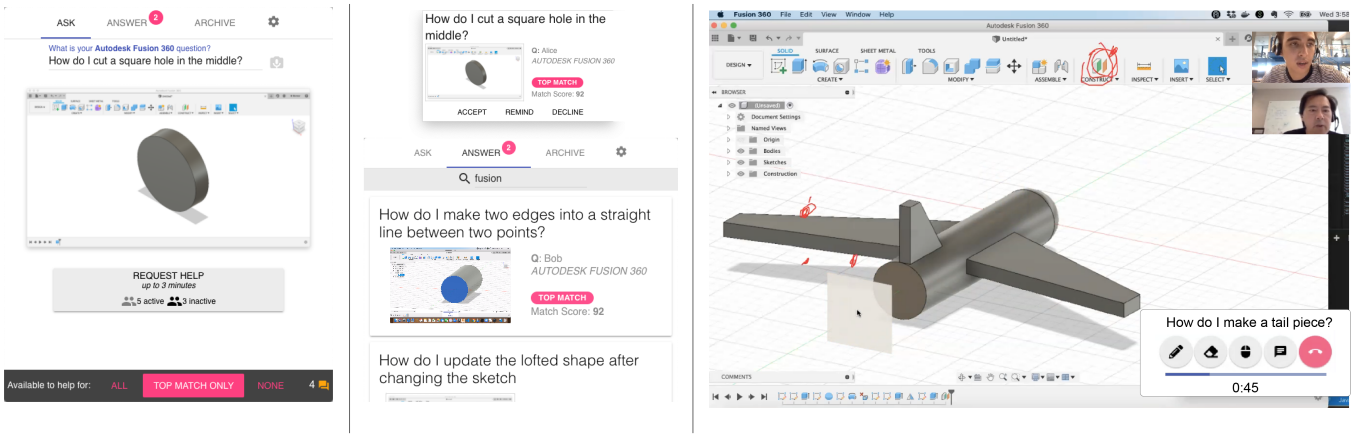
MicroMentor: Peer-to-Peer Software Help Sessions in Three Minutes or Less

Nikhita Joshi[⁂†], Justin Matejka[⁂], Fraser Anderson[⁂], Tovi Grossman[⁂‡], George Fitzmaurice[⁂]

[⁂]Autodesk Research
Toronto, ON, Canada
{first.last}@autodesk.com

[†]University of Waterloo
Waterloo, ON, Canada
nvjoshi@uwaterloo.ca

[‡]University of Toronto
Toronto, ON, Canada
tovi@dgp.toronto.edu



Submit a Help Request **Answer Community Questions** **Connect with a Helper for 3 Minute Help Sessions**

Figure 1. MicroMentor system enabling rapid help via short 1-on-1 help sessions.

ABSTRACT

While synchronous one-on-one help for software learning is rich and valuable, it can be difficult to find and connect with someone who can provide assistance. Through a formative user study, we explore the idea of fixed-duration, one-on-one help sessions and find that 3 minutes is often enough time for novice users to explain their problem and receive meaningful help from an expert. To facilitate this type of interaction, we developed *MicroMentor*, an on-demand help system that connects users via video chat for 3-minute help sessions. MicroMentor automatically attaches relevant supplementary materials and uses contextual information, such as command history and expertise, to encourage the most qualified users to accept incoming requests. These help sessions are recorded and archived, building a bank of knowledge that can further help a broader audience. Through a user study, we find MicroMentor to be useful and successful in connecting users for short teaching moments.

Author Keywords

Software learning; quick help; one-on-one help; mentoring

CHI '20, April 25–30, 2020, Honolulu, HI, USA

ACM 978-1-4503-6708-0/20/04…$15.00
https://doi.org/10.1145/3313831.3376230

INTRODUCTION

Feature-rich software can be difficult to learn due to complex user interfaces, number of tools available, and unique vocabularies [20, 29]. Many resources are available online such as videos and blog posts, but the best support is often provided by other users [10]. "Over-the-shoulder" learning (informal help between colleagues in a workplace) is one of the most popular [41] and preferred [25, 32] learning strategies. Directly speaking to a colleague allows for more targeted questions, shared context [5, 41], and incidental learning [5]. Despite its popularity, 1-on-1 learning rarely occurs outside formal learning environments [12].

Leveraging the benefits of peer assistance, community question and answer (CQA) sites have become popular help seeking tools [13, 33]. StackOverflow, for example, encourages users to ask concise questions that could be asked to a "busy colleague" in one sentence [43]. However, many users have trouble formulating targeted questions [10, 29]. Users often omit important background information [4, 26] and supplementary materials like screenshots [11], making it difficult for other users to respond.

A few paid systems, like Codementor [14], mimic over-the-shoulder learning more closely by connecting users to remote experts who then collaborate with one another via video chat and a shared code editor. However, there is a coordination cost associated with finding the right helper and explaining the problem. Even when one-on-one help is available, people believe an expert's time is valuable and may be reluctant to ask for help and will even modify their working habits to minimize the impact on the expert's time [5]. In summary,

existing remote video assistance is useful, but has a high transaction cost which prevents its widespread adoption.

We believe there is an opportunity to better utilize one-on-one help mechanisms outside of formal learning environments, while minimizing the burden for both the asker and helper. With the growth of social media, we have become accustomed to consuming small 'tidbits' of information, and with the growth of the 'gig economy' we are accustomed to providing others with easy-to-access short-term services. We believe there is a novel opportunity to leverage these trends to support one-on-one software help through easy-to-access micro-mentoring sessions.

In this paper, we present *MicroMentor* (Figure 1), an on-demand help system that automatically connects question askers with helpers for rapid one-on-one help sessions. The system's design is guided by an initial formative study, which revealed that 3 minutes is often more than enough time for novice users to explain their problem and receive a meaningful answer from an expert. MicroMentor includes several elements which facilitate the short help request, including the addition of contextual information, and intelligent matching of askers and helpers. Archiving the help sessions enables future users to learn from each short mentoring session. Through an evaluation, we find that MicroMentor is useful in solving many questions, and may be comparable to in-person, expert assistance. Our work makes the following three contributions:

1) A formative study revealing challenges and opportunities of using micro-help sessions for software learning.
2) The development of MicroMentor, a novel system that minimizes the transaction cost of remote 1-1 help through automatic recruitment and time-constrained sessions.
3) An empirical evaluation showing the promise of MicroMentor and revealing insights and guidance for future software learning research.

BACKGROUND AND RELATED WORK

Our work builds upon help seeking behaviours and systems, and behavioural theory.

Help Seeking Behaviours

Previous work highlights the challenges users face when accessing help resources. Often, users do not understand application-specific vocabulary [16] making it difficult for users to form relevant search queries [10, 29]. Explaining specific problems in plain language [1, 10], within the target application [33] may help users form meaningful questions. Kiani et al. [29] study software newcomers' help seeking behaviours, and note that most participants preferred videos and some expressed a need for shorter videos. Video archives generated from quick help sessions may address this issue.

Remote video assistance is a common approach for providing technical support [17]. However, Chilana et al.'s study of product support practices [11] found that support specialists had mixed feelings about screen sharing. The advantage of sharing screens was that it maximized "shared understanding" [17]. The main drawback reported by specialists was that sharing sessions took too much of their time. With MicroMentor, we look to leverage the benefits of video screen-sharing, while mitigating this transaction cost.

