Understanding Documentation Use Through Log Analysis

An Exploratory Case Study of Four Cloud Services

Daye Nam
Carnegie Mellon University
Pittsburgh, U.S.A.
dayen@cs.cmu.edu

Andrew Macvean
Google, Inc.
Seattle, U.S.A.
amacvean@google.com

Brad Myers
Carnegie Mellon University
Pittsburgh, U.S.A.
bam@cs.cmu.edu

Bogdan Vasilescu
Carnegie Mellon University
Pittsburgh, U.S.A.
vasilescu@cmu.edu

ABSTRACT

Almost no modern software system is written from scratch, and developers are required to effectively learn to use third-party libraries and software services. Thus, many practitioners and researchers have looked for ways to create effective documentation that supports developers' learning. However, few efforts have focused on how people actually use the documentation. In this paper, we report on an exploratory, multi-phase, mixed methods empirical study of documentation page-view logs from four cloud-based industrial services. By analyzing page-view logs for over 100,000 users, we find diverse patterns of documentation page visits. Moreover, we show statistically that which documentation pages people visit often correlates with user characteristics such as past experience with the specific product, on the one hand, and with future adoption of the API on the other hand. We discuss the implications of these results on documentation design and propose documentation page-view log analysis as a feasible technique for design audits of documentation, from ones written for software developers to ones designed to support end users (e.g., Adobe Photoshop).

CCS CONCEPTS

• **Software and its engineering** → **Documentation**; • **Human-centered computing** → **Empirical studies in HCI**.

KEYWORDS

Documentation, Log analysis, Empirical study, Design review

ACM Reference Format:
Daye Nam, Andrew Macvean, Brad Myers, and Bogdan Vasilescu. 2024. Understanding Documentation Use Through Log Analysis: An Exploratory Case Study of Four Cloud Services. In *Proceedings of the CHI Conference on Human Factors in Computing Systems (CHI '24), May 11–16, 2024, Honolulu, HI, USA.* ACM, New York, NY, USA, 17 pages. https://doi.org/10.1145/3613904.3642721

CHI '24, May 11–16, 2024, Honolulu, HI, USA

ACM ISBN 979-8-4007-0330-0/24/05
https://doi.org/10.1145/3613904.3642721

1 INTRODUCTION

Almost no modern software system is written from scratch, and many third-party libraries and software services are available to be reused and composed. Thus, the productivity of programmers in many domains and contexts depends on rapidly searching for relevant information to make decisions about third-party libraries or services [51, 59], and learning to use them correctly for their own systems [66, 67]. Practitioners spend a lot of time searching for and digesting relevant API information, e.g., 20% of their time according to Brandt et al. [10]. And while many sources are useful, including code examples, question and answer (Q&A) websites, and expert advice, in obtaining API-relevant information, the official software documentation remains essential [13, 26, 67].

Efforts to improve software documentation span decades, with many researchers studying documentation design experts and users to catalogue problems [13, 66, 67] and recommend best practices [66, 67, 83]. Much documentation now follows such guidelines, and new tools [46, 80] and ideas [68] have been proposed to further support developers' information needs based on such studies.

Most of these efforts involve qualitative research methods such as interviews [47, 57, 67] or lab studies with human participants [20, 29, 31, 49]. However, while generally highly informative for understanding usability issues during the early design review phase [18], such methods capture only what participants say they do, or what they do in a controlled setting. Moreover, the number of participants that can be observed this way is typically small.

Our research goal is similar to most prior software documentation research—improving the design and usability of documentation. However, our approach is novel and complementary—mining documentation page-view logs at scale. Web mining has long been used to analyze people's experience online in more general contexts, e.g., user engagement in online news reading [37] or user satisfaction during online shopping [75]. We argue that similar approaches could apply to documentation since, after all, documentation webpages are just another type of webpage. In other words, any software documentation published on the Web, such as Adobe Photoshop and Autodesk AutoCAD, which comprise numerous documentation webpages and users, could potentially be analyzed using an approach similar to ours.

We believe that our large-scale log analysis will complement existing documentation review methods, by providing the following additional methodological advantages:

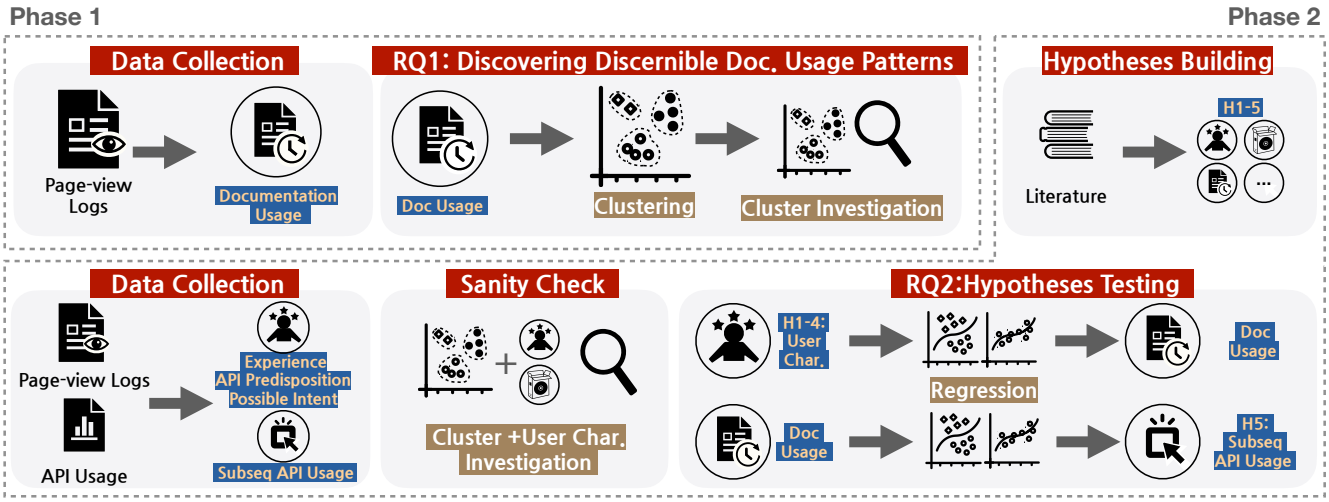

Figure 1: Overview of our data collection and analysis.

- **Allowing more scalable, computational design review**: Relying on web analytics to understand documentation usage is considerably less expensive for software providers if they have access to telemetry data for the documentation pages (e.g., from self-hosted web servers)—we expect that one quantitative user experience researcher on staff could analyze the page-view logs of hundreds of thousands of users of dozens of APIs or services in a matter of days, if not hours, following our methodology. In contrast, qualitative studies tend to focus on one API or service at a time, may require complex participant recruitment, and usually involve orders of magnitude fewer subjects. They also often involve monetary compensation (e.g., Duala-Ekoko and Robillard [20] compensated each participant with $20 for a one-hour programming study), in addition to the researcher team's time for running the studies, and collecting and analyzing the data.
- **Allowing for the discovery of less-studied documentation user groups**: As most of the smaller scale studies require study design prior to the data collection, researchers specify research questions and target participants in advance. For documentation, as it is expensive to conduct these qualitative studies, most of the studies have focused on the professional developers who use the documentation for API learning [21, 35, 47, 48], the main target usage scenario of software documentation. Large-scale log analysis, on the other hand, does contain the entire user population's data, allowing the discovery of more diverse user groups, including users who use documentation to make API adoption decisions (e.g., Product explorers in Section 4.3), or users who are only concerned with the cost of querying APIs (Financial users in Section 4.3).
- **Capturing a perspective less prone to response biases:** With qualitative studies where users need to report (e.g., survey, interview) or show their behaviors (e.g., observation study, lab study), the data can only capture what participants recall or show, which might be different from what actually happens in the wild, i.e., response bias [45]. As qualitative studies often ask participants to focus on "software documentation regularly used by participants" [13, 23, 62] to help participants recall specifics of their experience, the bias might be even amplified. Log analysis can minimize the response bias, as the telemetry data is automatically collected.

However, to be clear, traditional non logs-based approaches can be extremely valuable, and we don't advocate replacing them. Instead, we argue that a logs-based analysis like ours could be used as a first pass, to guide the design of more complex (but rich in terms of insights) approaches such as human studies.

To this end, in this paper we report on an exploratory, two-phase, mixed-methods empirical study of documentation page-view logs from over 100,000 users of four popular services of Google; see Figure 1 for an overview. The documentation page-view logs we had access to were privacy-preserving in a number of ways (section 3.3) and contained only aggregated monthly totals of which specific documentation pages someone visited and how much time they spent on each page, over the course of that month (possibly across multiple sessions). This is likely a common scenario — many companies and open-source projects can be in a position to instrument their documentation web servers to collect such basic telemetry data; at the same time, it may be undesirable to collect more fine-grained or personally identifiable data for privacy reasons. The research challenge, therefore, is determining whether there is enough signal in this big but shallow data to generate actionable insights for the documentation designers by mining it.

Overall, our two-phase study argues that the answer is "yes." In Phase I (section 4) we set out to explore the log data, looking for patterns of page views and trying to explain them *without* knowing who the users are or anything else about them. Given the large size of our sample, we do this using a combination of automatic

clustering analysis followed by qualitative explorations and show that many page-view clusters are discernible in the log data.

In Phase II (section 5) we set out to formalize and generalize our qualitative observations from earlier. In an effort to understand *why* such discernible patterns exist in the page-view data, we formulate testable hypotheses about the "average" characteristics and subsequent behavior of those users, based on findings from the literature on general information seeking of developers [10, 25, 34, 39, 43, 64, 73, 77, 89] and from small-scale documentation usability studies [20, 29, 31, 35, 48, 49]. We then use the fact that all users who make requests to Google services, or had otherwise registered for accounts on Google and were browsing the documentation pages while logged in, have persistent (pseudonymized) IDs across the data. This way, we join the page-view log data with user-level data about their experience with the respective service and the platform overall, and with data about subsequent requests (after the documentation page views) to the service APIs. We first revisit the clustering results and check if the hypotheses built based on the general information seeking literature make sense in the documentation usage setting. We then conduct multiple regression analysis to test the hypotheses formulated in the first phase on this aggregated data, finding multiple sizeable correlations between patterns of documentation page-views, on the one hand, and user-level characteristics and subsequent API use, on the other hand. That is, one's level of experience partially explains one's documentation browsing patterns, and one's documentation browsing patterns partially explain one's intent to subsequently use the APIs.

While not intended as an exhaustive exploration of all patterns of documentation page views identifiable for the four Google services in our sample, our study does show that it is feasible to analyze page-view logs at scale to inform documentation design reviews, or to corroborate observations from smaller-scale studies [17, 47] or the anecdotal experiences of professional software engineers. Concretely, we argue (section 6) that even when not knowing anything else about the documentation users, the interaction histories and dwell times that are likely to be contained in the page-view logs can provide actionable information at scale for providers which can help companies decide which documentation pages to redesign, and even to potentially automatically personalize documentation pages in the future, to better align with their users' needs.

2 RELATED WORK

Studies on Software Documentation. There have been many investigations to improve software documentation, e.g., cataloging problems [13, 66, 67], identifying desirable quality attributes [9, 19], and recommending best practices [66, 67, 83]. Some of these studies provided concrete insights into what developers need from the documentation. For example, developers have expressed the need for complete and up-to-date documentation [6], because many developers rely on API reference information and code examples [47, 57] when they approach documentation with a problem or task in mind [47]. Developers also asked for a concise overview of the documentation, more rationale, and adequate explanation for code examples [47, 66, 67, 81]. Researchers have also proposed tools that can assist in more effective usage of documentation, by providing easier access to the documentation contents within developers' workflow [27, 30, 58].

