondents said that writers are often a critical component to success (Figure 6). These two sets of respondents also noted that they felt they were included in all the team’s decisions.

Another group of seven responded that while their contributions are necessary, they are not considered part of the team. Two respondents felt that their only role was to clean up content generated by the developers. However, the average of the TCs’ valuation for TCs who worked as part of the scrum team (average = 4.09) and TCs who began documentation after development was complete (average = 3.65) was not significantly different.

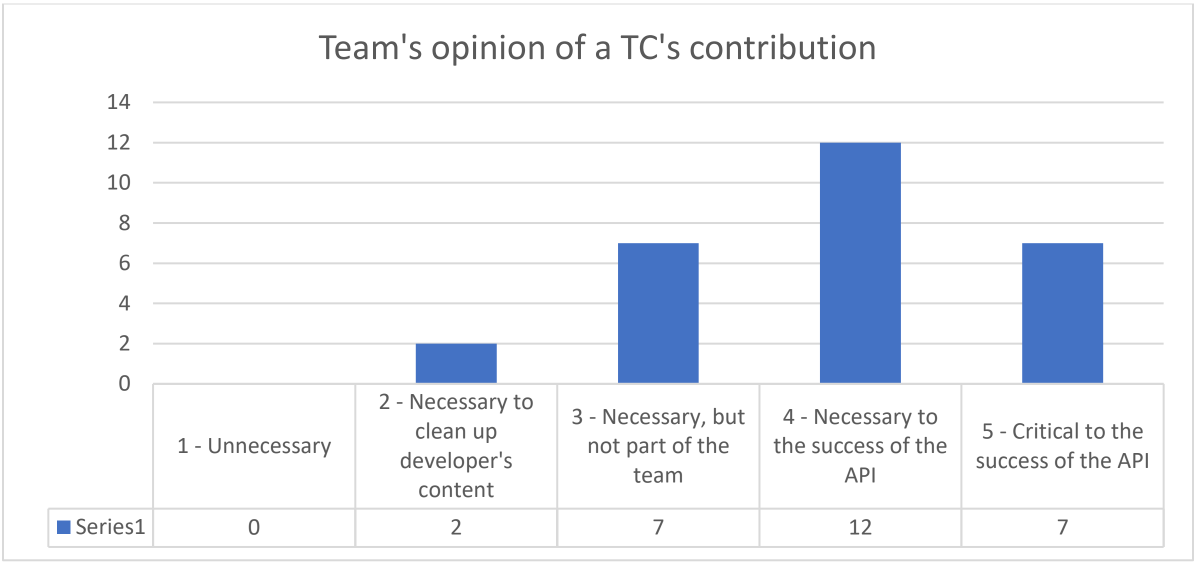


Figure 6: Organization & team's valuation of TC contributions (N=30)

The 19 TCs who saw themselves as both necessary and part of the team also noted that they contributed to more areas of the developer portal than the nine respondents who didn't feel part of the team or only cleaned up developer content. This is an indication that when allowed to fully participate in the API and developer portal process, TCs are willing to take on more roles in order to contribute in multiple areas, while those who are left out of the team are not able to expand their contributions to the level they would prefer.

Fifteen respondents said they were either responsible for or are part of the team responsible for maintaining the developer portal, and it should be noted that a similar number feel they have the complete freedom to make improvements or updates to the developer portal (Figure 7). However, 11 respondents would like to do the work for maintaining the portal, but simply lack the time to take on any additional work, and eight felt they lacked the technical skills required. A smaller group (6) noted that their developer portal has a fixed content structure that they are not allowed to change.