The best support is often provided by members of the community [10]. CQA sites have become popular in recent years. However, remote helpers have little background knowledge of the asker's problem, necessitating askers to include the right contextual information [10]. A lack of background information may cause posts to go unanswered [4] or prompt clarification requests [26] that take time to answer. Automatically capturing and attaching supplementary content may mitigate these challenges.

When possible, people often prefer asking their colleagues directly for help [25, 32]. "Over-the-shoulder" learning between colleagues in the workplace, is one of the most popular help seeking strategies [41] and offers many benefits. It is convenient to initiate in the moment, provides rich shared context, and allows for more targeted queries and follow-up questions [5, 41]. Directly observing a helper's actions results in more incidental learning as conversations can shift and span many domains [5].

One-on-one help is rarely available outside the workplace [12]. Even when it is available, people are reluctant to use it. Finding the right helper and coordinating a meeting time may be challenging [10]. People may feel embarrassed to ask questions or reluctant to disturb other people [41]. Berlin and Jeffries [5] found mentees made an effort to minimize their impact on a mentor's time. They spent time narrowing the problem space before going to the mentor, sent emails to be less intrusive, and began working on other tasks when they got stuck; building a set of questions to ask their mentor at a later time. Our goal is to support short help sessions which may be less of a burden on the helper's time and reduce the asker's feelings of guilt when requesting help.

Systems for Seeking Help

Numerous interactive systems have been developed in the HCI literature to address the aforementioned challenges. Codementor [14] connects developers for long term mentoring or one-off help sessions. The asker submits a request, selects a helper, and schedules a meeting time. The asker and helper collaborate with one another using a shared code editor. If the helper does not provide a good solution, the asker must find and schedule time with another helper. MicroMentor lowers the transaction cost by automating the helper selection process and the resubmission of requests.

Previous systems have focused on providing additional context when asking questions. Codeon [9] allows developers to describe their problem in plain language within their IDE and automatically attaches relevant code context to help requests. LemonAid [13] users can ask for help by selecting UI widgets of interest. IP-QAT [33] encourages askers to submit relevant commands and screenshots. MicroMentor encourages askers to submit screen recordings,

allowing the user to highlight areas of interest using their cursor and describe problems in plain language.

Some systems have focused on routing questions to online experts capable of responding [24, 36] which can improve response times [40]. Answer Garden [1, 2] stores previously answered questions, and routes unanswered questions to experts. This ensures experts are answering useful questions over duplicates. Real-time collaborative tools [18, 21] can facilitate shared learning and work, and teaching opportunities between peers. MicroMentor could be used to promote collaborative learning and shared work, but we focus on teaching opportunities.

Behavioural Theory

Carroll's work [7] suggests a minimalist approach to instruction may be more effective and notes the mental costs of acquiring, reading, and organizing written information. His "Paradox of the Active User" [8] describes a production bias, where users are motivated by throughput and are likely to use known, but inefficient methods to accomplish a task over learning new approaches. As such, many of today's help systems have been designed to minimize the transaction cost of learning new content by providing help in context [3, 30] and presenting information in small chunks. For example, early research showed limitations of long video tutorials [22, 39] while short contextual videos can be effective [19].

An important example of reducing transaction costs is the recent phenomena of micro-blogging. Most notably, Twitter is a popular social media system which enforces a 280-character count (originally 140 characters) for all posts. Jave et al. discussed how such enforced constraints may have resulted in the platform's immediate growth [28]:

"Microblogging fulfills a need for an even faster mode of communication. By encouraging shorter posts, it lowers users' requirement of time and thought investment for content generation"

This idea has support within the HCI community, with researchers advocating for all requests to be made through constrained communication channels to facilitate concise requests and lower the expectation of a lengthy response [44]. MicroMentor builds on this idea of length constraints by enforcing a 3-minute time interval on 1-1 help sessions.

Many theories on ambiguity and risk aversion, like the Ellsberg paradox [15], state people prefer to bet on specific, known odds over ambiguous probabilities, even when the known odds are low [6]. Simply knowing more information can have a positive impact, especially when it involves time. As stated by Maria Konnikova in The New Yorker [31]:

"The more we know about something—including precisely how much time it will consume—the greater the chance we will commit to it."

To-do lists with time estimates [42] can help overcome procrastination. Many written articles include estimated read times [37] to increase engagement [23]. We believe there is a unique opportunity here; traditional one-on-one help has no definite end time, and ambiguous durations may deter experts from helping their peers.

MicroMentor is inspired by the strengths and weaknesses of different help seeking strategies and systems. Previous systems have attempted to improve submitting a request, receiving a response, and accessing it later. MicroMentor addresses similar issues, but focuses on synchronous, one-on-one help. We design MicroMentor with quick help in mind and use this idea to drive design decisions.

FORMATIVE STUDY – UNDERSTANDING QUICK HELP

A formative study was conducted to better understand the effectiveness, opportunities and challenges of rapid one-on-one software help sessions. In particular, the study was designed to explore the effects an enforced time-limit on help sessions would have on the outcome and experience of the asker and helper. We tested 1-minute and 3-minute help sessions and compared these timed conditions to a baseline with no time constraints. One minute is enough time for users to perform simple tasks and 3 minutes reflects the average duration of most website visits [45]. In the study, the askers were attempting to complete a task within Fusion 360, a complex 3D design software application. Help sessions were conducted in-person, as our goal was to understand the impact of duration on the best-case learning scenario.

Participants and Apparatus

We recruited 6 participants. Three helpers were Fusion 360 experts, ages 22 to 30 (M = 27, SD = 4.6). All helpers were male and reported using Fusion 360 often in their work. Three askers, ages 26 to 32 (M = 30, SD = 3.5) self-identified as Fusion 360 novices. Two askers were male, 1 was female. Remuneration was a $50 gift card. All askers worked at a desk with a laptop and mouse in an individual room while all helpers sat in a common waiting area.

Procedure

The study took place in a single group session with all 6 participants at the same time. Askers were told to design a car in Fusion 360 and that they could call for a helper when they had questions. Each asker was provided a collection of car images for inspiration, but not a specific task or workflow they needed to follow to encourage a variety of questions.

Calling for help was the askers' only means of assistance (they could not look for help resources in the software, or on the internet). However, once a helper was present, they could use other help resources if necessary. We instructed askers to try to ask no fewer than 9 questions, and up to 18 questions (1 question every 5 minutes). This would allow every asker to interact with every helper for all 3 time limit conditions (*1 minute, 3 minutes, no time limit*). A facilitator encouraged askers to think of a question they could ask a helper if several minutes passed since their last question.

When the asker wanted to ask a question, a facilitator assigned a helper and time limit to the asker, using a predetermined randomized order. If the assigned helper was busy assisting another asker, another helper/time limit

combination was selected based on the next available helper in the list. The helper would then enter the asker's room, and a timer recorded the time it took the asker to ask their question and receive assistance. To better capture the time needed to receive help, we told the askers to dismiss the helpers once their question was answered, or the helpers were dismissed by the facilitator once the allocated time had elapsed. After each help session, we asked both parties to complete short questionnaires about their experience, including if they thought the question had been answered. The entire session lasted 90 minutes. We then conducted semi-structured interviews with each participant.

Results

Help Session Durations

Overall, 37 questions were asked in total (Figure 2). The shortest session was 17s and the longest was over 15m (902s). The average session was just over 2m (138s), and it took askers 16s on average to ask their question.