However, most of these studies focused on the documentation artifacts, not the user context. Still, we know from the literature that the documentation users' needs tend to vary with experience [21, 36, 38, 40, 44, 63], roles, and learning styles [17, 21, 40, 48]. For example, Costa et al. [17] found that documentation users with less experience with the software tended to use more types of documentation than more experienced users, and that tutorials and how-to videos were used by a greater percentage of newer users, and the newer users tended to use tech notes and forums less. Similarly, in the literature, programmers are sometimes categorized into three personas, which summarize their information seeking and problem solving strategies – systematic, opportunistic, and pragmatic [15] – that reportedly also correlate with documentation use [48]. For example, opportunistic developers tended to use documentation in a task-oriented way, focusing less on the general overview of APIs or the suggestions described in the documentation; in contrast, systematic developers tried to understand how the API works before diving into the details of a task, by systematically searching and regularly consulting documentation provided by the API supplier [48]. In our study, we provide evidence that these different user characteristics and information needs indeed correlate with their documentation usage, underscoring the importance of accounting for distinct user groups in documentation design.

In terms of research methods, most of the past studies on software documentation relied on interviews [47, 57, 67], surveys [21, 47, 66, 67], observation studies [35, 48], and lab studies [20, 29, 31, 49] that usually involve a small number of participants. Our work stands out in that we do not rely on self-reported data, nor lab studies, but rather automatically collected logs of actual documentation page views, thereby offering a complementary approach to studying software documentation and documentation users, based on real-world telemetry data in an industrial context.

Studies on Developer Information Foraging. To learn to use, or reuse, new software frameworks or libraries, developers need a variety of kinds of knowledge [34, 43, 73, 77, 89], so it is important to understand how they search for and acquire information. There is some prior work on the information seeking strategies of developers, but mostly in general software maintenance [25, 34, 39] or web search settings [10, 64] rather than learning. For example, prior work [10, 64] found that developers' web search behaviors vary with their information seeking intent: they visit different types of web pages, use different queries, and overall interact with webpages differently. In particular, developers were more likely to visit official documentation during reminding sessions, versus third-party tutorials during learning sessions [10].

More recently, with the advent of large language models, developers have embraced generative models as alternatives to conventional information retrieval from existing sources [22, 41]. Still, researchers have found that the strategies employed by developers to generate necessary information can vary based on factors such as their intent, programming experience, and familiarity with AI tools [55, 70, 88].

Document Design. Documentation in fields beyond software development has a richer history [71]. Researchers have dedicated

Table 1: Types of documentation provided for the selected products.

Genre	Type	Description
Meta	Landing (L)	Links to core documentation pages.
	Marketing (M)	A brief introduction to a product, incl. the benefits, target users, and highlights current customers.
Guide	Tutorial (T)	Walkthroughs for common usage scenarios.
	How-to (H)	Guidance on completing specific tasks.
	Quickstart (Q)	A quick intro to using the product.
	Concept (C)	Explanations for product- or domain-specific concepts.
Dev	Reference (Ref)	Details about the API elements, including API endpoints and code-level details.
	Release note (Rn)	Specific changes included in a new version.
Admin	Pricing (P)	Pricing information.
	Legal (Lg)	Legal agreement details.
	Other (O)	Other resources not included in other types, e.g., locations of the servers.

their efforts to enhancing document design by delving into audience analysis [7], refining content based on user feedback [11, 14], and evaluating the documentation [8, 52]. However, many of these endeavors were primarily geared towards relatively simpler products like educational brochures or games, which may not fully align with the large-scale software systems with more than hundreds of documentation pages that we analyzed in this work.

Studies on Web Usage Mining. To improve the usability of web content, extensive research has been conducted in web usage mining [74]. By analyzing logs stored in web servers using data mining techniques, it is possible to identify interesting usage patterns, identify different navigational behaviors, and discover potential correlations between Web pages and user groups. Among the different types of data available in usage logs, page dwell time has been the primary source of understanding users' needs and intentions, along with the search queries [33, 85, 86]. Page dwell time has also been found to correlate with document relevance and user satisfaction [12, 16, 24]. To the best of our knowledge, our work is the first large-scale study of developers' dwell time on documentation pages, showing the feasibility of applying similar approaches in analyzing documentation users' needs.

3 DATASET

We started by compiling a dataset of documentation page-view logs for four web-based services of Google.

3.1 Product Selection

Google provides hundreds of web-based services to a diverse group of users and businesses, and most of the services come with one or more types of APIs, including REST APIs and gRPC APIs, as well as client libraries. However, since our study is primarily exploratory in nature, we selected only four Google products following a maximum variation sampling strategy [76], to gain an understanding of documentation use from a variety of angles. Concretely, we diversified our sample in terms of the application domain (machine learning / natural language processing vs. event analytics and management), usage context (operations infrastructure vs. potentially end-user facing), and product size and complexity (ranging from a few API methods to hundreds of API methods offered by the products). These differences are also reflected in the documentation pages, which vary in their contents across the four products, e.g., with more or less marketing materials, how-to guides, pricing information, *etc.* All four web-based services we selected are popular, having large user bases. Specifically, P1 and P2 are machine learning / natural language processing-related products for machine translation and text analysis. And P3 and P4 are operations-related products for managing event streams and log data.

3.2 Documentation Usage Data Preprocessing

For each of the four products, we had access to pseudonymized **documentation page-view logs** [42] for users who visited the documentation from May 1, 2020 to May 31, 2020, UTC, while they were logged into their accounts. The page-view log data are collected automatically by the documentation servers and include the specific documentation pages visited by someone, as well as the timestamps and dwell times for each visit. We use *dwell time* to estimate user engagement with the content, following prior work [24, 86]. The data was aggregated at the month level, partially due to the volume of data being analyzed, and also to enhance the pseudonymization of the data for the privacy protection of the users (see Section 3.3 for details).

To reason about more general patterns of documentation use, we further labeled each individual documentation page (URL) in our sample according to its contents into one of 11 possible *types* and four aggregate categories (or documentation *genres* [21]) summarized in Table 1. For the first-level labeling we relied on an internal mapping table created by the documentation team, which contains meta information for the different documentation pages, including what we refer to as the *type*. The second-level labeling reflects our subjective grouping of documentation types into four high-level categories that provide related kinds of information and presentation format; we expect that these are likely to be consulted together given specific tasks and target reader familiarity with the API. To this end, we followed an open card sorting process involving two authors, one of whom is a domain expert. There were two documentation types (Other and Release note) that the two authors did not

agree on. The two authors resolved disagreements by comparing their definitions of genres and the rationale behind the categorization, until there were no more disagreements. All official Google documentation pages in our sample were assigned to exactly one of the documentation types and genres listed in Table 1, and the documentation of all four products we analyzed included all 11 documentation types. The volumes of each documentation type varied, but in general, each product documentation consisted of around 5 pages of Meta genre documentation, around 150 pages of Guide genre documentation, around 300 pages of Dev genre documentation, and around 15 pages of Meta genre documentation. The documentation of all products followed the same documentation style guide [1]; thus, the contents and styles used for each documentation type are consistent, even across the products.

Finally, we followed prior work by Fox et al. [24] and excluded page-view sessions shorter than 30 seconds, since these are more likely to be noise (e.g., a user accidentally clicked the documentation page and then left the page quickly) than meaningful visits.

3.3 Privacy Protections

As we analyzed the user data of Google, we followed Google's strict privacy and data access policies [2, 3] which ensure appropriate, legal, and ethical access, storing, and analysis of user data. This included, but was not limited to, internal privacy reviews with security and privacy experts, the use of differential privacy processes (more details below), wipeout and data access processes, and more. In addition, the study designs were reviewed by internal research ethics experts, methodological experts, and product experts.

We also used numerous privacy protection techniques. First, all user-level data was pseudonymized before any of the authors had access to it. Pseudonymization maps users' accounts to randomly but persistently generated pseudonymized IDs. As the IDs were randomly generated, they could not be reversed without access to a mapping table, which the authors did not have.

Additionally, usage data was aggregated (e.g., we looked only at the number of API requests aggregated at the month granularity, not individual API requests), and had Differential Privacy [84] applied. In brief, differential privacy was used to apply sufficient noise to the aggregations such that individual records could not be identified, but the overall shape of the dataset remained meaningful / sufficiently accurate. Using established best practices, and based on guidance with internal privacy experts, we used Epsilon <1.1, (where lower numbers yield higher degrees of privacy protection). This allowed us to analyze trends in user behavior while preserving the privacy of those in the dataset. Later in this paper (e.g., in Figure 3), we include polar plots of our clusters, but choose only to visualize clusters with over 500 users, as an additional privacy consideration.

3.4 Limitations & Threats to Validity

First, a month might not be enough to capture the full process of learning and adopting a complex API, on the one hand, and might not capture the differences in documentation usage patterns that appear in significantly shorter periods, such as patterns in an hour or in a day, on the other hand. It is also possible that some users happen to register in the middle of our one-month window, or one may learn an API intermittently over a few months. However, such an operationalization was necessary to balance data collection complexity, privacy, and analysis scale. Given that the size of Google's general user base is very large, and the services we analyzed were already all mature, we believe that our dataset should still capture snapshots of developers at every stage of the learning process, as well as cyclical patterns of use, without the number of newly registered or intermittent users significantly affecting our results.

The use of a particular month (May 2020) can also be too short to generalize, as the documentation access might change throughout the year, and it could have been influenced by any major event related to the four target products. We did our best to choose a month without major events related to the four products we studied, and there was no event for P1 and P2, but there were two minor feature additions for P3 and two minor beta releases for P4. However, as we chose to analyze popular products with large active user groups, it is practically not possible to choose a month without any updates. We believe that selecting four very different products reduces the risk of biasing the results in a meaningful way, especially given that there was no major event that affected all of the four products during that period.

The documentation and API usage data we used for our analysis can only provide a partial representation of the entire user group's usage. Since the documentation usage data only include page-view logs of logged-in users, the analysis does not capture the behavior of users who were not logged in, who may behave differently. In addition, the aggregated API usage data can only partially represent the outcome of API learning. For example, while making an API request requires a user to sign into the platform, browsing documentation does not, so not all documentation usage is linked to the corresponding API use. Multiple developers can make API requests using shared corporate accounts, which can obfuscate the connection between their documentation and API use.

Although the dwell time was logged when the pages were actually accessed, our measurements of time spent on each documentation page are only (over)approximations. For example, some users may keep a page open without actively consuming it the whole time, while they grab a coffee or read code from their IDE. As part of our analysis, we applied several heuristics and filters to our data to identify and remove outliers and noise, as described in section 4.1.

The analyses at the documentation type and genre levels introduce threats to internal validity: the analyses might not capture the possible influence of content and length of individual documentation pages, and other external confounding factors. However, the abstraction of data was inevitable due to the number of documentation pages available. We provide potential ways of introducing additional internal validity control for page-view log analysis in Section 6.

While our dataset includes many relevant variables, it certainly does not include all. For example, a user's position or role, their expertise in programming or in the product domain, the specific tasks during which they visited documentation pages, and the actual contents of the documentation pages, are all likely to also correlate with differences in documentation usage but are absent from our data. Moreover, we only analyze data for four products of Google, therefore it remains unclear how our findings would generalize.

Finally, the page-view analysis can only be conducted after the documentation has been available for some time, allowing for the accumulation of extensive logs. Therefore, our analysis may not be applicable for documentation writers who need to assess their content pre-release or for documentation related to products with a limited user base.

Thus, we do not expect page-view log analyses like ours to obviate human studies or other more precise research methods. However, we do expect they could be fruitful as a first step or in conjunction with more precise but more costly research methods.

4 PHASE I: DISCERNING DOCUMENTATION USE PATTERNS IN LOG DATA

As an initial exploratory investigation to help contextualize our data, we conducted cluster analysis. This phase was necessary because although we know that developers will use documentation differently, we still know little about how much and in what ways it will differ "in the wild" and in our context. To efficiently explore the large dataset, we first used an automatic clustering analysis to discover discernible documentation usage patterns, and used sampling and qualitative analysis to further investigate the patterns.