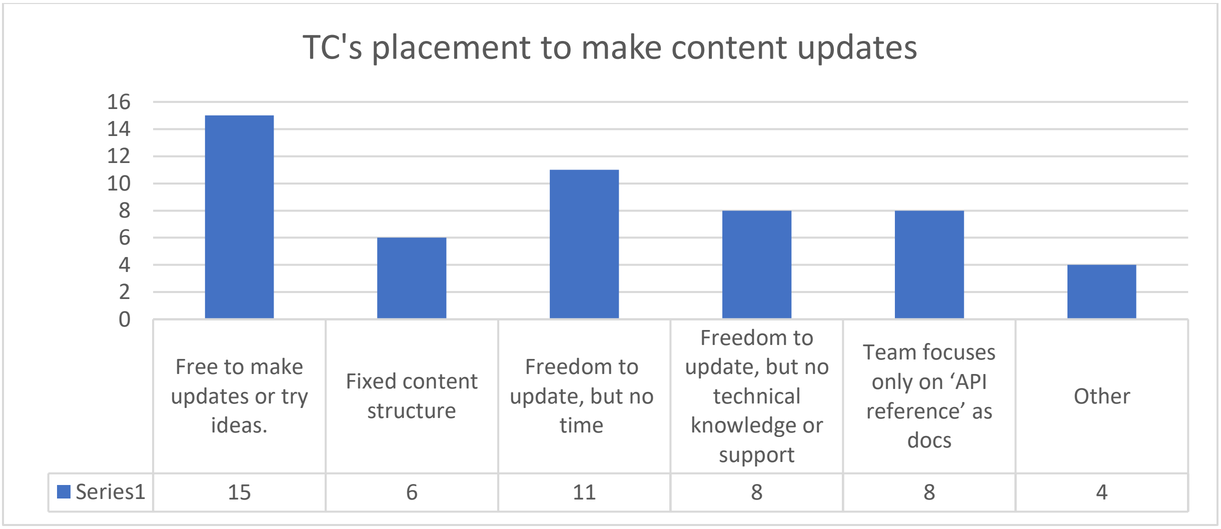


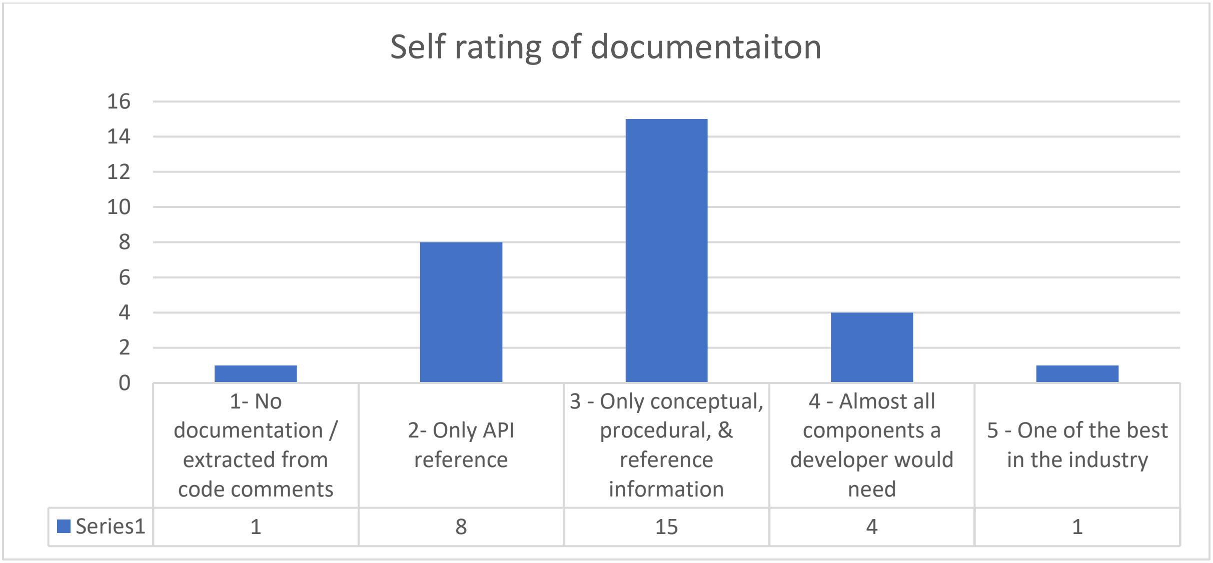
Figure 7: TCs placement to make content updates (N=29)

The fact that several respondents wanted to maintain the portal but felt they lacked the time is echoed by the fact that work overload is an issue reported when asked about other challenges faced by TCs. This could point to a high developer to technical communicator ratio, or the fact that they are working on multiple projects and do not have the time to make updates, even though they are aware that they are needed.

> Quote from respondent R6: "*Lack of time. We've automated our API docs from code & we review/revise new content through PRs*" (Note - a PR is a "pull request").

Developer Experience (DX)

When asked to self-rate the content of their developer portals on a scale of 1 to 5, almost half of the respondents, 12 respondents, gave their documentation a rating of 3, meaning it has conceptual, procedural, and reference information (Figure 8). A smaller but significant number of seven respondents gave their documentation a rating of 2, meaning it only contained the API reference. Only four respondents gave their documentation a rating of 4, meaning it has all, or almost all of the components a developer would need. There was one respondent with a rating of 5, meaning they consider their documentation best-in-class.