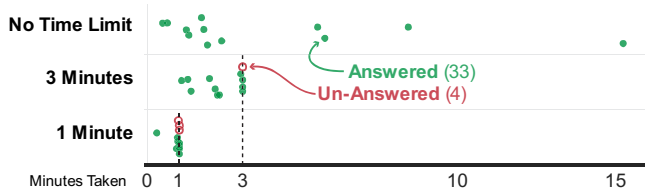


Figure 2. Question times in the formative study, many were successfully answered (rated 4 or 5) in 1 and 3 minutes.

Of the 37 questions, 13 were assigned a 1-minute time limit, twelve were assigned a 3-minute interval, and twelve questions had no time limit. For the 1-minute interval, the shortest session was 17s, and the average was 57s. Only 2/13 (15%) sessions were successfully answered under 60s. For the 3-minutes time limit, the shortest help session was 64s and the average was a little over 2 minutes (136s). Over two-thirds of the questions (8/12, 67%) were successfully answered in less than 3 minutes. For the questions with no time-limits, the shortest help session was 28s, the longest was 902s, and the average help session was slightly under 4 minutes (228s). Questions that were considered to have been answered (asker gave a rating of 4 or 5) in the no time-limit condition were answered under 3 minutes (166s) on average.

Stress

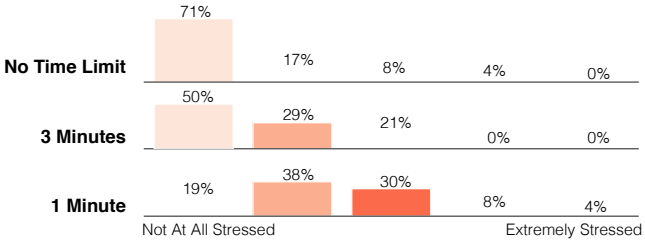


Figure 3. Reported stress levels for each time limit, note that 3 minutes has only slightly higher stress than no limit.

After every help request, askers and helpers indicated their level of stress on a 5-point Likert scale (Figure 3). Only 19% of all responses indicated no feelings of stress for the 1-minute condition, but this number is much higher for the 3-minutes (50%) and no time limit (71%) conditions.

Understanding the Question

Helpers indicated how well they understood the question after every help session using a 5-point Likert scale. Overall, helpers felt they understood all but 2 questions (95%).

Succesfully Answered Questions

After every help session, askers indicated whether the question was answered successfully. Askers felt questions were answered 92% of the time with no time limit, and 83% with 3 minutes. With 1 minute, askers felt their questions were answered only 39% of the time (Figure 4).

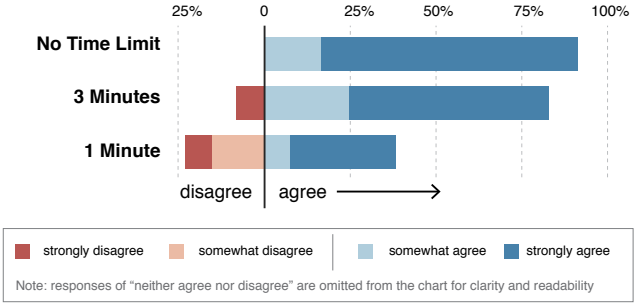


Figure 4: Responses to whether or not the asker felt their questions were successfully answered.

Interviews

Asking a Question

A few participants noted using different strategies for each time limit. With limited time, there was more pressure to ask focused, targeted questions with little elaboration, which was challenging. However, helpers appreciated and preferred these concise questions over longer questions. With no time limit, askers framed their help requests as discussions about a problem, but noted the inefficiencies of too many side conversations and follow-up questions. Helpers often asked clarifying questions. When there was only 1 minute, clarifying often took too much time, leaving little time for a response. Non-verbal cues, like pointing and making hand gestures, were useful when describing a problem or clarifying. For all three conditions, helpers overall felt they understood the questions, which may in part, be due to non-verbal cues. Many participants felt 3 minutes was optimal; there was enough time to ask a question while providing details, clarify, and receive a focused response.

Receiving Help

Like the askers, the helpers used different strategies when giving help. With 1 minute, helpers felt there was no time to fully walk through a solution and instead help was framed as concise, step-by-step instructions. When there were multiple ways to solve a problem, helpers often optimized for time by showing only one, often the easiest, way to solve the problem. This let helpers get to the point, but many felt they could not fully show the asker how to solve the problem. When there was less time remaining, helpers were tempted to take control of the asker's mouse, noting it was easier to show rather than explain. More time allowed helpers to give

context and thorough explanations. In addition, helpers could try different approaches and walk through entire solutions with the asker. As a result, 1 minute may be too short to receive quality help, but 3 minutes is likely enough time for many types of questions.

Participants agreed 1 minute was enough time for easy questions, such as locating a button. For complicated questions, 1 minute was not enough time and considered stressful (Figure 3). Some participants felt 1 minute help sessions would be less stressful with more practice. Most participants felt 3 minutes was more than enough time, even for trickier questions. Even if the question was not fully answered, askers felt they had received enough information to figure out the rest independently. Three minute sessions were not stressful. Having no time limit was not stressful and appropriate for complex tasks, but askers felt it was "unnecessary" or felt guilty using the helper's time.

Other Challenges and Opportunities

Some participants noted the value of having a pool of helpers available to tackle problems sooner and learn new skills. One helper suggested establishing a community-based triage approach to route incoming requests, noting the annoyances of answering the same question multiple times throughout the work day. Routing questions to other helpers could reduce this. One asker felt pairing with a helper capable of responding was more important than the duration itself, and pairing with the wrong helper may cause unnecessary stress.

Overall, the results presented in this study provide valuable insights into the concept of quick, bounded help sessions, and indicate the promise of systems that can enable rapid, time constrained, help sessions. The results suggest 3 minutes is a good candidate for quick help sessions. It was enough time for askers to ask a focused question while still allowing for additional details and clarifications. Three minutes was not stressful, and over 80% of questions, of varying difficulty, were successfully answered. This is not far off from the "best case scenario" of having no time limit (92%). Even if a question was not fully answered, askers felt they had received enough information to resolve the problem on their own. This may pair well with people's natural tendency to solve problems independently [8, 38].

DESIGN GUIDELINES

The results of our formative study, along with relevant related work, led us to develop a series of design guidelines.

Bounded Time Commitment *(time)*

A quick help system should be mindful of a helper's time and ensure all help requests end after 3 minutes. We feel a time limit will encourage more participation from helpers and askers by removing ambiguity around help durations [15] and reducing feelings of guilt for taking someone's time [5]. Our formative study showed askers felt the need to ask targeted questions with a time limit, but some helpers preferred this over longer questions. Thus, the success of 'quick help' may also rely on 'quick questions' that are easy to explain, understand, and clarify. Therefore, mechanisms could also be provided encourage question askers to constrain the duration of their questions.

Lower Transaction Costs *(low-cost)*

Requesting, receiving, and providing help should be as easy as possible to encourage more participation from the community. The idea of 'quick help' should extend beyond the help session itself, and be integrated within all interactions with the system. The system should automate all major steps including, but not limited to, attaching supplementary materials, finding the best mentor, initializing a help session, and saving archives.

Support Easy Reuse *(reuse)*

Some helpers noted the frustrations of answering the same questions multiple times, an issue which has been investigated in past work [1, 2]. While our system should encourage short help sessions, askers may still submit duplicates. To address this, the system should archive and make previously answered questions available and searchable for other users to view.

Provide Context *(context)*

Previous work highlights the importance of context when submitting a help request [10], and many systems have been designed to automatically capture contextual information [9, 13, 33]. The system should leverage and communicate as much contextual information between the asker and helper as possible. This may help askers formulate targeted questions [33], and assist the helper in understanding the problem and their ability to resolve the problem. This contextual information should also be leveraged when users are trying to find archived requests.