4.1 Data Preparation

In preparation for clustering, we first aggregated each user's total dwell times (i.e., times spent on the different pages) in May 2020 across the 11 documentation page types in Table 1. We recorded separate entries for each of our four separate products, if the same user happened to access documentation pages for more than one of the products that month. We then represented each user's documentation visit profile as a vector of 11 elements, capturing the total times spent across each page type that month. Our supplementary material contains a sample of this data and its distribution.

Note that as a precaution before clustering, we filtered out outliers with total dwell times (sum over the 11 page types) outside of the $[\mu-3\sigma, \mu+3\sigma]$ interval (i.e., more than three standard deviations from the mean), as customary. In our sample, this corresponds to users who stayed shorter than 1.39 minutes or longer than 961.91 minutes in total across all documentation pages of each of our four products during the month (in May 2020). In addition, as the distributions of dwell times we observed tend to be right-skewed, we log-transformed all positive values. This is a common transformation [69] when the data vary a lot on the relative scale, as in our case — spending one more minute on a page is arguably much more noticeable for a 3-minute dwell time than a 10-hour dwell time.

4.2 Methodology

Out of many clustering approaches available, we adopted a protocol proposed by Zhao et al. [90], which is particularly well suited for large datasets. A common challenge with standard clustering methods is determining the appropriate number of natural clusters. Typically, one either chooses the number of clusters a priori, or applies techniques to automatically determine the "optimal" number of clusters. The former scenario is not applicable in our case (we do not have any empirical basis to expect a particular number of clusters), while traditional techniques to select the number of clusters automatically tend to be slow for large datasets like ours. The key innovation in the protocol by Zhao et al. [90] is combining two standard clustering techniques: first using a fast clustering method (k-means) to reduce the dimensionality of the clustering problem, and then applying a second clustering method (MeanShift) that automatically determines the number of clusters. This is computationally effective, as the second method only runs on the centroids generated by the first (k-means). To select the number of clusters as input for the first (fast) method, one typically chooses a significantly larger number than the plausible number of natural clusters, expecting that the second method will merge closely located centroids eventually to match the natural clusters. In determining the quality of the clustering results from the different parameters used, we used the following clustering performance score, as per Zhao *et al.*.

$$cp = 0.3 * E + 0.23 * D + 0.23 * \frac{k-m}{k} + 0.23 * \frac{N-n}{N} \quad (1)$$

In this score, the first and the second factors are used to reward the clustering performance using two well-known metrics: Shannon's entropy (E) and Dunn's index (D). The probability used for calculating Shannon's entropy score is the normalized number of users in each cluster. Thus, entropy assigns a high value to clustering results that have a uniform distribution of users across clusters. Dunn's index measures the compactness and separation of the clusters, by calculating the ratio of the smallest distance between observations not in the same cluster to the largest intra-cluster distance. The third and fourth factors are to penalize clustering results that are too naive or complex. The third penalizes the results that are too complex, that do not improve over the naive k-means results, where m (the number of clusters after MeanShift clustering) is close to k (the number of target k-means cluster that is significantly larger than the number of natural clusters). The fourth factor penalizes results that one big cluster contains most of the users in the dataset, where N is the number of total product users and n is the number of users in the biggest cluster.

After trying several values for k and eps in the equation 1, we obtained the highest cp score, 0.50, which is comparable to other works [78, 90], with E = 0.71, D = 0.15, m = 316, N = 94096, n = 9789. This result was obtained for $k = 400, eps = 1.25$, resulting in 320 clusters. Most clusters consist of around 300 users and there are 18 clusters consisting of more than 1,000 users.

4.3 Resulting Clusters

Figure 2 provides a summary of the resulting clusters, illustrating a wide array of documentation usage patterns within the 320 clusters. These patterns are evident through the variations in page views across the 11 documentation types. To comprehensively investigate these distinct patterns, after excluding small clusters with fewer than 100 users, we conducted open coding. The primary author assigned codes to the clusters, refined them iteratively, and subjected the emerged categories to a thorough review by the entire team of authors. In light of space limitations, we are able to present only four codes that were frequently assigned to clusters, along with representative examples of these clusters for reference.

Product explorers (Clusters 11, 16, 21). The time this group spent on documentation is not seemingly enough to digest the

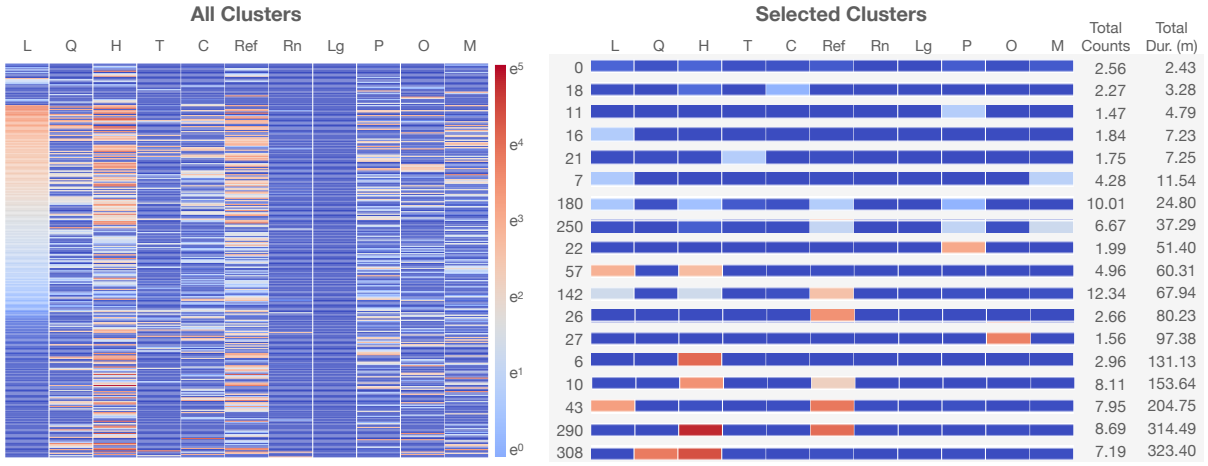

Figure 2: The heatmap of centroids of the 320 clusters (left), and a subset of them highlighted (right). Each row represents the documentation usage of each cluster (see Table 1 for the documentation type codes). The color indicates the dwell time in minutes, with the intensity encoded in e^n of time. The average total counts (# of documentation pages visited in May) and the average total dwell time (sum of dwell time on 11 documentation types) are also shown for the selected clusters (right) to help with interpretation, and the rows are sorted by the average total dwell time. For example, users of Cluster 18 (2nd row from the selected clusters) spent 3.28 minutes on average on the product documentation among 2.27 page visits on average, and spent $\approx e^1 = 2.7$ minutes on Concept type documentation.

information in the documentation, and would not help one to actually use a product. Furthermore, considering that the clustering was done with a month-full of user logs, visiting only one type of documentation for few times is not likely a usage pattern of an actual product user. Cluster 16, for instance, only visits one specific type of documentation, landing, a few times, and spent only a short time on the documentation (less than 10 minutes) on average.

Documentation Explorers (Clusters 7, 18, 180). Users in Cluster 180 were similar to product explorers in the sense that they only stayed for a relatively short time (less than 30 minutes over the month), but different in that they visited more documentation pages of multiple types. We infer that these users might be new to the documentation and might be exploring it to see the available information. For example, we speculate that they might have visited Landing documentation by searching for the product name in search engines, checked the prices from the Pricing page to see if they can adopt the product, and looked around Reference documentation to see the features available.

Task-oriented users (Clusters 6, 26, 27). Users of Cluster 6 showed more distinct behaviors. Although they only visited one specific type of documentation a few times, like product explorers, they stayed on the documentation pages much longer (on average, 131.13 minutes over the month). Based on the amount of time spent on documentation pages, we can infer that these users spent enough time to find what they were looking for from the information, and to fully digest it. From the number of visits, the users did not seem to explore the documentation, but stayed on few specific (or single) pages that they were interested in; this indicates that they were only interested in some of the product features, rather than an overall understanding of the product.

Versatile users (Clusters 43, 290, 308). These users visited multiple types of documentation pages and stayed long enough time on each of the type. For example, users in Cluster 290 seemed to be interested in the specific tasks described in How-to guide pages since they spent enough time on them, and perhaps visited Reference documentation from time to time when they needed more low-level information on the API calls and parameters.

We also discovered many other interesting documentation usage patterns, such as **Financial users** (Cluster 22), who stayed in Pricing documentation which only contains pricing information for an hour, and **Server engineering users** (Cluster 27), who almost exclusively visited Other documentation which provided resource-relevant information like locations of the servers.

5 PHASE II: FACTORS ASSOCIATED WITH DOCUMENTATION USE

With the exploration of clustering analysis results, we were able to discover various usage patterns, including those that were not actively discussed in the existing literature [21, 35, 47, 48], like product explorers or documentation explorers. However, we could only speculate about the intention and background of the users behind those diverse documentation usage patterns. Thus, we now bring together our informal observations from Part I with the literature on general information seeking and small-scale documentation studies, to derive and test hypotheses explaining the different usage patterns based on user characteristics.

5.1 Hypotheses Building

Given the absence of established theoretical frameworks elucidating documentation usage behaviors, we have chosen factors that might be associated with the developers' documentation usage, informed

by the prior work on developers' general information seeking in web search or software maintenance settings, and observations from the small-scale documentation studies [15, 17, 25, 40, 48].

Experience. Many studies have shown that the information seeking strategies of developers vary by their experience levels [21, 36, 38, 40, 44, 63]. For example, Costa et al. [17] found that documentation users with less experience with the software tended to use more types of documentation than more experienced users, and that tutorials and how-to videos were used by a greater percentage of newer users, and the newer users tended to use tech notes and forums less. Thus, we hypothesize,

$\mathbf{H}_1$. *High experience levels are positively associated with accessing documentation covering implementation details (Dev genre), whereas lower experience levels are positively associated with accessing documentation covering an overview of the products (Meta genre).*

Product Type. Differences in typical usage contexts of the products, such as project complexity and task categories, also influence developers general information seeking [21, 25, 50]. Within our dataset, P1 and P2 are application APIs whereas P3 and P4 are operations-related products for managing event streams and log data, and we expected to see different characteristics will come with different documentation usages, and we anticipate that these distinct characteristics will be associated with different documentation usage patterns. For instance, users of infrastructural APIs are more likely to be engaged in the maintenance of large-scale software projects, which implies a greater interest in system-level products and in system-level quality attributes. Conversely, application APIs are commonly adopted by smaller projects where the applications themselves serve as core components. Consequently, we expect that users of documentation for different products will tailor their utilization accordingly. Thus, we hypothesize:

$\mathbf{H}_2$. *Documentation usage of application APIs (P1, P2) differs from that of large-scale infrastructural APIs (P3, P4).*

Documentation Type Predisposition. Previous research has found that developers adopt different work styles, motivations, and characteristics, and they solve programming tasks differently [15], and human studies with documentation usage also reported similar findings [48]. The work styles of developers are less liable to change over time as opposed to levels of expertise, educational background, etc. Thus, we hypothesize that we can see similar patterns in the page-view logs, that developers will stick to documentation that suits their general information foraging strategy formed by their needs and preferences, without changing their documentation usage behavior much over time.