	1- No documentation / extracted from code comments	2- Only API reference	3 - Only conceptual, procedural, & reference information	4 - Almost all components a developer would need	5 - One of the best in the industry
■ Series1	1	8	15	4	1

Figure 8: TCs self-rating of the documentation (N=30)

There was no significant difference between the average rating of their documentation by TCs who worked as part of the scrum team (average = 2.9) and TCs who begin documentation after development was complete (average = 2.83).

This contrasts with the responses when asked which sections were present in their developer portals (Figure 9). In that question, many the respondents marked all, or almost all of the available selections as being present in their developer portals: onboarding information, getting started guides, tutorials, error codes, release notes, etc. Most respondents (27) also noted that their developer portals focused on reference information such as API endpoints and parameters, while items like community forums, blog posts, and newsletters were much less common.

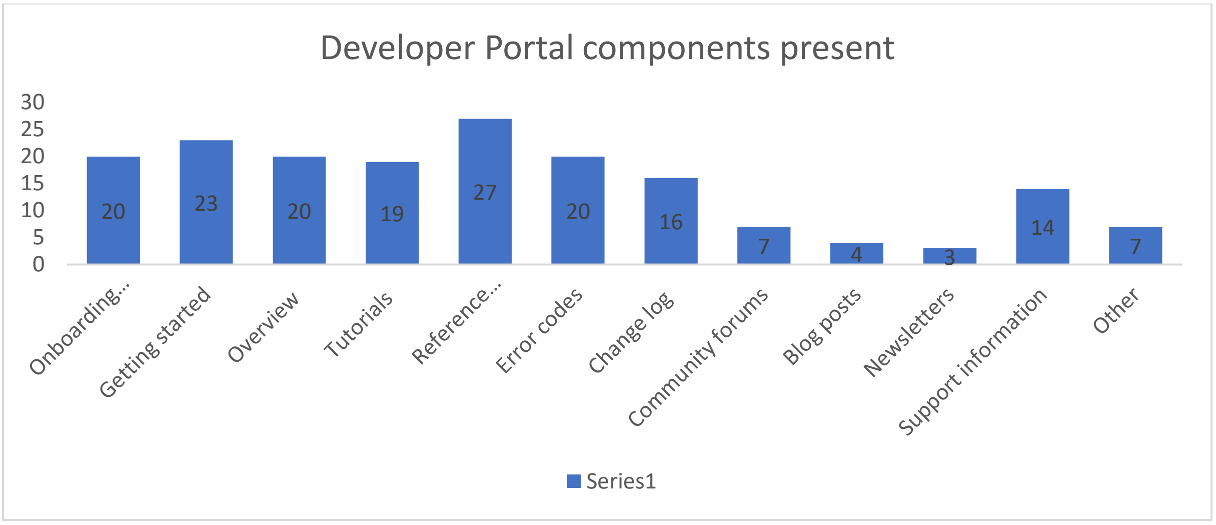
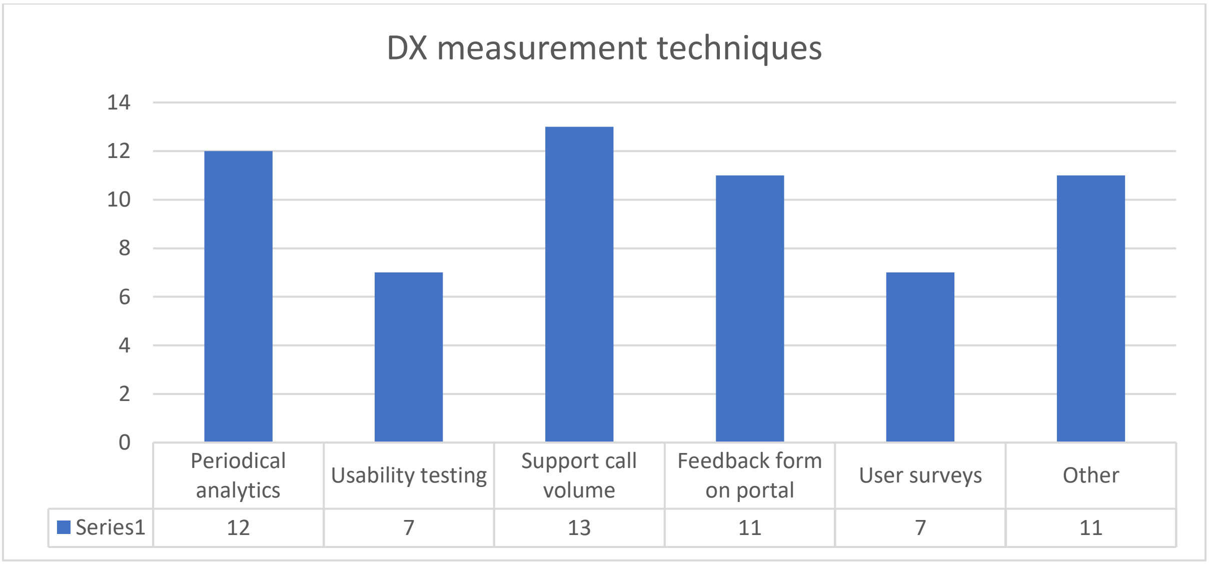