MICROMENTOR

Building on previous work, results from our formative study, and our design guidelines, we developed MicroMentor, an on-demand help system that enables peers to rapidly connect with one another to solve problems and answer questions. When describing the system, we refer to the associated design guidelines in parentheses.

Submitting a Request

MicroMentor is designed to occupy minimal screen foot-print, and to always be available to facilitate a help session. There are two ways to submit a request (Figure 5).

A small button is positioned on the bottom right corner of the user's screen. Clicking this button starts a recording of the user's screen and audio, allowing one to explain problems in plain language [10]. The recording automatically ends after 16s (*time*) or when the user presses the "Done" button. This recording mode can also be accessed in MicroMentor's "Ask" panel. MicroMentor automatically transcribes the captured audio using Microsoft Azure speech services to save time in typing (*low-cost*). However, users may not want to share their screen or record themselves talking, possibly due to confidence issues or language barriers [10], so the system also allows users to type their question if necessary.

MicroMentor automatically captures and attaches a screenshot instead of a video. The screenshots and screen recordings are automatically submitted with the help request (*context, low-cost*), along with the user's last 5 commands used. Clicking the "Request Help" button sends the request to all available helpers. The number of active users is visualized in the "Help Request" button.

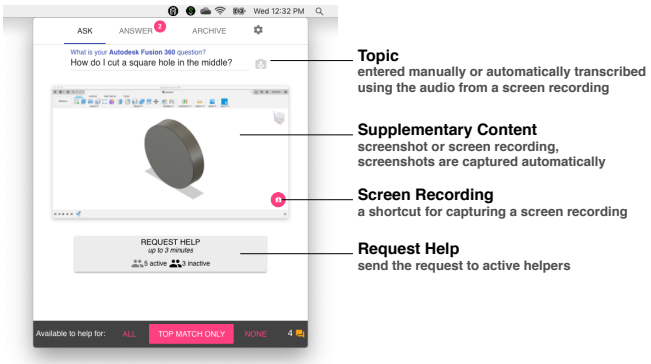


Figure 5: The view the asker sees when submitting a request, showing the topic and supplementary content.

Mentor Matching

Synchronous, one-on-one help can be difficult to coordinate [9, 10]. Furthermore, related work [10] and results from our formative study suggest getting paired with the wrong helper can cause stress. We believe helpers need to have the right background to engage in efficient, quick help sessions. MicroMentor uses contextual information (*context*) to create an ordinal rank of potential helpers. MicroMentor uses command history, expertise, the time since the last help session, previous interactions with a helper, overall average rating, additional skills, and the additional skills' relevance to the question topic to assign each helper a "match score." Some factors (e.g. time since last session, command history) were mentioned during follow-up interviews from the formative study, and others (e.g. average ratings, expertise) were inspired by previous systems like Codementor [14].

Command History – Users with similar command histories may try to perform similar tasks [35]. MicroMentor uses a potential helper's command frequencies to assess how often they used the asker's most recent commands and prioritizes helpers with the most experience using the same commands.

Expertise – Level of expertise (novice, intermediate, advanced) can be selected (Figure 6). MicroMentor compares the asker's expertise to that of the potential helper to encourage users more or equally experienced to assist.

Time Since Last Help Session – Helpers could be overwhelmed with multiple help requests in a short period of time. To mitigate this, MicroMentor places more emphasis on helpers who have not answered a help request in a while.

Previous Interactions with a Helper – Askers may enjoy receiving help from someone they recognize. If a potential helper previously assisted the asker, they are prioritized.

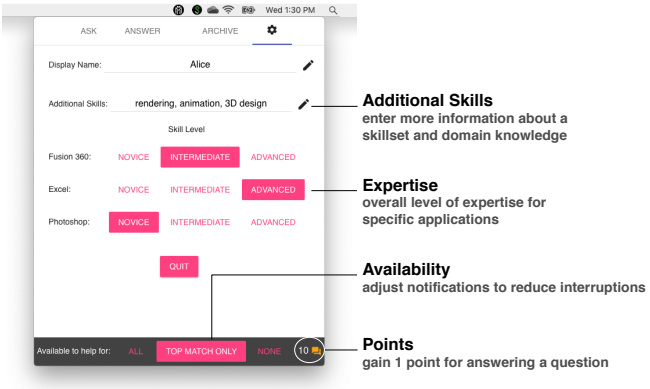


Figure 6: Settings page where users can specify their expertise, availability, and additional skills.

Average Rating – Some helpers may be better than others. MicroMentor uses the potential helper's average star rating to filter unsuccessful helpers.

Additional Skills – In the "Settings" tab (Figure 6), users can note additional skills they have, or any focus areas for a specific application. Like command history, users with similar skills may be better suited to help.

Relevance of Additional Skills – Some users may note domain-specific knowledge that cannot be captured through command history. MicroMentor compares the request's topic to the helper's additional skills to capture this information, encouraging those with domain-specific knowledge to help those who mention it in their requests.

MicroMentor uses these factors to generate a match score. The system sorts all helpers by match score and uses a triaged approach to encourage those with the top match score to receive and respond to a help request first; MicroMentor automatically sends a notification to a distinct user (*low-cost*) every 10s using the sorted order until a request is accepted.

Receiving a Request

When the mentor matching has completed, MicroMentor notifies potential helpers in two ways (Figure 7). Depending on the user's notification preferences, MicroMentor sends push notifications to potential helpers. Helpers can click "Accept" to launch a video conference, or "Decline" to dismiss the notification. Clicking "Remind" will dismiss the notification and re-display it 10s later if no one else accepted the request. Users can also browse open requests in the "Answer" tab. Clicking a card launches a video conference.

Both approaches display the same information: the request topic, the asker's name, and the target application. MicroMentor also displays the potential helper's match score and shows a pink "Top Match" badge if they are the best person to answer. All requests show supplementary materials (screenshot or screen recording) on the left (*context*). If the asker attached a video, the helper can hover over it with their cursor to unmute the audio. Users can search for open requests in this page, using all information (*context*).

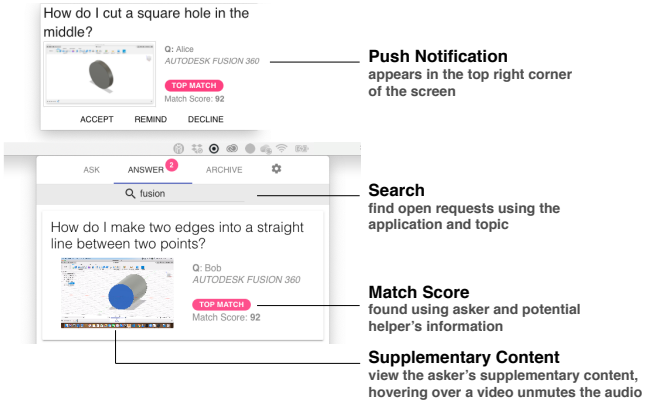


Figure 7. Receiving a help request in two ways, through a push notification, or by browsing a list of open questions.

MicroMentor allows users to specify their notification preferences (Figure 6). Selecting "All" will notify the user about all incoming help requests and selecting "None" will prevent any notifications from appearing on the user's screen. To reduce the number of notifications, the user can select "Top Match Only" to only be notified when they are the top match for a specific request.

Help Sessions

Accepting a help request automatically starts a video conference (*low-cost*). MicroMentor will notify the asker that their request has been answered, and automatically adds the asker to the video conference after 5s, giving them some warning the session is about to begin (*time*). The asker can click "Start" to join the meeting early.