$\mathbf{H}_3$. *Users tend to use the same documentation type over time.*

Possible Intent. Prior work [10, 64] found that developers' web search behaviors vary with their information seeking intent: they visit different types of web pages, use different queries, and overall interact with webpages differently. In particular, developers were more likely to visit official software documentation during reminding sessions and third-party tutorials during learning sessions. Developers also tend to spend tens of minutes with learning intent, but only tens of seconds to remind. Times spent in between the two extremes were mostly with clarification intent. We posit that a similar behavioral pattern can be identified within documentation-based information seeking, wherein users invest substantial time per visit when their objective is to grasp complex concepts or protocols. Conversely, they allocate less time when verifying straightforward facts or utilizing documentation as an external memory aid [35]. Thus, we hypothesize:

$\mathbf{H}_4$. *Users who exhibit extended average page dwell times are more inclined to access documentation that offers tutorials (Guide genre), whereas users with shorter average page dwell times are more likely to access documentation providing straightforward factual information (Admin genre).*

Subsequent API Use. From multiple empirical studies, developers have reported that the quality of documentation is a highly influential factor in API selection process [82], and failure in effective information seeking within documentation leads them to give up on using the APIs [65]. Developers specifically reported that they examine the documentation up-front to determine "if there are good examples or tutorials that clearly explain how to use the library" [82] before they decide to adopt a library, showing the need of onboarding materials. Thus, we expect that:

$\mathbf{H}_5$. *Accessing documentation pages providing technical information for newcomers (guide genre) is positively associated with subsequent API calls by the same users.*

5.2 Data Preparation

We collected **pseudonymized user-level data** and **API usage data** to extract such factors of Google's documentation users, and test whether the hypotheses in the previous section are supported by the developers' documentation usage data at scale.

Experience. To investigate the effect of experience levels in documentation usage, we measure the documentation users' experience level using two variables: *overall platform experience* and *specific product experience*. We define the experience with the platform as the user account age, i.e., years passed since signing the platform terms and conditions[1]. We define the experience with a specific product as the total number of successful API requests made to that API over the previous three months (February, March, April 2020).

Product Type. To analyze the differences in documentation usage patterns, we recorded what each documentation usage data point was for.

Documentation Type Predisposition. As a proxy for one's possible predisposition for certain documentation types, we recorded the user's documentation *page views in the previous three months* (February, March, and April 2020).

Possible Intent. As a proxy for possible user intent when accessing the documentation, we recorded the *average per-page dwell time*, by dividing the total dwell time in May by the total number of documentation pages a user had visited.[2] We further grouped the

[1] For users who have never signed the terms and conditions for API usage, we assigned a value of 0.

[2] A more direct comparison to Brandt et al. [10] would require per-session dwell times, which we did not have access to, hence this approximation.

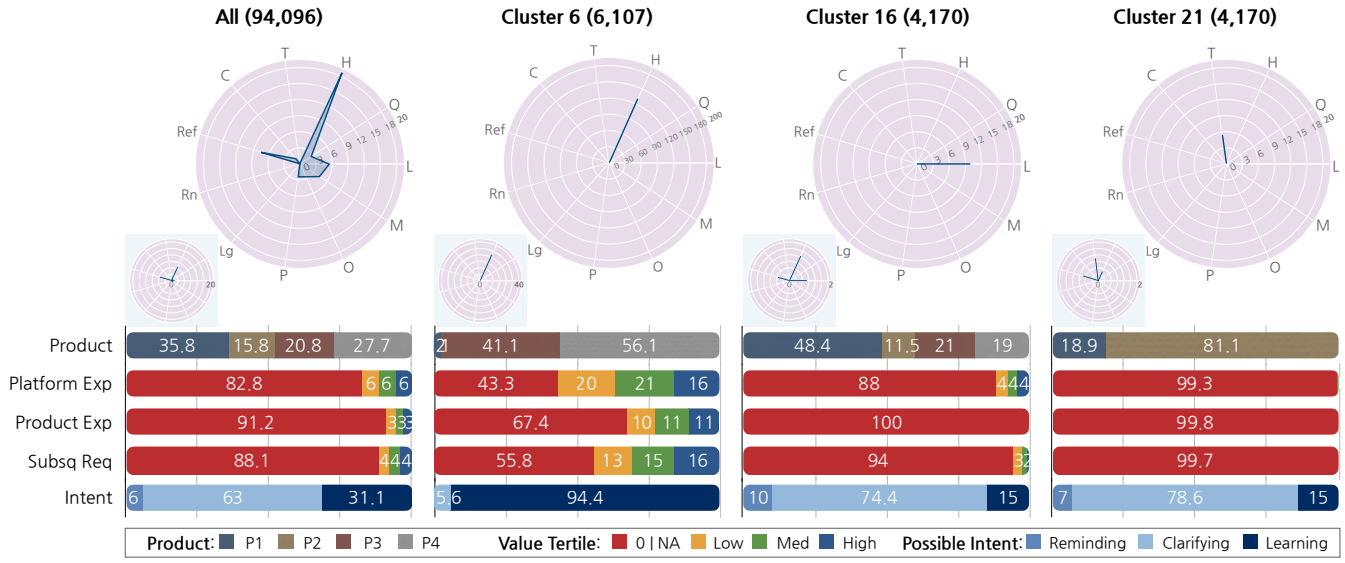

Figure 3: Highlights of the clustering analysis. Each polar plot displays the average time spent on each type of documentation (see Table 1 for the documentation type codes). The small polar plots show the average dwell time in the previous three months. Note that the ranges of the axes of the plots vary. Bar charts below the polar plots show the proportions (%) of each group in the cluster. For example, the charts of Cluster 21 can be interpreted as "In cluster 21, users without platform and product experience predominantly used Tutorial documentation (≈ 6 minutes) of P2 (81.1%) and P1 (18.9%), mostly for clarification purposes, without subsequent API requests."

data into three bins—less than 1 minute, between 1 minute and less than 10 minutes, and more than 10 minutes—to loosely correspond to the categories of intent (reminding, clarifying, and learning) identified by Brandt et al. [10]. The "more than 10 minutes" group most directly maps to a learning intent, while the other two groups possibly overlap with both reminding and clarifying.

Subsequent API Use. We mined the Google-collected API usage data (telemetry data) from June, July, and August 2020 corresponding to the subset of people in the aforementioned May-2020 documentation page-view log dataset, who also made subsequent API requests using the web-based services. This was possible because the pseudonymization strategy has random but persistent IDs that are consistent across documentation and API usage data. Specifically, we extracted the *number of successful API requests* made by each user (i.e., with 2XX return codes).

5.3 Sanity Test with Cluster Exploration

Before we formally test our hypotheses, we checked whether the hypotheses derived based on the general information-seeking literature apply at all to developers' documentation usage patterns observed in our data, by checking different clusters' user distributions. To help with our exploration, we first visualized each cluster's average dwell time, and discretized the numerical variables into four groups for each user factor, based on percentiles: `0|NA` (factor=0), `Low` (0-33%), `Medium` (34-66%), `High` (67-100%). Figure 3 shows the visualizations of the entire dataset, and three example clusters due to the space limit. We have included visualizations of other large clusters with over 500 users in our supplementary materials.[3] Using the visualizations, we selected clusters with different distributions for each factor we hypothesized would influence documentation usage. We then compared their documentation usage patterns to check if the factors were related to variations in these patterns.

H_1 (Experience): Comparing clusters with a lot of experienced users (e.g., Cluster 6, Cluster 26, Cluster 27) and clusters mostly with new users (e.g., Cluster 16, Cluster 21, Cluster 22), the dwell time of the latter was relatively shorter compared to the former. We also found that most of the clusters with more experienced users spend time on the documentation that describes lower-level details, such as Reference or How-to guide documentation, without needing to visit introductory documentation like landing or marketing pages. On the other hand, clusters with new users showed diverse documentation usage patterns, which might be because they browse the documentation while considering adopting the APIs while still being relatively unfamiliar with the products, instead of trying to learn to use the products.

H_2 (Product type): We observed that documentation usage for P1 and P2, on the one hand, and P3 and P4, on the other hand, is internally similar in different clusters — many users of the pairs ended up clustered together (e.g., Cluster 21 and Cluster 11 for P1 and P2, and Cluster 6 and Cluster 10 for P3 and P4). Clusters with a lot of P3 and P4 users visited How-to guide documentation, which might be due to their typical high project complexity requiring system-level configurations of multiple products in Google platform. In addition,

[3]To protect privacy (see section 3.3), we have not included visualizations of the remaining clusters. However, we note that these large clusters account for 77% of the total users in our dataset.

we observed that clusters with the majority of users using application APIs show longer pricing documentation usage, whereas clusters of infrastructural API users show almost zero usage of pricing documentation. This could be explained by the usage context of the products: Infrastructural API users maintaining large software systems are also often employees of large corporations, with accounting and legal teams taking care of administrative tasks, removing the need to visit Pricing or Legal documentation, whereas application APIs are often used by smaller companies or individual projects whose developers are more likely to be responsible for administrative tasks.

H_3 (Documentation type predisposition): We observed a consistent trend where clusters of users who spent an extended amount of time on specific types of documentation in May also exhibited a prolonged engagement with the same documentation in previous months. For instance, consider Cluster 6 (task-oriented users), whose members demonstrated a substantial dwell time on How-to documentation in May; they also ranked second in terms of How-to documentation usage in previous months, among the clusters we analyzed. Similarly, Cluster 22 (financial users), which had the longest dwell time of Pricing documentation in May, consistently showed the longest dwell time for Pricing documentation in preceding months. Furthermore, even among clusters with lighter documentation usage, we noticed a parallel pattern: the dwell times from previous months mirrored the patterns observed in May.

H_4 (Possible intent): Comparing clusters with a lot of users with "reminding" or "clarifying" intention (e.g., Cluster 0, Cluster 16, Cluster 18) with clusters with mostly "learning" intention (e.g., Cluster 6, Cluster 26, Cluster 27), we observe that the former users spent much less time on the documentation pages on average, and also focused on documentation types like marketing and landing pages, which often provide an overview and administrative facts of the APIs, consistent with the "reminding" and "clarifying" intent reported by Brandt et al. [10]. In contrast, clusters with a lot of "learning" users visited documentation that provides more detailed guidance on how to use the products, like How-to documentation.

H_5 (Subsequent API use): Comparing the clusters of users who made no or low subsequent API calls (e.g., Cluster 0, Cluster 16, Cluster 22) with the clusters of users who made subsequent API calls (e.g., Cluster 6, Cluster 26, Cluster 27), the latter had spent longer overall browsing documentation pages, and had spent most of their time on How-to guide and Reference pages as opposed to marketing pages, which could indicate that many had already decided to adopt the API. We also observed that the degree of such association may vary with the product. For example, compared to the users in Cluster 7 (documentation explorers) who visited Landing and Marketing documentation and had similar average dwell times, far more users in Cluster 16 (product explorers) actually made calls to the API in the subsequent months. This might be explained by their usage context: the product proportions were relatively equal in Cluster 16, but most of the Cluster 7 users visited only P1 documentation. Thus, we expect that users will need different types of information depending on their usage context, and thus the usefulness of documentation types may also vary.

5.4 Regression Analysis

Next, we formally test the hypotheses above on our entire sample. First, we use multiple regression to test how much the various user-level characteristics we hypothesized about in $\mathbf{H}_1$-$\mathbf{H}_4$ can explain people's logged documentation visits to pages in each of our four genres (recall Table 1). Second, we test $\mathbf{H}_5$, i.e., to what extent developers' documentation visits in each of our four genres can explain their subsequent API use, again using multiple regression.