Figure 9: Developer portal components selected per respondents (N=29)

The fact that TCs see their developer portals as having all or many of the sections needed by developers but still only self-rate their portals in the middle of the scale may be due to the fact that while they see that they have all the necessary content available, they do not think it has enough detail or depth to give it a higher rating.

Measuring DX

Organizations are using multiple methods to try and measure their developer experience. A common method reported by 12 respondents is analytics, along with tracking the volume of support calls as indicated by 13 respondents (Figure 10). Having a feedback form on the developer portal is also a common method for measurement, reported by 11 respondents.



	Periodical analytics	Usability testing	Support call volume	Feedback form on portal	User surveys	Other
■ Series1	12	7	13	11	7	11

Figure 10: Developer experience measurement techniques (N=27)

Other common tools are usability testing and user interviews and surveys, reported by seven respondents each. There were only four respondents who did not know how the developer experience was measured. It is interesting to note that while many TCs said that analytics are a major component of how they measure their developer experience, the lack of analytics was also noted as a major challenge that they faced in terms of understanding how their content is being used.

Suggestions for improving DX

There are a number of areas where TCs feel they can make improvements to their developer portals and developer experience, including having more clarity around the authentication and authorization process, better API explorers to show the requests and responses, test environments, code samples in multiple programming languages, and troubleshooting information. In fact, most of the respondents selected either every available answer or almost every available answer (Figure 11).

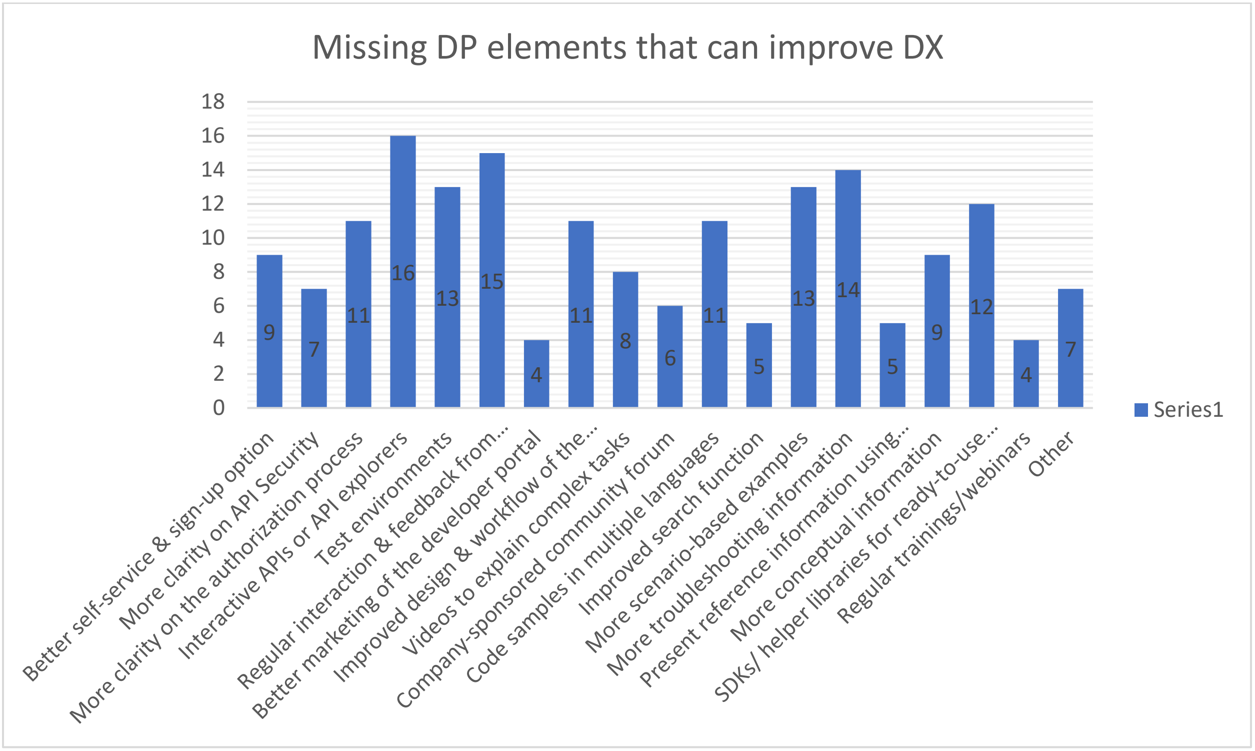


Figure 11: Elements that can improve DX (N=29)

All these parameters indicate that TCs are very interested in providing a robust developer experience and see multiple ways to make their developer portals better and more useful. However, TCs who are overloaded or who do not have the technical knowledge or support need more resources from their organization/teams to implement these ideas.