When the asker joins the session, MicroMentor automatically shares their desktop and starts webcam video to allow for a shared visual context (*context*) and non-verbal cues, like hand gestures. Both users can see a MicroMentor toolbar (Figure 8). MicroMentor supports basic annotation features to enhance shared context and to mimic over-the-shoulder cues, like pointing, that would be difficult for remote users to perform (*context*). The asker can give control to the helper, allowing the helper to directly manipulate the software interface. The toolbar also shows the question and remaining time. A countdown turns red when 30s are left, and flashes with 10s left. The meeting can be ended by either users, or automatically when 3 minutes have passed (*time*). The helper earns a 'point' (Figure 6) for responding.

Once a help session ends, MicroMentor polls the asker and helper to give a satisfaction score out of 5 stars. If the asker gives a rating of 3 or less, MicroMentor will ask if they want to resubmit their help request. Clicking "Yes" will resend the request to all other helpers, requiring little action from the user (*low-cost*). The helper receiving the low rating will not receive a push notification for this new request.

Archives

MicroMentor automatically records all help sessions, automatically capturing and transcribing the users' speech

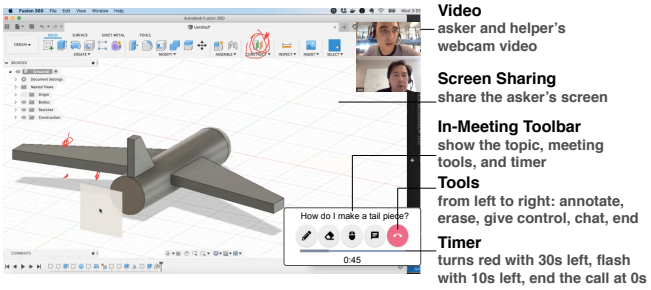


Figure 8. A live help session with active video chat, screen sharing, annotation tools, and a countdown timer.

and commands used during the session. This information is used to create public archives once the help session ends (*low-cost*, *reuse*). Archives are accessible through the "Archive" tab (Figure 9) and are similar to the open request cards in the "Answer" panel. However, MicroMentor includes some additional information, such as the helper's name, commands used during the session (*context*), the session duration, and a checkbox in the bottom right corner if the asker was satisfied (rating of 4 or 5) with the response. Users can search for archives using all request information.

Clicking a card opens a detailed archive view (Figure 9). In this view, users can watch a help session. Using the speech and command data, MicroMentor creates a clickable transcript, allowing users to navigate to relevant points in the video by selecting commands or utterances of interest (*context, low-cost*). Clicking the asker or helper's name opens the Mail app, allowing the user to ask follow-up questions as needed. Users can also share a link to the video to other users by clicking the email icon.

Implementation

MicroMentor was developed using the Electron framework. Every computer running MicroMentor is a client of a custom WebSocket server. The server manages and broadcasts incoming help requests to all active MicroMentor users. A shared Dropbox folder is used to synchronize videos and images across every user. MicroMentor uses the Zoom Electron SDK for all video conferences.

Extracting Command Histories

The system tracks all commands issued by the user for most software applications using a Python module [27] that uses the native OSX accessibility API. MicroMentor tracks when users click on different GUI elements. These GUI elements often have names that correspond to the commands they trigger; clicking the "Pivot Table" button in Excel, for example, would expose a button with the name "Pivot Table" to the accessibility API. MicroMentor takes advantage of these names to create human-readable command histories. Fusion 360 does not route user commands to the built-in accessibility framework, so MicroMentor uses a custom Python script to log all Fusion 360 commands to a text file. Note that MicroMentor does not require command histories to work, but the presence of command history adds a richer form of context to each help request.

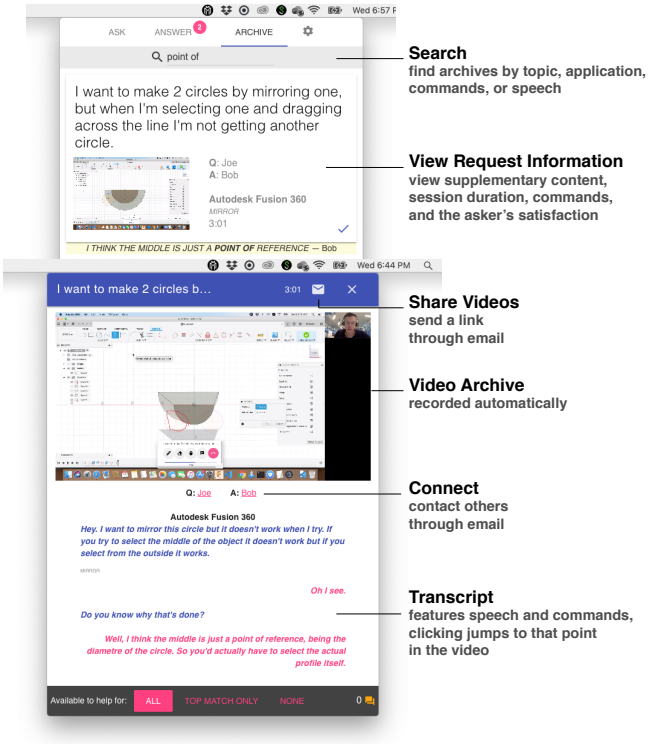


Figure 9. Archive view, where users can search through past help sessions, read the transcript, connect with the helper and share the video with others.

EVALUATION

Askers in the formative study could not access help resources online to encourage more help sessions between askers and helpers. However, that does not represent a realistic scenario; in practice, MicroMentor users would also have access to various online resources, like videos and blog posts. As such, we compare MicroMentor to a realistic baseline condition: using traditional, web-based help. Other baseline conditions were considered, such as in-person help, but we felt online resources made for a better comparisons since MicroMentor and online resources would be always-available (but in-person help may be difficult to find).

The goal of the evaluation is to elicit qualitative feedback from participants to further validate the concept of quick, on-demand video-based help. Quantitative comparisons are made where applicable, but the baseline is mainly included to provide participants a point of reference when discussing MicroMentor.

Fusion 360 was the target application as it is representative of modern, feature-rich software. Mentor matching was disabled, so helpers would have equal opportunity to answer different types of questions. Evaluating mentor matching is left for future work when the full data may be available for a large population of helpers and askers.

Participants and Apparatus

We recruited 12 participants. Four helpers were Fusion 360 experts, ages 22 to 30 (M = 28, SD = 4). All helpers were male and reported using Fusion 360 sometimes or often in their work. Three of the helpers had also took part in our formative study. Eight askers, ages 21 to 57 (M = 37, SD = 12.4) self-identified as Fusion 360 novices. Five were male, 3 were female. Remuneration was a $50 gift card, plus travel expenses where applicable. Everyone worked at a desk with a laptop with a mouse. Askers sat in individual rooms while all helpers sat in a common waiting area.

Procedure

The study was conducted as two consecutive within-subjects group sessions consisting of 4 helpers, 4 askers, and 2 conditions. Both conditions required askers to work on open-ended design tasks (design a boat or plane). Askers were provided a series of boat and plane images as inspirations. The design task was counterbalanced within each session and the condition order was also counterbalanced across both group sessions. The two conditions were:

1) **Independent help** – askers worked independently on their design task. When they needed help or had a question, they could access any online resource.
2) **MicroMentor** – askers worked on the next design task and could only seek help using MicroMentor. Helpers worked on any design task they were comfortable with. As they received notifications for incoming MicroMentor requests, helpers could accept them to start a help session.