We start by estimating four logistic regression models, one for each documentation genre; see model specification in the supplementary material. In each model, the dependent variable is a boolean variable "*dwell time* > 0" indicating whether or not a user in our sample accessed documentation pages of that particular genre.[4] In addition, each model includes explanatory variables corresponding to $\mathbf{H}_1$ (overall platform experience, specific product experience), $\mathbf{H}_2$ (product), $\mathbf{H}_3$ (documentation use in the previous three months), and $\mathbf{H}_4$ (average page dwell time); see section 3.2 for definitions. All models include all variables. By jointly estimating the different β coefficients, this model allows us to estimate the strength of the association between each explanatory variable and the likelihood that users access documentation pages from each genre, *independently of the other variables included in the model*. Then, the p-value of, say, the estimated β_1 coefficient allows us to test $\mathbf{H}_1$, i.e., whether there is a correlation between platform experience and the likelihood of accessing documentation genres being modeled. Similarly, we could test for correlations between platform experience and likelihood of accessing documentation pages from the other three genres with the other three models.[5]

To test $\mathbf{H}_5$ we use a similar strategy, estimating one logistic regression model with a boolean-dependent variable

"*subsequent requests* > 0". We restrict this analysis to the subset of users who have not made any API requests in the past months (more likely to be new users), since we expect the results to be more actionable for this subset in terms of growing the API user base. We include all the same independent variables as before (the ones not directly tied to the hypotheses act as controls), except specific_product_experience which is by definition null for these users. We also include an interaction with product to test the effect of differences in products.

Overall, we took several steps to increase the robustness of our estimated regression results. First, we removed outliers (i.e., observations more than 3 standard deviations beyond the mean) for highly skewed count variables and applied log-transformations to improve heteroskedasticity. Second, we checked for multicollinearity using the Variation Influence Factor (VIF) and only kept variables having VIF lower than 2.5, following Johnston et al. [32]. Third, since we estimate multiple models, each with multiple variables, thus increasing the risk of Type I errors, we conservatively adjusted all p-values using Holm's correction procedure [28]. Furthermore,

[4] See the supplementary material for consistent results for complementary count-based, linear regression models that further investigate the time spent on the different pages.

[5] Note that our research hypotheses in section 5.1 are not all equally broad, i.e., they don't all cover all documentation genres or even the same documentation genres. Our choice to model each documentation genre separately is flexible enough to allow us to draw conclusions about all hypotheses, including the broader ones, by qualitatively comparing results from the relevant models. For example, we can reason about a particular estimated coefficient β being statistically significant in multiple models corresponding to multiple documentation genres.

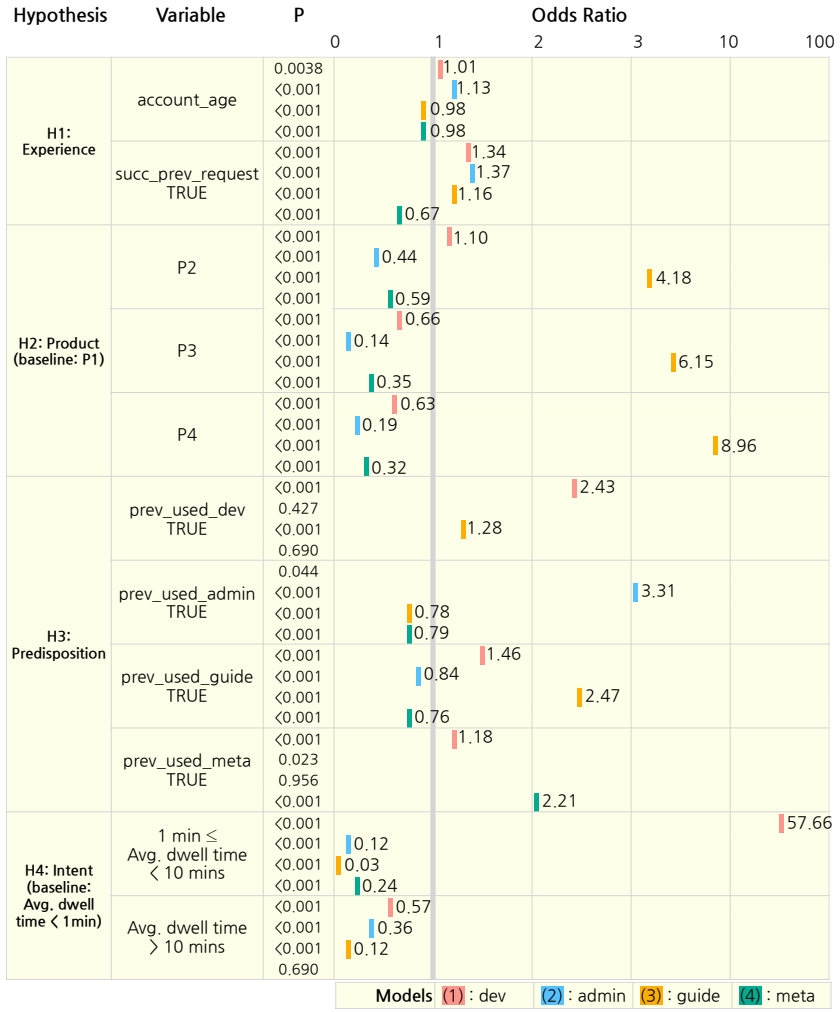

Figure 4: *Top*: Estimated odds ratios from the regression modeling *dwell time* > 0 for our four documentation genres. For example, the odds of accessing Dev type documentation (pink) are 1.01 times higher among users with one extra year of platform experience. *Bottom*: Estimated odds ratios from the regression modeling *subsequent requests* > 0. Variables without statistically significant coefficients (adjusted $p \geq 0.01$) are omitted.

we only considered model coefficients worthy of discussion if the adjusted p-values were statistically significant at 0.01 level instead of the more common 0.05.

5.5 Results

Figure 4-top summarizes the documentation-access logistic regression results across the four models we estimated (one per genre) to test $\mathbf{H}_1$-$\mathbf{H}_4$. We present our results in terms of odds ratios (OR) instead of regression coefficients to ease interpretation. All four models are plausible, with Nagelkerke [53] pseudo R^2 values (deviance explained) of 74% for dev, 16% for admin, 44% for guide, and 55% for the meta documentation genre. Similarly, Figure 4-bottom summarizes the subsequent-usage logistic regression model testing $\mathbf{H}_5$. The relatively high explanatory power of the models indicates that at least some of the patterns of documentation usage align with user characteristics and API usage behaviors.

$\mathbf{H}_1$ (Experience): supported. Results from the dev and meta-genre models are consistent with the hypothesis. For example, the odds of accessing reference documentation and other dev pages are 1.34 times higher among people with prior experience with the products (product experience), i.e., those who made successful API requests in the past, compared to those without, and the odds of accessing such pages are 1.01 times as high among users with one extra year of platform experience. Similarly, the odds of accessing marketing and other meta documentation are lower (OR = 0.67) among people with prior experience with the products (product experience), and the odds of accessing such pages are 0.98 times as high among users with one extra year of platform experience.

Interestingly, the results from the admin-genre model align more with the documentation genres covering implementation details than meta: the odds of accessing pricing, legal, and other admin documentation are also higher (1.37 times) among people with prior experience with the products compared to those without. This could indicate that the information in admin documentation is not only needed once, when people make API adoption decisions, but rather is consistently needed throughout their use of the API.

$\mathbf{H}_2$ (Product type): supported. All four models support the hypothesis: taking P1 as the reference, the magnitude of differences between P1 and P2 is consistently smaller than either P1 and P3 or P1 and P4; i.e., the documentation page visits of large-scale infrastructural products tends to differ starkly from that of application products. Taking the dev-genre model as an example, the odds of accessing the documentation pages are only 1.1 times higher among visitors to P2 documentation compared to P1, but 0.66 and 0.63 times as high among visitors to P3 and P4 compared to P1.

$\mathbf{H}_3$ (Documentation type predisposition): supported. All models show strong effects of documentation type consistency: there are correlations between the past and the future access to some types of documentation. For instance, in the admin-genre model the odds of accessing admin-genre documentation pages are 3.31 times higher among people who had also accessed such pages in the past three months compared to people who had not. As many different pages of documentation are included in each type of documentation, and the analysis is done at a month-level, this result does not provide conclusive evidence of the users' preference for the contents or structure of documentation pages. However, it still suggests that the documentation users have types of documentation they are more familiar with, and can access them repeatedly.

$\mathbf{H}_4$ (Possible intent): only partially supported. The results for this hypothesis are mixed. On the one hand, the dev-genre model reveals a clear difference between people with long and short average per-page dwell times, as hypothesized: the odds of accessing reference documentation and other dev pages are 0.57 times lower among people with average dwell times greater than 10 minutes compared to those with average dwell times less than a minute. The model also reveals that the odds of accessing dev-genre documentation are greatest (57 times higher) among people with average dwell times between one and 10 minutes. Similarly, the models for admin- and meta-genre pages, which include marketing and pricing, are generally supporting the hypothesis.

In contrast, the model for guide-genre documentation points to the opposite finding than hypothesized when comparing to people with average dwell times less than a minute (the group with the shortest dwell times, set as the baseline in our models): the odds of accessing tutorials, how-to documentation, and the like are lower, not higher, among both people with average dwell times between one and 10 minutes as well as people with average dwell times greater than 10 minutes, compared to those with average dwell times less than a minute.

One potential explanation is that many users might use the guide documentation as a cheat-sheet, from where they copy and paste various API boilerplate [54] or usage examples. Although guide documentation was originally intended to introduce and explain products to relatively inexperienced users, it appears to be widely used by users with diverse intentions.

$\mathbf{H}_5$ (Subsequent API use): supported. The model reveals a strong correlation between accessing guide-genre documentation pages and subsequent API calls: the odds of making successful API calls in the subsequent three months are 3.92 times higher among people who visited guide documentation compared to those who did not.

Modeling interactions revealed that the strength of the association between visiting guide documentation and making subsequent API requests is weaker for P3 and P4 users relative to P1, while the interaction is not statistically significant for P2 users relative to P1. That is, as above, we see consistent differences between the two large-scale infrastructural APIs and the two application APIs.

6 DISCUSSION

We investigated the feasibility of using documentation page-view logs to inform the design of documentation. Through a series of hypotheses derived from the literature, contextualized by an exploratory analysis of our page-view log data (§4), and subsequently validated through a large-scale regression analysis (§5), we discovered that there are multiple discernible patterns of documentation use, even when the documentation pertains to the same platform, or even the same products.

6.1 Feasibility of Log Analysis for Documentation Review

Large-scale log analysis helps discover unexpected use. As large-scale log analysis allows analyzing all documentation usage,

with less researcher efforts and costs, we could explore diverse documentation usage patterns. For example, in addition to users mainly using documentation for API learning, which was often studied in the existing literature that used smaller-scale qualitative approaches [21, 35, 47, 48], the clustering analysis discovered additional large clusters of users who only check pricing documentation (Cluster 22: financial users), or that many users make many API requests without even visiting reference documentation (Cluster 27: task-oriented users). The cluster exploration and the hypotheses testing also revealed that expected documentation usage can differ from actual use. For example, although how-to documentation is often regarded as introductory for new users [21], we observed that users with more product experience made more visits to the guide documentation (Cluster 6: task-oriented users) than those with less product experience, which was also confirmed by the regression analyses ($\mathbf{H}_1$). While the cause or intent behind these unexpected uses cannot be found solely with log analysis, our observations might be useful in designing more focused human studies. Moreover, we believe that a similar analysis can be used to answer broader research questions like "How does documentation usage change over time as users develop their expertise with the products?", or "What are the strategies developers use for information seeking in documentation?"

Page-view log analysis is informative but could be further refined. The analysis could be extended to also account for the structure and content of the documentation pages, in addition to the factors we considered. For example, although the top web search results given the query `Google [product]` were marketing documentation for all four products, the second result varied between a guide documentation page for P4, and landing pages for P1, P2, and P3. Thus, in interpreting the differences in documentation usage between products, whether intended or not, differences between the documentation structure and external resources should also be taken into account. Analyzing *referrer* pages, i.e., the pages accessed by a user prior to loading a particular web page, might be useful in understanding how such differences affect the documentation use [60]. We propose this direction for future research.