Eight TCs also reported that their teams only focus on the API reference as the required information. These teams need an awareness of the importance of API documentation as an integral part of the product and how incomplete documentation can distract potential users who visit their portal to evaluate the available APIs.

Research question 3

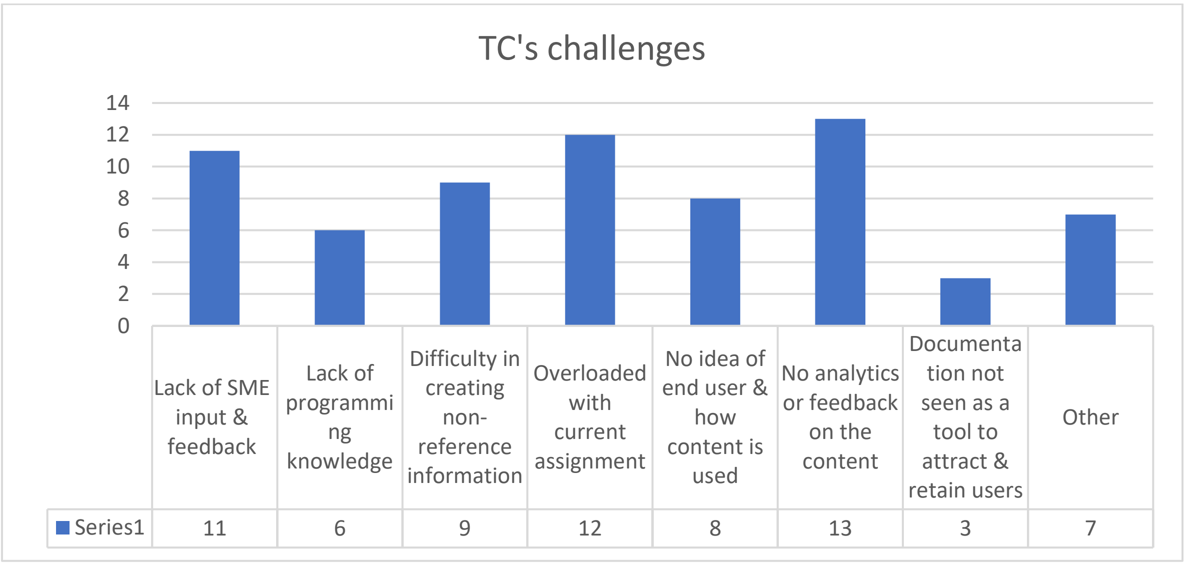
Our third research question was “Can technical communicators do a better job of articulating their influence or usefulness to the developer experience?”

To answer this question, we categorized our findings into following subcategories: Challenges and Opportunities.

Note that for each chart the numbers (N) may not add to 30 (total number of responses) since respondents were able to select multiple answers or no answer.

Challenges

Technical communicators reported facing several challenges related to their work on developer portals and developer experience (Figure 12). The top three were no analytics or feedback on their content as reported by 13 respondents, 12 respondents mentioning work overload, and 11 respondents indicating a lack of SME input and feedback. Lack of SME input and feedback is a serious challenge for API documentation, where the TC may not have the programming or software development skills needed to be able to fully understand the internals of the API or SDK they are documenting.



	Lack of SME input & feedback	Lack of programming knowledge	Difficulty in creating non-reference information	Overloaded with current assignment	No idea of end user & how content is used	No analytics or feedback on the content	Documentation not seen as a tool to attract & retain users	Other
■ Series1	11	6	9	12	8	13	3	7

Figure 12: TCs challenges in contributing to DX (N=27)

Six respondents did say that they felt they lacked the technical knowledge, such as programming skills, to fully integrate into the development teams. Another challenge that was reported by eight respondents is the lack of knowledge of the end user and how their documentation is being used.

Nine respondents cited writing non-reference information such as conceptual information, how different endpoints work together, workflows, etc. as one of their challenges, whereas three respondents feel that their companies do not view documentation as an important tool to attract and retain customers.