Each condition lasted 45 minutes. Askers were instructed to try to ask no fewer than 5 questions, and up to 9 questions (1 question every 5 minutes) to ensure every participant could try using MicroMentor multiple times. After completing both group sessions, participants completed a survey.

Results

Participants took part in 40 help sessions using MicroMentor. During the baseline condition (online resources), askers made 65 help-seeking attempts. This number is greater than that of MicroMentor, but askers often accessed the same resources multiple times. For example, watching a few seconds of a video tutorial, applying this knowledge in Fusion, before returning to the same video. It is unclear whether these help-seeking attempts were useful. Data from 12 sessions was discarded due to technical errors. As such, the results from 28 help sessions are presented.

Help Session Duration

It took between 7s and 91s (28s on average) for a request to be accepted by a helper. The asker joined the video conference roughly 5s later. The shortest help session was 33s and the average session lasted 131s. Overall, the duration of the average request was 164s (Figure 10). Half of the

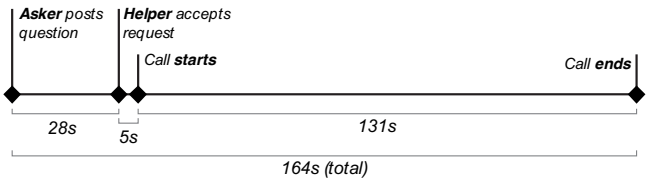


Figure 10: Timeline of an average help session, from the time posting the question to when the call ends.

requests had video submissions (9s on average). Considering these requests, it took 175s on average from the time askers began recording their question to receiving a response.

Satisfaction with Answer

After every help session, the asker and helper gave satisfaction scores out of 5 stars. Due to a technical issue, one session's ratings were lost, so we look at the results from 27 help sessions (Figure 11). Overall, askers were satisfied with the help they received using MicroMentor. Twenty-three out of 27 requests (85%) received a 4 or 5 star rating. Likewise, the helpers were satisfied with the help they delivered and gave 5 star ratings for 21 (78%) requests. This was confirmed in the participants' responses in the follow-up survey.

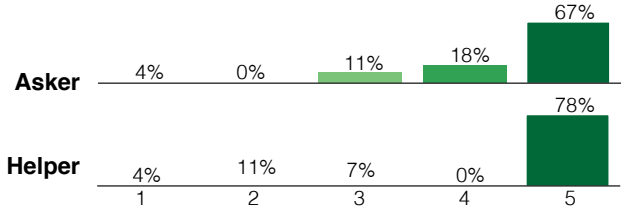


Figure 11. Asker and helper's ratings (out of 5 stars)

Survey Results

Askers and helpers answered the following questions using a 7-point Likert scale.

Ease of use – Askers and helpers noted how easy MicroMentor was to use for seeking or giving help; 1 being very difficult and 7 being very easy. Everyone gave ratings of 5 or more and 9 participants (75%) gave a rating of 6 or 7.

Frequency of use – Many participants felt they would use MicroMentor frequently. Seven stated they would use it a few times a week, 1 helper stated several times a day, and 1 asker stated several times an hour.

Subjective Feedback

Every asker felt they learned something using MicroMentor and many participants were satisfied with the help requests (Figure 11). Participants felt 3 minutes was enough time for most problems, like task-based questions, and many noted the value of keeping sessions focused.

> *"[MicroMentor] forces you to get to the crux of a problem ASAP"* – Helper 1

Some participants felt that speaking with a helper was more engaging and helped them remember more information:

> *"I liked MicroMentor much better, because I could get the answer right away. It saved me time and also helped me to remember the process better."* – Asker 2

Most askers felt it was easier and faster to receive help with MicroMentor compared to online resources, noting the time needed to review irrelevant or superfluous details:

> *"[MicroMentor] was much more helpful. I found myself not having that enough patien[ce] to look through the web for the answers to my specific question, so I would end up figuring things out by trial-and-error. Either way it's much less efficient."* – Asker 5

Contrasting with online resources, participants preferred MicroMentor:

> *"I learned much more in the 1 hour session with MicroMentor than the other hour where I just followed along a YouTube tutorial. This is especially true in the tutorial I was following where the Fusion command locations are different!"* – Asker 8

> *"Seeking help independently usually results in general tutorials that may not cover the specifics of what you want to do and there is a lot of sifting through irrelevant information. Also finding the solution to your exact case is a matter of luck because it depends on someone else already having that issue and posting about it online. With the MicroMentor system, you can find an answer to the specific problem you're having and get related tips or best practices to gain even more knowledge."* – Asker 6

> *"I think that it was a much more rapid way to receive information from experts compared to lengthy forum discussion"* – Helper 2

Participants felt MicroMentor would be beneficial for any software user, regardless of expertise. They also felt quick help systems could be used to solve a wide variety of problems. Novices and experts could learn about best practices, specialized software, different strategies for solving a problem, and ask for online resource recommendations. When online documentation is limited, quickly connecting with another user could fill in the gaps.

DISCUSSION AND FUTURE WORK

Overall, askers and helpers were satisfied with the help they received (Figure 11). Compared to the formative study, askers were satisfied with the answer for roughly the same percentage of requests (85% vs. 83%), suggesting remote quick help could be as effective as in-person quick help. As with the formative study, participants overall felt 3 minutes was enough time to give or receive the help needed to solve a problem. Participants also thought using a quick help system like MicroMentor was a valuable way to seek help and preferred it over traditional help seeking strategies. These promising results suggest quick help systems like MicroMentor could become popular help seeking tools.

MicroMentor sets a foundation for future work on quick help and we believe there are many exciting opportunities in this space. First, more work is needed to better understand the impacts of quick help during long term, in-the-wild use. With more practice, users may adopt new strategies for asking and answering questions. The volume of, and type of questions being asked may also change as users gain more skills.

With long term use, the same askers and helpers may interact with one another multiple times. Future quick help systems should consider detecting these moments and providing additional communication channels to foster a deeper mentor-mentee relationship. For example, a user could volunteer to act as someone's "buddy," and become the first point of contact when there is a new request.

There are different approaches for matching mentors to mentees and many factors of varying importance that could be leveraged. For our proof-of-concept, we selected factors inspired by previous work and results from the formative study. However, future versions could allow users to select factors they value the most and assign their own weights. Alternatively, more context, like archive search history and transcripts, could be leveraged to improve these algorithms. Future work should validate alternative approaches.

There are many triaging styles for routing questions to helpers, especially when a question is asked multiple times. Archives could include more information, like an "archive score" that specifies how relevant an archive is to what you are doing. This would encourage more use and help uncover useful archives in a context-aware manner [34].

MicroMentor uses push notifications to notify helpers of an incoming request, but this approach may be too intrusive. The system allows users to specify their availability to reduce the number push notifications they receive, and while this is a good first step towards reducing the number of notifications, automatically detecting breaks or moments of inactivity may be a better strategy.

There are some privacy concerns. Users may not want to show their face or screen with random community members, especially if this information is archived for others to access and share. Future quick help systems should consider implementing different privacy settings, with the option for an "anonymous" mode where all personal information is removed from the request and the session is not archived.

While we recruited Fusion 360 experts and novices to become helpers and askers, respectively, anyone could ask questions and provide help using MicroMentor. The help session dynamics between different types of users would be interesting to explore and future quick help systems may need to adjust in-meeting tools to provide more support for less experienced users who are acting as helpers.