In practice, the analysis plan can be adapted based on the analysis goals. In this paper, we employed a mixed-method approach to gain a comprehensive understanding of Google documentation usage. This involved both exploring the data and validating our hypotheses. Each of these analyses complements the other, offering distinct advantages and considerations. For example, clustering analysis proves valuable in uncovering common and unexpected usage patterns, requiring less quantitative data analysis expertise to get started. However, it is important to note that interpreting clustering results can be subjective, and conducting a detailed investigation of every cluster may not always be practical. Subsequently, performing regression analysis adds a layer of confidence to our findings, providing a comprehensive overview of the dataset. In practice, it may not always be necessary or feasible to conduct both types of analysis due to differences in skill requirements. In such cases, the choice between the two can be made based on the specific goals of the log analysis. For instance, a user experience (UX) researcher seeking a lightweight usability review might opt for a quick cluster analysis and interpretation as demonstrated in Section 5.3. If stronger evidence is needed to support hypotheses, especially for design refactoring, engaging a quantitative UX researcher or data scientist to perform regression analysis following a clustering study would be a more suitable approach.

6.2 Recommendations for Documentation Providers

Through the log analysis, we found that documentation usage can vary based on the users' experience in product and platform ($\mathbf{H}_1$), the type of product described ($\mathbf{H}_2$), and many other factors ($\mathbf{H}_3$, $\mathbf{H}_4$). This suggests that established knowledge on documentation usage may not always be generalizable to all target users. Here, we highlight some of the specific implications for how to design improved documentation catering to users with different characteristics.

Explicitly mention the target audience of documentation. Previous studies [47] found that developers often experience difficulty in determining which documentation type to select when searching for a particular piece of information. We posit that this is because documentation for different products adheres to varying documentation standards and categorizes information differently, and it takes time for developers to learn these distinctions. Since we confirmed that developers' documentation visits are correlated with their characteristics, we posit that explicitly indicating the intended audience of the documentation will assist them in selecting the appropriate types and pages of documentation to access (i.e., provide strong "scent" in the information foraging theory [61]).

Duplicate important information for information discovery. As our models show ($\mathbf{H}_3$), users are more likely to visit types of documentation when they have accessed in the past. Although it is often considered to be better to *modularize* the documentation, this can be problematic if important information is only presented on a specific page, as the user might not always discover that [29, 56]. This observation is consistent with the finding of Meng et al. [47] that developers often skip sections in the software documentation based on their problem solving strategies. Thus, to reduce the risk of developers missing important information, we recommend providing such information in multiple types of documentation, or at least providing prominent functional links to the page providing such information.

Provide product-specific starting points. We discovered that there are variations in visit patterns among products with distinct characteristics. This is expected because different types of information are provided and needed depending on the purpose or domain of the product ($\mathbf{H}_2$). For instance, for infrastructural products such as P3 and P4, many users (Cluster 2: task-oriented users) accessed how-to guides providing instructions for the configuration settings, but for application products like P1 and P2, many users visited tutorial documentation pages providing walkthroughs for a simple use case (Cluster 21: product explorers) that aid new users in quickly familiarizing themselves with the products. However, for users who are new to the products with little understanding of them, it will be challenging to know what documentation type or page will be the best starting point [34], especially because there are a plethora of documentation pages per product. Thus, to help the new users quickly grasp the gist of the products, we recommend

providing product-specific recommendations about which documentation pages to use to start learning, as similarly recommended in Jeong et al. [31]. Most commonly accessed documentation pages or pages that correlate with subsequent API requests, which can be acquired from the page-view logs, will be good candidates for the recommendation, as they were already proven to be useful for other users. We note that we do not recommend changing the documentation templates or navigation structures, because inconsistent inter-product information organization can hinder information foraging of users, especially those who use multiple products from Google. A designated space for the product-specific documentation recommendation in a landing page or a navigation tab will allow users to know where to look if they become lost.

Nudge new product users to visit guide-genre documentation. When developers select third-party libraries, the quality of documentation is perceived as a good sign of the library's quality [82], and when a user is not able to find appropriate learning resources, it becomes a major obstacle in getting to know the libraries resulting in user frustration [65]. Our results suggest that guide-genre documentation is particularly effective in influencing the decision to adopt a product ($\mathbf{H}_5$), although one might think that landing documentation is beneficial for them since it provides an overview of the products. We believe that guide-genre documentation is helpful in making the adoption decision, as it describes what the products offer and help developers gauge what they need to do for onboarding, which corresponds to what new users look for from documentation [65]. Thus, although other documentation pages will be useful in the end, nudging developers to visit guide-genre documentation as early as possible may help them perceive the quality of documentation positively, and adopt the API.

6.3 Longer-term Vision: Personalization

While we distilled actionable recommendations for how to adjust the design of software documentation taking into account many dimensions of user characteristics that might affect their usage, doing this manually may be unrealistic when many products are involved. Instead, we argue that the time is ripe for approaches to *automatically personalize* the documentation. Personalization is not a new topic and has already proven to be effective for other services like media streaming and search engines [79]. Prior research on general web search has also made significant progress in designing effective personalized recommender systems to increase the long-term engagement of users [86], using both implicit (e.g., dwell time [86, 87]) and explicit (e.g., item rating [4, 5]) feedback mined from historical interaction data as an indicator of users' interests and needs. As the dwell time mined from documentation page-view logs can capture some user characteristics, in addition to the interaction histories that page-view logs contain by design, we expect that personalizing approaches can also be used in the documentation domain. Here, we present three directions to improve developers' information foraging on documentation using page-view logs.

Documentation recommendation. First, we argue that it is time to go from static approaches of documentation recommendation (for example, consider the omnipresent navigation links like "Recommended content" or "What's next" or "Next topic," that typically point to the same target page regardless of which user is browsing) to dynamic ones that take user characteristics into account to provide more relevant suggestions. An ideal scenario is perhaps one where the recommender system has access to the developer's code repository or profile, that reveal the developers' needs and background that are known to correlate with their documentation usage (e.g., their product and platform experience), as we discovered from the analysis. Short of that, we show that some signals about user-level characteristics are present in much more modest and more widely-available log data on previous documentation page visits. A recommender system could learn to profile users based on previous page visits (similar to our clustering) and, given that knowledge, suggest the next documentation pages to visit from among those that users in the same cluster have visited or interacted with before.

Within-documentation search. Personalization can also be applied to within-documentation search engines. Many previous studies of within-documentation search engines showed the need for efficient navigation [31]. Typically, software documentation contains information for both novices and experts, sometimes implicitly within a single page, other times explicitly across dedicated separate pages. For example, a difference between a 'basic' and an 'advanced' tutorial could be that the advanced tutorial describes APIs with more flexible capabilities, which require additional parameters. One way to personalize is query modification [72], by expanding the user query using additional terms inferred from user profiles. As above, the user profiles can be approximated from documentation page view logs; for example, when a user's documentation page view pattern is similar to Cluster 6 (task-oriented users), with high levels of guide documentation visits that correlate with product experience level, the system can infer that the user is experienced. Then, given a search query "how to set up P1," the system could augment the query along the lines "how to set up P1 *advanced user*," which should bias the search results towards the dedicated advanced pages.

Documentation filtering. Another idea is that a "smart" documentation system could automatically filter what information is being shown depending on the user. For example, when a user has already accessed platform-common information (e.g., authentication) from other products, the system can hide/fold such parts for new APIs the user is reading about, to make information foraging more efficient. Similarly, one could imagine hiding/folding other parts of a documentation page, such as the code examples, for users that prefer to develop a more conceptual understanding first [47]. These examples both require data on historical accesses of other documentation pages by the same users (or by users in the same cluster), which is often included in the page-view logs.

7 CONCLUSION

Based on our exploratory clustering analysis and hypothesis testing, we identified distinct documentation usage patterns and demonstrated that user factors partially explain the differences in such patterns. This enabled us to derive meaningful implications for documentation design, both specific to Google and in a broader context. Thus, we conclude that leveraging documentation logs at scale is both feasible and valuable and will allow documentation designers to generate actionable insights during their documentation design review.

ACKNOWLEDGMENTS

This research was funded in part by the NSF under grant CCF-2007482. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsors. We would like to thank anonymous external reviewers for their valuable feedback.