TCs who are not included in the scrum teams or who are only working from the developer’s first drafts are more prone to having the challenge of writing all the non-reference information. The lack of SME feedback can greatly compound these challenges.

These challenges show that TCs need to make sure that they are communicating that while they are making contributions, their contributions could be more useful with additional training to improve their technical knowledge and with additional information related to their users.

Opportunities

There is an interesting overlap in where TCs are contributing to the developer portal and experience outside of the API documentation, and where they want to contribute. The top three areas where they are contributing to are usability testing, analytics, and user advocacy, which were noted by nine, eight, and seven respondents respectively.

These are also the areas where TCs want to be contributing. For example, respondents who said they contributed to usability testing but not to analytics noted that analytics was an area where they wanted to contribute, and vice versa. This shows that these skills are viewed as an opportunity for further contribution.

Other areas that writers contribute to are community management (5), blog posts (5), and product management (5).

These opportunities are where TCs should be first communicating their existing contributions, but also making it known that they also want to contribute in other areas to improve the developer experience and portal.

> Quote from respondent R26, who works as a Senior manager of a developer relations education team: “*Get involved with product & engineering early and stay involved*.”

Conclusion

In this section, we summarize our findings, discuss the limitations of the research that constrain the interpretation and application of the findings, and provide suggestions for the future research.

Developer experience is of utmost importance for companies whose product is an API. To sell an API, not only a robust product i.e. an API is important, its documentation plays an equally important role. Decision makers such as project managers, architects or implementors such as developers often first visit the developer portal and test the APIs before deciding on integrating or implementing it. In short, the experience these users get on the portal can make or break the deal for a company. Technical communicators are one of the biggest contributors to the developer portal not only through the API documentation but also by contributing to the other aspects that constitute the developer experience. We conducted this research to study how TCs are contributing to various aspects of developer experience, the processes and challenges they face, to identify areas where they can contribute more, and how they are articulating their contributions.

We conducted a survey with writers who work on developer documentation using the #documenting-apis channel of the ‘Write the Docs’ Slack group. We received and analyzed data of 30 respondents using two different methods: frequency distribution for quantitative data and open coding for qualitative data.

Summary

- **Most respondents worked as technical writers and majority of them worked either as a lone writer or as part of a small team of writers (2-9)**. Majority of our respondents (25) had less than five years of experience in API documentation and 19 of them have spent less than two years in their current organization.
- **The involvement or contribution of TC is largely determined by the importance the organization and the team give to the documentation**. For example, companies that do not value documentation as a selling point for their API involve the TC after the development is complete or treat them as editors for the draft created by developers. In such companies, the focus is more on the 'reference' part of the documentation, which is usually auto generated through the code and cleaned up by the writer. For such writers, writing the non-reference part of the documentation is one of the challenge areas. Whereas companies and teams that value documentation involve the writer as part of the team and provide regular inputs and feedback for documentation. Writers who work with such teams feel more involved in the process and are better aware of the product concepts too. They are also able to test the APIs using tools such as Postman.
- **A developer portal is the most popular publishing destination for most companies**, but there are a few who still publish the docs on their company website in a separate section. Nineteen respondents either manage or are a part of the team that manages their developer portal. This indicates that many writers are already using a docs-as-code tool for authoring and publishing their content. Those who are not using such tools already might need to upgrade their skills when their company either expects the documentation to also follow the docs-as-code approach or if they switch jobs to a different company.
- **Respondents who manage developer portals also have more freedom to make structural updates to the portal as compared to writers who are dependent on others**. However, writers

who are a part of larger teams or work in organizations that have a defined structure find this challenging. Writers who work as lone writers or in smaller teams are challenged by work overload and lack of technical knowledge and support.

- **Respondents listed work overload, lack of input and feedback, no interaction or feedback from users, and lack of analytics on the existing content as the primary challenges they face**. Most respondents identified multiple areas that are currently missing in their portal and can help to improve the developer experience. However, they also face challenges that stop them from contributing to these areas.
- **TCs not only work on API documentation, but many are also contributing to related areas** such as user advocacy and usability testing. A significant number of respondents also contribute to content analytics and product management of the portal. These along with blog posts, community management, and developer relations are some avenues where TCs contribute to or are willing to contribute. It means that TCs need additional skill sets to contribute to developer experience whether it is UX, analytics, community building, marketing, or product management. TCs must also ensure that they are making sure that their contributions are known by the rest of the developer portal and developer experience teams.