Related, MicroMentor was informally tested using other applications like Excel, but the main design was formed with GUI creativity and 3D design tools in mind. As such, the time limit and command extractions may not generalize to other domains. For example, MicroMentor relies on clicked GUI elements to extract command histories, which may not be relevant to the main functionalities of a particular application, like a programming or math tool. We recognize other limitations of this approach (false command classifications, no keyboard shortcuts, applications must be accessible), but it was useful for our proof-of-concept. MicroMentor may still be helpful for users trying to find GUI settings and menus in other types of applications, such as finding the option to download a development package in a programming tool or export a graph in a statistics tool. However, future work should validate the time limit for other types of applications and explore richer interactions with software that could shared with other quick help systems.

One outstanding challenge is extending the system to a larger user base and promoting greater participation. MicroMentor gives users a 'point' when they answer a request, but advanced gamifications or additional incentives, such as cloud credits for the target application, could work well. Complex questions may need additional time to receive a more complete solution. When submitting a request, askers could spend points for a "premium help request" (Figure 12), securing an expert helper and more time (5 minutes). Alternatively, askers and helpers could agree to extend a help session in the moment, up until a new time limit (5 minutes) by spending points. There are many possibilities, and future work should evaluate the best models for quick help sessions.

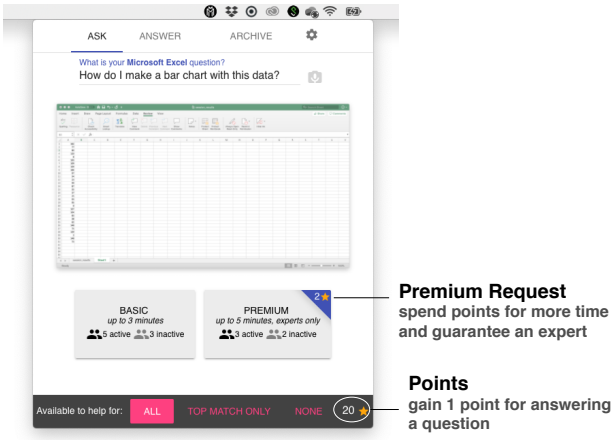


Figure 12: Alternative design for 'premium' help sessions that could incentivize greater participation from experts.

Another interesting avenue for future work, is how the data gathered from MicroMentor could be used to inform future interface design. For example, if there are many questions about particular command, MicroMentor could automatically send this data to the respective companies and suggest improvements for future software updates.

CONCLUSION

We explore the idea of quick, one-on-one help sessions. Results from a formative study suggest 3 minutes is often enough time for a novice to receive help from an expert, with requests marked as having been answered 83% of the time. We use these findings to drive the development of our proof-of-concept system, MicroMentor. MicroMentor promotes a rich shared context and lowers the transaction costs of requesting help from other members in the community by automatically capturing and attaching supplementary materials, finding suitable helpers, initializing a video conference, and archiving requests for later use. Results from a 12 participant lab study found users were satisfied with the answer they received through MicroMentor 85% of the time. Our work serves as a foundation for future quick 1-1 help systems, and we hope more research will focus on turning impactful, micro moments into big learning opportunities.

ACKNOWLEDGEMENTS

We thank Rebecca Krosnick and Kimia Kiani for their help facilitating our user studies.

REFERENCES

[1] Mark S. Ackerman and Thomas W. Malone. 1990. Answer Garden: a tool for growing organizational memory. *Proceedings of the conference on Office information systems -*, ACM Press, 31–39. http://doi.org/10.1145/91474.91485

[2] Mark S. Ackerman and David W. McDonald. 1996. Answer Garden 2: merging organizational memory with collaborative help. *Proceedings of the 1996 ACM conference on Computer supported cooperative work - CSCW '96*, ACM Press, 97–105. http://doi.org/10.1145/240080.240203

[3] Andrea L. Ames. 2001. Just what they need, just when they need it: an introduction to embedded assistance. *Proceedings of the 19th annual international conference on Computer documentation - SIGDOC '01*, ACM Press, 111. http://doi.org/10.1145/501516.501539

[4] Muhammad Asaduzzaman, Ahmed Shah Mashiyat, Chanchal K. Roy, and Kevin A. Schneider. 2013. Answering questions about unanswered questions of Stack Overflow. *2013 10th Working Conference on Mining Software Repositories (MSR)*, IEEE, 97–100. http://doi.org/10.1109/MSR.2013.6624015

[5] Lucy M. Berlin and Robin Jeffries. 1992. Consultants and apprentices: observations about learning and collaborative problem solving. *Proceedings of the 1992 ACM conference on Computer-supported cooperative work - CSCW '92*, ACM Press, 130–137. http://doi.org/10.1145/143457.143471

[6] Colin Camerer and Martin Weber. 1992. Recent developments in modeling preferences: Uncertainty and ambiguity. *Journal of Risk and Uncertainty* 5, 4: 325–370. http://doi.org/10.1007/BF00122575

[7] John M. Carroll. 1990. *The Nurnberg funnel: designing minimalist instruction for practical computer skill*. MIT Press, Cambridge, MA, USA.

[8] John M. Carroll and Mary Beth Rosson. 1987. Paradox of the Active User. In *Interfacing Thought: Cognitive Aspects of Human-Computer Interaction*. MIT Press, Cambridge, MA, USA, 80–111.

[9] Yan Chen, Sang Won Lee, Yin Xie, YiWei Yang, Walter S. Lasecki, and Steve Oney. 2017. Codeon: On-Demand Software Development Assistance. *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems - CHI '17*, ACM Press, 6220–6231. http://doi.org/10.1145/3025453.3025972

[10] Yan Chen, Steve Oney, and Walter S. Lasecki. 2016. Towards Providing On-Demand Expert Support for Software Developers. *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems - CHI '16*, ACM Press, 3192–3203. http://doi.org/10.1145/2858036.2858512

[11] Parmit K. Chilana, Tovi Grossman, and George Fitzmaurice. 2011. Modern software product support processes and the usage of multimedia formats. *Proceedings of the 2011 annual conference on Human factors in computing systems - CHI '11*, ACM Press, 3093. http://doi.org/10.1145/1978942.1979400

[12] Parmit K. Chilana, Nathaniel Hudson, Srinjita Bhaduri, Prashant Shashikumar, and Shaun Kane. 2018. Supporting Remote Real-Time Expert Help: Opportunities and Challenges for Novice 3D Modelers. *2018 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, IEEE, 157–166. http://doi.org/10.1109/VLHCC.2018.8506568

[13] Parmit K. Chilana, Andrew J. Ko, and Jacob O. Wobbrock. 2012. LemonAid: selection-based crowdsourced contextual help for web applications. *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems - CHI '12*, ACM Press, 1549. http://doi.org/10.1145/2207676.2208620

[14] Codementor. Codementor | Get live 1:1 coding help, hire a developer, & more. Retrieved September 8, 2019 from https://www.codementor.io

[15] Daniel Ellsberg. 1961. Risk, Ambiguity, and the Savage Axioms. *The Quarterly Journal of Economics* 75, 4: 27. http://doi.org/10.2307/1884324

[16] G. W. Furnas, T. K. Landauer, L. M. Gomez, and S. T. Dumais. 1987. The vocabulary problem in human-system communication. *Communications of the ACM* 30, 11: 964–971. http://doi.org/10.1145/32206.32212

[17] Susan R. Fussell, Robert E. Kraut, and Jane Siegel. 2000. Coordination of communication: effects of shared visual context on collaborative work. *Proceedings of the 2000 ACM conference on Computer supported cooperative work - CSCW '00*, ACM Press, 21–30. http://doi.org/10.1145/358916.358947