REFERENCES

[1] 2023. Google Documentation Style Guide. https://developers.google.com/style
[2] 2023. Google Privacy Principles. https://policies.google.com/privacy
[3] 2023. Google Safety Principles. https://safety.google/principles/
[4] Mohamed Hussein Abdi, George Onyango Okeyo, and Ronald Waweru Mwangi. 2018. Matrix Factorization Techniques for Context-Aware Collaborative Filtering Recommender Systems: A Survey. *Comput. Inf. Sci.* 11, 2 (2018), 1–10. https://doi.org/10.5539/cis.v11n2p1
[5] Deepak Agarwal, Bee-Chung Chen, Pradheep Elango, and Raghu Ramakrishnan. 2013. Content recommendation on web portals. *Commun. ACM* 56, 6 (2013), 92–101. https://doi.org/10.1145/2461256.2461277
[6] Emad Aghajani, Csaba Nagy, Olga Lucero Vega-Márquez, Mario Linares-Vásquez, Laura Moreno, Gabriele Bavota, and Michele Lanza. 2019. Software documentation issues unveiled. In *Proceedings of the 41st International Conference on Software Engineering, ICSE 2019, Montreal, QC, Canada, May 25-31, 2019*. IEEE / ACM, 1199–1210. https://doi.org/10.1109/ICSE.2019.00122
[7] Michael J Albers. 2003. Multidimensional audience analysis for dynamic information. *Journal of Technical Writing and Communication* 33, 3 (2003), 263–279.
[8] Erik Andersen, Eleanor O'rourke, Yun-En Liu, Rich Snider, Jeff Lowdermilk, David Truong, Seth Cooper, and Zoran Popovic. 2012. The impact of tutorials on games of varying complexity. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 59–68.
[9] James D. Arthur and K. Todd Stevens. 1992. Document quality indicators: A framework for assessing documentation adequacy. *J. Softw. Maintenance Res. Pract.* 4, 3 (1992), 129–142. https://doi.org/10.1002/SMR.4360040303
[10] Joel Brandt, Philip J. Guo, Joel Lewenstein, Mira Dontcheva, and Scott R. Klemmer. 2009. Two studies of opportunistic programming: interleaving web foraging, learning, and writing code. In *Proceedings of the 27th International Conference on Human Factors in Computing Systems, CHI 2009, Boston, MA, USA, April 4-9, 2009*. ACM, 1589–1598. https://doi.org/10.1145/1518701.1518944
[11] Andrea Bunt, Patrick M. J. Dubois, Ben Lafreniere, Michael A. Terry, and David T. Cormack. 2014. TaggedComments: promoting and integrating user comments in online application tutorials. In *CHI Conference on Human Factors in Computing Systems, CHI'14, Toronto, ON, Canada - April 26 - May 01, 2014*. ACM, 4037–4046. https://doi.org/10.1145/2556288.2557118
[12] Georg Buscher, Ludger van Elst, and Andreas Dengel. 2009. Segment-level display time as implicit feedback: a comparison to eye tracking. In *Proceedings of the 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2009, Boston, MA, USA, July 19-23, 2009*. ACM, 67–74. https://doi.org/10.1145/1571941.1571955
[13] Jie-Cherng Chen and Sun-Jen Huang. 2009. An empirical analysis of the impact of software development problem factors on software maintainability. *J. Syst. Softw.* 82, 6 (2009), 981–992. https://doi.org/10.1016/J.JSS.2008.12.036
[14] Parmit K. Chilana, Amy J. Ko, and Jacob O. Wobbrock. 2012. LemonAid: selection-based crowdsourced contextual help for web applications. (2012), 1549–1558. https://doi.org/10.1145/2207676.2208620
[15] Steven Clarke. 2007. What is an End User Software Engineer?. In *End-User Software Engineering, 18.02. - 23.02.2007 (Dagstuhl Seminar Proceedings, Vol. 07081)*. Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany. http://drops.dagstuhl.de/opus/volltexte/2007/1080
[16] Mark Claypool, Phong Le, Makoto Waseda, and David Brown. 2001. Implicit interest indicators. In *Proceedings of the 6th International Conference on Intelligent User Interfaces, IUI 2001, Santa Fe, NM, USA, January 14-17, 2001*. ACM, 33–40. https://doi.org/10.1145/359784.359836
[17] Carlos J Costa, Manuela Aparicio, and Robert Pierce. 2009. Evaluating information sources for computer programming learning and problem solving. In *Proceedings of the 9th WSEAS International Conference on APPLIED COMPUTER SCIENCE*. 218–223.
[18] Bill Curtis and Jakob Nielsen. 1995. Applying Discount Usability Engineering. *IEEE Softw.* 12, 1 (1995), 98–100. https://doi.org/10.1109/52.363161
[19] Andreas Dautovic. 2011. Automatic assessment of software documentation quality. In *26th IEEE/ACM International Conference on Automated Software Engineering (ASE 2011), Lawrence, KS, USA, November 6-10, 2011*. IEEE Computer Society, 665–669. https://doi.org/10.1109/ASE.2011.6100151
[20] Ekwa Duala-Ekoko and Martin P. Robillard. 2012. Asking and answering questions about unfamiliar APIs: An exploratory study. In *34th International Conference on Software Engineering, ICSE 2012, June 2-9, 2012, Zurich, Switzerland*. IEEE Computer Society, 266–276. https://doi.org/10.1109/ICSE.2012.6227187
[21] Ralph H. Earle, Mark A. Rosso, and Kathryn E. Alexander. 2015. User preferences of software documentation genres. In *Proceedings of the 33rd Annual International Conference on the Design of Communication, SIGDOC 2015, Limerick, Ireland, July 16-17, 2015*. ACM, 46:1–46:10. https://doi.org/10.1145/2775441.2775457
[22] Angela Fan, Beliz Gokkaya, Mark Harman, Mitya Lyubarskiy, Shubho Sengupta, Shin Yoo, and Jie M. Zhang. 2023. Large Language Models for Software Engineering: Survey and Open Problems. arXiv:2310.03533 [cs.SE]
[23] Andrew Forward and Timothy Lethbridge. 2002. The relevance of software documentation, tools and technologies: a survey. In *Proceedings of the 2002 ACM Symposium on Document Engineering, McLean, Virginia, USA, November 8-9, 2002*. ACM, 26–33. https://doi.org/10.1145/585058.585065
[24] Steve Fox, Kuldeep Karnawat, Mark Mydland, Susan T. Dumais, and Thomas White. 2005. Evaluating implicit measures to improve web search. *ACM Trans. Inf. Syst.* 23, 2 (2005), 147–168. https://doi.org/10.1145/1059981.1059982
[25] Luanne Freund. 2015. Contextualizing the information-seeking behavior of software engineers. *J. Assoc. Inf. Sci. Technol.* 66, 8 (2015), 1594–1605. https://doi.org/10.1002/ASI.23278
[26] Golara Garousi, Vahid Garousi-Yusifoglu, Günther Ruhe, Junji Zhi, Mahmood Moussavi, and Brian Smith. 2015. Usage and usefulness of technical software documentation: An industrial case study. *Inf. Softw. Technol.* 57 (2015), 664–682. https://doi.org/10.1016/J.INFSOF.2014.08.003
[27] Tovi Grossman, Justin Matejka, and George W. Fitzmaurice. 2010. Chronicle: capture, exploration, and playback of document workflow histories. In *Proceedings of the 23rd Annual ACM Symposium on User Interface Software and Technology, New York, NY, USA, October 3-6, 2010*. ACM, 143–152. https://doi.org/10.1145/1866029.1866054
[28] Sture Holm. 1979. A Simple Sequentially Rejective Multiple Test Procedure. *Scandinavian Journal of Statistics* 6, 2 (1979), 65–70. http://www.jstor.org/stable/4615733
[29] Amber Horvath, Sachin Grover, Sihan Dong, Emily Zhou, Finn Voichick, Mary Beth Kery, Shwetha Shinju, Daye Nam, Mariann Nagy, and Brad A. Myers. 2019. The Long Tail: Understanding the Discoverability of API Functionality. In *2019 IEEE Symposium on Visual Languages and Human-Centric Computing, VL/HCC 2019, Memphis, Tennessee, USA, October 14-18, 2019*. IEEE Computer Society, 157–161. https://doi.org/10.1109/VLHCC.2019.8818681
[30] Amber Horvath, Michael Xieyang Liu, River Hendriksen, Connor Shannon, Emma Paterson, Kazi Jawad, Andrew Macvean, and Brad A. Myers. 2022. Understanding How Programmers Can Use Annotations on Documentation. In *CHI '22: CHI Conference on Human Factors in Computing Systems, New Orleans, LA, USA, 29 April 2022 - 5 May 2022*. ACM, 69:1–69:16. https://doi.org/10.1145/3491102.3502095
[31] Sae Young Jeong, Yingyu Xie, Jack Beaton, Brad A. Myers, Jeffrey Stylos, Ralf Ehret, Jan Karstens, Arkin Efeoglu, and Daniela K. Busse. 2009. Improving Documentation for eSOA APIs through User Studies. In *End-User Development, 2nd International Symposium, IS-EUD 2009, Siegen, Germany, March 2-4, 2009. Proceedings (Lecture Notes in Computer Science, Vol. 5435)*. Springer, 86–105. https://doi.org/10.1007/978-3-642-00427-8_6
[32] Ron Johnston, Kelvyn Jones, and David Manley. 2018. Confounding and collinearity in regression analysis: a cautionary tale and an alternative procedure, illustrated by studies of British voting behaviour. *Quality & Quantity* 52 (07 2018), 1–20. https://doi.org/10.1007/s11135-017-0584-6
[33] Youngho Kim, Ahmed Hassan Awadallah, Ryen W. White, and Imed Zitouni. 2014. Modeling dwell time to predict click-level satisfaction. In *Seventh ACM International Conference on Web Search and Data Mining, WSDM 2014, New York, NY, USA, February 24-28, 2014*. ACM, 193–202. https://doi.org/10.1145/2556195.2556220
[34] Amy J. Ko, Robert DeLine, and Gina Venolia. 2007. Information Needs in Collocated Software Development Teams. In *29th International Conference on Software Engineering (ICSE 2007), Minneapolis, MN, USA, May 20-26, 2007*. IEEE Computer Society, 344–353. https://doi.org/10.1109/ICSE.2007.45
[35] Amy J. Ko, Brad A. Myers, Michael J. Coblenz, and Htet Htet Aung. 2006. An Exploratory Study of How Developers Seek, Relate, and Collect Relevant Information during Software Maintenance Tasks. *IEEE Trans. Software Eng.* 32, 12 (2006), 971–987. https://doi.org/10.1109/TSE.2006.116
[36] Amy J. Ko and Yann Riche. 2011. The role of conceptual knowledge in API usability. In *2011 IEEE Symposium on Visual Languages and Human-Centric Computing, VL/HCC 2011, Pittsburgh, PA, USA, September 18-22, 2011*. IEEE, 173–176. https://doi.org/10.1109/VLHCC.2011.6070395
[37] Dmitry Lagun and Mounia Lalmas. 2016. Understanding User Attention and Engagement in Online News Reading. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining, San Francisco, CA, USA, February 22-25, 2016*. ACM, 113–122. https://doi.org/10.1145/2835776.2835833
[38] Thomas D. LaToza, Gina Venolia, and Robert DeLine. 2006. Maintaining mental models: a study of developer work habits. In *28th International Conference on Software Engineering (ICSE 2006), Shanghai, China, May 20-28, 2006*. ACM, 492–501. https://doi.org/10.1145/1134285.1134355

[39] Joseph Lawrance, Christopher Bogart, Margaret M. Burnett, Rachel K. E. Bellamy, Kyle Rector, and Scott D. Fleming. 2013. How Programmers Debug, Revisited: An Information Foraging Theory Perspective. *IEEE Trans. Software Eng.* 39, 2 (2013), 197–215. https://doi.org/10.1109/TSE.2010.111

[40] Hongwei Li, Zhenchang Xing, Xin Peng, and Wenyun Zhao. 2013. What help do developers seek, when and how?. In *20th Working Conference on Reverse Engineering, WCRE 2013, Koblenz, Germany, October 14-17, 2013*. IEEE Computer Society, 142–151. https://doi.org/10.1109/WCRE.2013.6671289

[41] J. T. Liang, C. Yang, and B. A. Myers. 2024. A Large-Scale Survey on the Usability of AI Programming Assistants: Successes and Challenges. In *2024 IEEE/ACM 46th International Conference on Software Engineering (ICSE)*. IEEE Computer Society, Los Alamitos, CA, USA, 605–617. https://doi.ieeecomputersociety.org/

[42] Bing Liu. 2011. *Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data. Second Edition.* Springer. https://doi.org/10.1007/978-3-642-19460-3

[43] Walid Maalej and Martin P. Robillard. 2014. Patterns of Knowledge in API Reference Documentation. P-227 (2014), 29. https://dl.gi.de/handle/20.500.12116/30991

[44] Walid Maalej, Rebecca Tiarks, Tobias Roehm, and Rainer Koschke. 2014. On the Comprehension of Program Comprehension. *ACM Trans. Softw. Eng. Methodol.* 23, 4 (2014), 31:1–31:37. https://doi.org/10.1145/2622669

[45] Rob McCarney, James Warner, Steve Iliffe, Robbert Haselen, Mark Griffin, and Peter Fisher. 2007. The Hawthorne Effect: A Randomised, Controlled Trial. *BMC medical research methodology* 7 (02 2007), 30. https://doi.org/10.1186/1471-2288-7-30

[46] Sahar Mehrpour, Thomas D. LaToza, and Rahul K. Kindi. 2019. Active Documentation: Helping Developers Follow Design Decisions. *2019 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)* 00 (2019), 87–96. https://doi.org/10.1109/vlhcc.2019.8818816

[47] Michael Meng, Stephanie Steinhardt, and Andreas Schubert. 2018. Application Programming Interface Documentation: What Do Software Developers Want? *Journal of Technical Writing and Communication* 48 (07 2018), 295–330. https://doi.org/10.1177/0047281617721853

[48] Michael Meng, Stephanie Steinhardt, and Andreas Schubert. 2019. How developers use API documentation: an observation study. *Communication Design Quarterly Review* 7, 2 (2019), 40–49.