Research Limitations

- The reach of the study was limited to the number of TCs who are part of the #documenting-apis channel on the WtD Slack workspace and were willing to reply to the survey in the allotted time.
- Our research had only 30 respondents for a population of approximately 1500 API writers (2%), so the findings of this research paper cannot be generalized to the population.
- Respondents were not a heterogeneous representation of the population because 13 respondents (43%) of the total respondents (30) had 0-2 years of experience in API documentation.
- Many respondents chose to not answer the open-ended questions which could have provided additional insights. Due to time constraints, the researchers could not triangulate the answers using any other research method.
- Total duration of the research was 7-8 weeks approximately which was not enough to collect and analyze data.
- Time constraint of two weeks allotted for data collection did not allow the researchers to triangulate the data using other methods or collect more feedback on responses that needed more elaboration.

Suggestions for future research

- With perspective to the above limitations, we have following suggestions for any future research on this topic:
- Get responses from at least 100 writers. The respondents must be a good mix of writers with different experience levels, writers with programming vs non programming backgrounds, writers who work as lone writer's vs writers who work in larger teams.
- Researchers must interview writers to get more details other than the questions we covered through the survey.
- Get a perspective of product managers and developers of how they perceive the role of TCs in DX.

Acknowledgements

We would like to thank Dr. Brewer, our research guide for TCO 685 course and Professor, MSTCM program at Mercer University for her support and guidance throughout the research project. Additionally, we would also like to thank our pilot respondents and our survey respondents for their time and valuable inputs. Finally, we would like to thank our family members for their constant support during the research project duration.

References

7 Mantras For Quality Developer Experience | Nordic APIs |. (2019, November 5). Nordic APIs. https://nordicapis.com/7-mantras-for-quality-developer-experience/

API Developer Experience (DX) Resources. (2020, May 17). API Developer Experience (DX) Resources | API Guide. https://www.moesif.com/blog/api-guide/api-developer-experience/

Eyolfson, J., Tan, L., & Lam, P. (2011). Do time of day and developer experience affect commit bugginess? *Proceedings of the 8th Working Conference on Mining Software Repositories*, 153–162. https://doi.org/10.1145/1985441.1985464

Fagerholm, F., & Münch, J. (2012). Developer experience: Concept and definition. *Proceedings of the International Conference on Software and System Process*, 73–77.

Fontão, A., Bonifácio, B., Santos, R. P. dos, & Dias-Neto, A. C. (2018). Mobile Application Development Training in Mobile Software Ecosystem: Investigating the Developer eXperience. *Proceedings of the 17th Brazilian Symposium on Software Quality*, 160–169. https://doi.org/10.1145/3275245.3275262

Gentle, A., Fleming, D., & Holcomb, K. (2017). *Docs like code: Write, review, test, merge, build, deploy, repeat*.

Johnson, T. (2020, March 30). *Are technical writers increasingly playing non-technical roles? Some thoughts on the evolution of technical writing roles*. I'd Rather Be Writing. https://idratherbewriting.com/blog/evolving-roles-of-technical-wrters/

Kuusinen, K. (2016). *Are Software Developers Just Users of Development Tools? Assessing Developer Experience of a Graphical User Interface Designer*. *LNCS-9856*, 215–233. https://doi.org/10.1007/978-3-319-44902-9_14

Latorre, R. (2014). Effects of Developer Experience on Learning and Applying Unit Test-Driven Development. *IEEE Transactions on Software Engineering*, *40*(4), 381–395. https://doi.org/10.1109/TSE.2013.2295827

Meng, M., Steinhardt, S., & Schubert, A. (2018). Application Programming Interface Documentation: What Do Software Developers Want? *Journal of Technical Writing and Communication*, *48*(3), 295–330. https://doi.org/10.1177/0047281617721853

Myllärniemi, V., Kujala, S., Raatikainen, M., & Sevońn, P. (2018). Development as a journey: Factors supporting the adoption and use of software frameworks. *Journal of Software Engineering Research and Development*, *6*(1), 6. https://doi.org/10.1186/s40411-018-0050-8

Robillard, M. P. (2009). What Makes APIs Hard to Learn? Answers from Developers. *IEEE Software*, *26*(6), 27–34. https://doi.org/10.1109/MS.2009.193

Robillard, M. P., & DeLine, R. (2011). A field study of API learning obstacles. *Empirical Software Engineering*, *16*(6), 703–732. https://doi.org/10.1007/s10664-010-9150-8

Santos, W. (2019, November 19). *Best Practices: How to Engage Developers with a World-Class API Portal*. ProgrammableWeb. https://www.programmableweb.com/news/best-practices-how-to-engage-developers-world-class-api-portal/how-to/2019/11/19

Society for Technical Communication. (2014). *Intercom*. *61*(8), 40.

van Tomme, K. (2018, March 29). *Developer Portals: When Docs become DX* [Text]. Pronovix. https://pronovix.com/blog/developer-portals-when-docs-become-dx

Watson, R. B. (2012). Development and application of a heuristic to assess trends in API documentation. *Proceedings of the 30th ACM International Conference on Design of Communication - SIGDOC ’12*, 295. https://doi.org/10.1145/2379057.2379112

Appendix A

In this appendix, we have listed only the survey questions and the pre-defined options we provided in the survey for some questions.