[18] Max Goldman, Greg Little, and Robert C. Miller. 2011. Real-time collaborative coding in a web IDE. *Proceedings of the 24th annual ACM symposium on User interface software and technology - UIST '11*, ACM Press, 155. http://doi.org/10.1145/2047196.2047215

[19] Tovi Grossman and George Fitzmaurice. 2010. ToolClips: an investigation of contextual video assistance for functionality understanding. *Proceedings of the 28th international conference on Human factors in computing systems - CHI '10*, ACM Press, 1515. http://doi.org/10.1145/1753326.1753552

[20] Tovi Grossman, George Fitzmaurice, and Ramtin Attar. 2009. A survey of software learnability: metrics, methodologies and guidelines. *Proceedings of the 27th international conference on Human factors in computing systems - CHI 09*, ACM Press, 649. http://doi.org/10.1145/1518701.1518803

[21] Philip J. Guo, Jeffery White, and Renan Zanelatto. 2015. Codechella: Multi-user program visualizations for real-time tutoring and collaborative learning. *2015 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, IEEE, 79–87. http://doi.org/10.1109/VLHCC.2015.7357201

[22] Susan M. Harrison. 1995. A comparison of still, animated, or nonillustrated on-line help with written or spoken instructions in a graphical user interface. *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '95*, ACM Press, 82–89. http://doi.org/10.1145/223904.223915

[23] Arienne Holland. 2014. How Estimated Reading Times Increase Engagement With Content. *Marketing Land*. Retrieved September 13, 2019 from https://marketingland.com/estimated-reading-times-increase-engagement-79830

[24] Damon Horowitz and Sepandar D. Kamvar. 2010. The anatomy of a large-scale social search engine. *Proceedings of the 19th international conference on World wide web - WWW '10*, ACM Press, 431. http://doi.org/10.1145/1772690.1772735

[25] Nathaniel Hudson, Celena Alcock, and Parmit K. Chilana. 2016. Understanding Newcomers to 3D Printing: Motivations, Workflows, and Barriers of Casual Makers. *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems - CHI '16*, ACM Press, 384–396. http://doi.org/10.1145/2858036.2858266

[26] Nathaniel Hudson, Parmit K. Chilana, Xiaoyu Guo, Jason Day, and Edmund Liu. 2015. Understanding triggers for clarification requests in community-based software help forums. *2015 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, IEEE, 189–193. http://doi.org/10.1109/VLHCC.2015.7357216

[27] Aaron Jacobs. *accessibility: Extension module that wraps the Accessibility API for Mac OS X.* Retrieved December 20, 2019 from https://github.com/atheriel/accessibility

[28] Akshay Java, Xiaodan Song, Tim Finin, and Belle Tseng. 2007. Why we twitter: understanding microblogging usage and communities. *Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 workshop on Web mining and social network analysis - WebKDD/SNA-KDD '07*, ACM Press, 56–65. http://doi.org/10.1145/1348549.1348556

[29] Kimia Kiani, George Cui, Andrea Bunt, Joanna McGrenere, and Parmit K. Chilana. 2019. Beyond "One-Size-Fits-All": Understanding the Diversity in How Software Newcomers Discover and Make Use of Help Resources. *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems - CHI '19*, ACM Press, 1–14. http://doi.org/10.1145/3290605.3300570

[30] Kevin Knabe. 1995. Apple guide: a case study in user-aided design of online help. *Conference companion on Human factors in computing systems - CHI '95*, ACM Press, 286–287. http://doi.org/10.1145/223355.223677

[31] Maria Konnikova. 2013. A List of Reasons Why Our Brains Love Lists. Retrieved September 13, 2019 from https://www.newyorker.com/tech/elements/a-list-of-reasons-why-our-brains-love-lists

[32] Wendy E. Mackay. 1990. Patterns of sharing customizable software. *Proceedings of the 1990 ACM conference on Computer-supported cooperative work - CSCW '90*, ACM Press, 209–221. http://doi.org/10.1145/99332.99356

[33] Justin Matejka, Tovi Grossman, and George Fitzmaurice. 2011. IP-QAT: in-product questions, answers, & tips. *Proceedings of the 24th annual ACM symposium on User interface software and technology - UIST '11*, ACM Press, 175. http://doi.org/10.1145/2047196.2047218

[34] Justin Matejka, Tovi Grossman, and George Fitzmaurice. 2011. Ambient help. *Proceedings of the 2011 annual conference on Human factors in computing systems - CHI '11*, ACM Press, 2751. http://doi.org/10.1145/1978942.1979349

[35] Justin Matejka, Wei Li, Tovi Grossman, and George Fitzmaurice. 2009. CommunityCommands: command recommendations for software applications. *Proceedings of the 22nd annual ACM symposium on User interface software and technology - UIST '09*, ACM Press, 193. http://doi.org/10.1145/1622176.1622214

[36] David W. McDonald and Mark S. Ackerman. 2000. Expertise recommender: a flexible recommendation system and architecture. *Proceedings of the 2000 ACM conference on Computer supported cooperative work - CSCW '00*, ACM Press, 231–240. http://doi.org/10.1145/358916.358994

[37] Medium. 2014. Read Time and You. *Medium*. Retrieved September 13, 2019 from https://blog.medium.com/read-time-and-you-bc2048ab620c

[38] David G. Novick, Edith Elizalde, and Nathaniel Bean. 2007. Toward a more accurate view of when and how people seek help with computer applications. *Proceedings of the 25th annual ACM international conference on Design of communication - SIGDOC '07*, ACM Press, 95. http://doi.org/10.1145/1297144.1297165

[39] Susan Palmiter and Jay Elkerton. 1991. An evaluation of animated demonstrations of learning computer-based tasks. *Proceedings of the SIGCHI conference on*

Human factors in computing systems Reaching through technology - CHI '91, ACM Press, 257–263. http://doi.org/10.1145/108844.108906

[40] Fatemeh Riahi, Zainab Zolaktaf, Mahdi Shafiei, and Evangelos Milios. 2012. Finding expert users in community question answering. *Proceedings of the 21st international conference companion on World Wide Web - WWW '12 Companion*, ACM Press, 791. http://doi.org/10.1145/2187980.2188202

[41] Michael B. Twidale. 2005. Over the Shoulder Learning: Supporting Brief Informal Learning. *Computer Supported Cooperative Work (CSCW)* 14, 6: 505–547. http://doi.org/10.1007/s10606-005-9007-7

[42] Christina Willner. 2017. 5 Benefits of using Time Estimates in your To-do List. *Amazing Marvin*. Retrieved September 13, 2019 from https://blog.amazingmarvin.com/5-benefits-of-using-time-estimates-in-your-to-do-list/

[43] How do I ask a good question? - Help Center. *Stack Overflow*. Retrieved September 8, 2019 from https://stackoverflow.com/help/how-to-ask

[44] Rogue 🌻. Bigham on Twitter: "@deliprao haha, tweets are way better. first, you did the tiniest amount (at least) of background on me. second, they're short! easier to read, and incentivizes getting to the point. third, my response is expected to be short, so need to ask something that gets to the heart of it!" / Twitter. *Twitter*. Retrieved September 19, 2019 from https://twitter.com/jeffbigham/status/1165383660666859520

[45] Powers of 10: Time Scales in User Experience. *Nielsen Norman Group*. Retrieved September 19, 2019 from https://www.nngroup.com/articles/powers-of-10-time-scales-in-ux/