[49] Michael Meng, Stephanie M. Steinhardt, and Andreas Schubert. 2020. Optimizing API Documentation: Some Guidelines and Effects. In *SIGDOC '20: The 38th ACM International Conference on Design of Communication, Denton, TX, USA, October 5-9, 2020*. ACM, 24:1–24:11. https://doi.org/10.1145/3380851.3416759

[50] Michela Montesi and Trilce Navarrete. 2008. Classifying web genres in context: A case study documenting the web genres used by a software engineer. *Inf. Process. Manag.* 44, 4 (2008), 1410–1430. https://doi.org/10.1016/J.IPM.2008.02.001

[51] Varvana Myllärniemi, Sari Kujala, Mikko Raatikainen, and Piia Sevón. 2018. Development as a journey: factors supporting the adoption and use of software frameworks. *J. Softw. Eng. Res. Dev.* 6 (2018), 6. https://doi.org/10.1186/S40411-018-0050-8

[52] Alok Mysore and Philip J. Guo. 2018. Porta: Profiling Software Tutorials Using Operating-System-Wide Activity Tracing. In *The 31st Annual ACM Symposium on User Interface Software and Technology, UIST 2018, Berlin, Germany, October 14-17, 2018*. ACM, 201–212. https://doi.org/10.1145/3242587.3242633

[53] N. J. D. NAGELKERKE. 1991. A note on a general definition of the coefficient of determination. *Biometrika* 78, 3 (09 1991), 691–692. https://doi.org/10.1093/biomet/78.3.691 arXiv:https://academic.oup.com/biomet/article-pdf/78/3/691/712023/78-3-691.pdf

[54] Daye Nam, Amber Horvath, Andrew Macvean, Brad A. Myers, and Bogdan Vasilescu. 2019. MARBLE: Mining for Boilerplate Code to Identify API Usability Problems. In *34th IEEE/ACM International Conference on Automated Software Engineering, ASE 2019, San Diego, CA, USA, November 11-15, 2019*. IEEE, 615–627. https://doi.org/10.1109/ASE.2019.00063

[55] Daye Nam, Andrew Macvean, Vincent J. Hellendoorn, Bogdan Vasilescu, and Brad A. Myers. 2023. In-IDE Generation-based Information Support with a Large Language Model. *CoRR* abs/2307.08177 (2023). https://doi.org/10.48550/ARXIV.2307.08177 arXiv:2307.08177

[56] Daye Nam, Brad A. Myers, Bogdan Vasilescu, and Vincent J. Hellendoorn. 2023. Improving API Knowledge Discovery with ML: A Case Study of Comparable API Methods. In *45th IEEE/ACM International Conference on Software Engineering, ICSE 2023, Melbourne, Australia, May 14-20, 2023*. IEEE, 1890–1906. https://doi.org/10.1109/ICSE48619.2023.00161

[57] Janet Nykaza, Rhonda Messinger, Fran Boehme, Cherie L. Norman, Matthew Mace, and Manuel Gordon. 2002. What programmers really want: results of a needs assessment for SDK documentation. In *Proceedings of the 20st annual international conference on Documentation, SIGDOC 2002, Toronto, Ontario, Canada, October 20-23, 2002*. ACM, 133–141. https://doi.org/10.1145/584955.584976

[58] Stephen Oney and Joel Brandt. 2012. Codelets: linking interactive documentation and example code in the editor. In *CHI Conference on Human Factors in Computing Systems, CHI '12, Austin, TX, USA - May 05 - 10, 2012*. ACM, 2697–2706. https://doi.org/10.1145/2207676.2208664

[59] Amantia Pano, Daniel Graziotin, and Pekka Abrahamsson. 2018. Factors and actors leading to the adoption of a JavaScript framework. *Empir. Softw. Eng.* 23, 6 (2018), 3503–3534. https://doi.org/10.1007/S10664-018-9613-X

[60] Dimitrios Pierrakos, Georgios Paliouras, Christos Papatheodorou, and Constantine D. Spyropoulos. 2003. Web Usage Mining as a Tool for Personalization: A Survey. *User Model. User Adapt. Interact.* 13, 4 (2003), 311–372. https://doi.org/10.1023/A:1026238916441

[61] Peter L. T. Pirolli. 2007. *Information Foraging Theory: Adaptive Interaction with Information*. Oxford University Press. https://doi.org/10.1093/acprof:oso/9780195173321.001.0001

[62] Reinhold Plösch, Andreas Dautovic, and Matthias Saft. 2014. The Value of Software Documentation Quality. In *2014 14th International Conference on Quality Software, Allen, TX, USA, October 2-3, 2014*. IEEE, 333–342. https://doi.org/10.1109/QSIC.2014.22

[63] Christi-Anne Postava-Davignon, Candice Kamachi, Cory Clarke, Gregory Kushmerek, Mary Beth Rettger, Pete Monchamp, and Rich Ellis. 2004. Incorporating Usability Testing into the Documentation Process. *Technical Communication* 51 (02 2004), 36–44.

[64] Nikitha Rao, Chetan Bansal, Thomas Zimmermann, Ahmed Hassan Awadallah, and Nachiappan Nagappan. 2020. Analyzing Web Search Behavior for Software Engineering Tasks. In *2020 IEEE International Conference on Big Data (IEEE BigData 2020), Atlanta, GA, USA, December 10-13, 2020*. IEEE, 768–777. https://doi.org/10.1109/BIGDATA50022.2020.9378083

[65] Irum Rauf, Pekka Perälä, Jouni Huotari, and Ivan Porres. 2016. Perceived Obstacles by Novice Developers Adopting User Interface APIs and Tools. *2016 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)* (2016), 223–227. https://doi.org/10.1109/vlhcc.2016.7739689

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

[67] Martin P. Robillard and Robert DeLine. 2011. A field study of API learning obstacles. *Empir. Softw. Eng.* 16, 6 (2011), 703–732. https://doi.org/10.1007/S10664-010-9150-8

[68] Martin P. Robillard, Andrian Marcus, Christoph Treude, Gabriele Bavota, Oscar Chaparro, Neil A. Ernst, Marco Aurélio Gerosa, Michael W. Godfrey, Michele Lanza, Mario Linares Vásquez, Gail C. Murphy, Laura Moreno, David C. Shepherd, and Edmund Wong. 2017. On-demand Developer Documentation. In *2017 IEEE International Conference on Software Maintenance and Evolution, ICSME 2017, Shanghai, China, September 17-22, 2017*. IEEE Computer Society, 479–483. https://doi.org/10.1109/ICSME.2017.17

[69] Charles Romesburg. 2004. *Cluster analysis for researchers*. Lulu. com.

[70] Steven I. Ross, Fernando Martinez, Stephanie Houde, Michael J. Muller, and Justin D. Weisz. 2023. The Programmer's Assistant: Conversational Interaction with a Large Language Model for Software Development. In *Proceedings of the 28th International Conference on Intelligent User Interfaces, IUI 2023, Sydney, NSW, Australia, March 27-31, 2023*. ACM, 491–514. https://doi.org/10.1145/3581641.3584037

[71] Karen A. Schriver. 1997. *Dynamics in document design*. Wiley Computer Pub., New York.

[72] Xuehua Shen, Bin Tan, and ChengXiang Zhai. 2005. Implicit user modeling for personalized search. *Proceedings of the 14th ACM international conference on Information and knowledge management - CIKM '05* (2005), 824–831. https://doi.org/10.1145/1099554.1099747

[73] Jonathan Sillito, Gail C. Murphy, and Kris De Volder. 2008. Asking and Answering Questions during a Programming Change Task. *IEEE Trans. Software Eng.* 34, 4 (2008), 434–451. https://doi.org/10.1109/TSE.2008.26

[74] Jaideep Srivastava, Robert Cooley, Mukund Deshpande, and Pang-Ning Tan. 2000. Web Usage Mining: Discovery and Applications of Usage Patterns from Web Data. *SIGKDD Explor.* 1, 2 (2000), 12–23. https://doi.org/10.1145/846183.846188

[75] Ning Su, Jiyin He, Yiqun Liu, Min Zhang, and Shaoping Ma. 2018. User Intent, Behaviour, and Perceived Satisfaction in Product Search. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, WSDM 2018, Marina Del Rey, CA, USA, February 5-9, 2018*. ACM, 547–555. https://doi.org/10.1145/3159652.3159714

[76] Harsh Suri. 2011. Purposeful Sampling in Qualitative Research Synthesis. *Qualitative research journal* 11, 2 (2011), 63–75.

[77] Kyle Thayer, Sarah E. Chasins, and Amy J. Ko. 2021. A Theory of Robust API Knowledge. *ACM Trans. Comput. Educ.* 21, 1 (2021), 8:1–8:32. https://doi.org/10.1145/3444945

[78] Yuan Tian, Ke Zhou, and Dan Pelleg. 2023. Characterization and Prediction of Mobile Tasks. *ACM Trans. Inf. Syst.* 41, 1 (2023), 13:1–13:39. https://doi.org/10.1145/3522711

[79] Nava Tintarev and Judith Masthoff. 2012. Evaluating the effectiveness of explanations for recommender systems - Methodological issues and empirical studies on the impact of personalization. *User Model. User Adapt. Interact.* 22, 4-5 (2012), 399–439. https://doi.org/10.1007/s11257-011-9117-5

[80] Christoph Treude and Martin P Robillard. 2016. Augmenting API documentation with insights from stack overflow. *ICSE* (2016). https://doi.org/10.1145/2890000/2884800/p392-treude.pdf

[81] Gias Uddin and Martin P. Robillard. 2015. How API Documentation Fails. *IEEE Softw.* 32, 4 (2015), 68–75. https://doi.org/10.1109/MS.2014.80

[82] Enrique Larios Vargas, Maurício Finavaro Aniche, Christoph Treude, Magiel Bruntink, and Georgios Gousios. 2020. Selecting third-party libraries: the practitioners' perspective. In *ESEC/FSE '20: 28th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering, Virtual Event, USA, November 8-13, 2020*. ACM, 245–256. https://doi.org/10.1145/3368089.3409711

[83] Marcello Visconti and Curtis R. Cook. 2004. Assessing the State of Software Documentation Practices. In *Product Focused Software Process Improvement, 5th International Conference, PROFES 2004, Kausai Science City, Japan, April 5-8, 2004, Proceedings (Lecture Notes in Computer Science, Vol. 3009)*. Springer, 485–496. https://doi.org/10.1007/978-3-540-24659-6_35

[84] Alexandra Wood, Micah Altman, Aaron Bembenek, Mark Bun, Marco Gaboardi, James Honaker, Kobbi Nissim, David R O'Brien, Thomas Steinke, and Salil Vadhan. 2018. Differential Privacy: A Primer for a Non-Technical Audience. *Vanderbilt journal of entertainment and technology law* 21, 1 (2018), 209–.

[85] Songhua Xu, Yi Zhu, Hao Jiang, and Francis C. M. Lau. 2008. A User-Oriented Webpage Ranking Algorithm Based on User Attention Time. In *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence, AAAI 2008, Chicago, Illinois, USA, July 13-17, 2008*. AAAI Press, 1255–1260. http://www.aaai.org/Library/AAAI/2008/aaai08-199.php

[86] Xing Yi, Liangjie Hong, Erheng Zhong, Nanthan Nan Liu, and Suju Rajan. 2014. Beyond Clicks: Dwell Time for Personalization. In *Proceedings of the 8th ACM Conference on Recommender Systems* (Foster City, Silicon Valley, California, USA) *(RecSys '14)*. Association for Computing Machinery, New York, NY, USA, 113–120. https://doi.org/10.1145/2645710.2645724

[87] Peifeng Yin, Ping Luo, Wang-Chien Lee, and Min Wang. 2013. Silence is also evidence: interpreting dwell time for recommendation from psychological perspective. In *The 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2013, Chicago, IL, USA, August 11-14, 2013*. ACM, 989–997. https://doi.org/10.1145/2487575.2487663

[88] J. D. Zamfirescu-Pereira, Richmond Y. Wong, Bjoern Hartmann, and Qian Yang. 2023. Why Johnny Can't Prompt: How Non-AI Experts Try (and Fail) to Design LLM Prompts. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems, CHI 2023, Hamburg, Germany, April 23-28, 2023*. ACM, 437:1–437:21. https://doi.org/10.1145/3544548.3581388

[89] Tianyi Zhang, Björn Hartmann, Miryung Kim, and Elena L. Glassman. 2020. Enabling Data-Driven API Design with Community Usage Data: A Need-Finding Study. In *CHI '20: CHI Conference on Human Factors in Computing Systems, Honolulu, HI, USA, April 25-30, 2020*. ACM, 1–13. https://doi.org/10.1145/3313831.3376382

[90] Sha Zhao, Julian Ramos, Jianrong Tao, Ziwen Jiang, Shijian Li, Zhaohui Wu, Gang Pan, and Anind K. Dey. 2016. Discovering different kinds of smartphone users through their application usage behaviors. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing, UbiComp 2016, Heidelberg, Germany, September 12-16, 2016*. ACM, 498–509. https://doi.org/10.1145/2971648.2971696