Screener question: Do you work on developer documentation?

Q 1. What is your primary role in your current organization?

Q 2. How many years of experience do you have in API documentation?

Q 3. How long have you been working with this company?

Q 4. How many writers does your company employ and how many of the total are API writers?

Q 5. What is the content creation process in your company?

- TCs are part of the scrum team and get inputs from developers
- API design document used as reference for development and documentation
- Documentation begins after development is complete
- Developers create first draft, TCs edit the draft, and add more information
- TCs use the source code, document information, and get it reviewed by SMEs
- TCs edit the OpenAPI spec by dev to add more information
- Information from product demos
- Other

Q 6. How do you test your content?

- SME review
- Tool such as Postman
- Peer review by another writer
- Usability tests
- Other

Q 7. Where is the API documentation published?

- Developer portal
- Open source doc repository
- PDFs through email
- Company website
- Other

Q 8. Which of these sections are present in your developer portal?

- Onboarding information
- Getting started
- Overview
- Tutorials
- Reference information
- Error codes

- Change log
- Community forums
- Blog posts
- Newsletters
- Support information
- Other

Q 9. On a scale of 1 - 5, how would you rate your current documentation?

1. No documentation / extracted from code comments
2. Only API reference
3. Only conceptual, procedural, & reference information
4. Almost all components a developer would possibly need
5. Labeled as one of the best in the market

- Other

Q 10. Who maintains the developer portal in your company?

- Team of multidisciplinary people such as developers, QA, UX, TCs, PM
- Same team that maintains the company website
- Technical communicators
- Other

Q 11. On a scale of 1 - 5, how do you think your organization values your contributions?

1. Unnecessary
2. Necessary but ancillary; Clean up developer-generated content
3. Necessary, but not considered part of the team
4. Necessary to the success of the API
5. Critical component to the success of the API

- Other

Q 12. A developer portal houses much more than API documentation. Which other areas do you contribute to?

- Usability testing
- Content analytics
- Marketing of the portal
- User advocacy
- Community management
- Write blog posts
- Product management of the portal
- Other

Q 13. In continuation to the above question, to which areas do you want to contribute that you don't already?

- Usability testing

- Content analytics
- Marketing of the portal
- User advocacy
- Community management
- Write blog posts
- Product management of the portal
- Other

Q 14. How does your company measure developer experience?

- Periodical analytics
- Usability testing
- Support call volume
- Response in the feedback form on the developer portal
- User surveys in marketing forums and conferences
- Other

Q 15. Which of these ideas can help your company to improve developer experience?

- Better self-service & sign-up option
- More clarity on API Security
- More clarity on the authorization process
- Interactive APIs or API explorers
- Test environments
- Regular interaction & feedback from users
- Better marketing of the developer portal
- Improved design & workflow of the developer portal
- Videos to explain complex tasks
- Company-sponsored community forum
- Code samples in multiple languages
- Improved search function
- More scenario-based examples
- More troubleshooting information
- Present reference information using three-column layout
- More conceptual information
- SDKs/ helper libraries for ready-to-use information
- Regular trainings/webinars
- Other

Q 16. If you think that your content structure needs improvement or there are new ideas that can help to improve the developer experience, how are you placed to make those changes?

- Free to make structural updates or try new ideas.
- Fixed content structure for all products
- Freedom to update, but busy to take up more work
- Freedom to update, but no technical knowledge or support

- Team focuses only on ‘API reference’ as docs
- Other

Q 17. What are your primary challenges in your contribution to developer experience?

- Lack of SME input and feedback
- Lack of programming knowledge
- Difficulty in creating non-reference information
- Overloaded with current assignment
- No idea of end user & how content is used
- No analytics or feedback on the content
- Documentation not seen as a tool to attract & retain users
- Other

Q 18. Do you have an API documentation website that you refer to? If so, what is it? Why did you choose it?

Q 19. Would you like to add any other information that is relevant to the topic of technical communicator's influence on developer's experience?

Index: