Deja Queue: Examining Office Hours Queues

A Research Paper submitted to the Department of Engineering and Society

Presented to the Faculty of the School of Engineering and Applied Science
University of Virginia • Charlottesville, Virginia

In Partial Fulfillment of the Requirements for the Degree
Bachelor of Science, School of Engineering

Winston Liu
Spring, 2020

On my honor as a University Student, I have neither given nor received unauthorized aid on this assignment as defined by the Honor Guidelines for Thesis-Related Assignments

Signature [illegible] Date May 9, 2020
Winston Liu

Approved S. Travis Elliott Date May 9, 2020
S. Travis Elliott, Department of Engineering and Society

Introduction

Professors and teaching assistants hold office hours to provide students with dedicated time periods where they may seek help, advice, or otherwise be able to meet with the course staff. With courses facing enrollment sizes of over 1,200 in some cases ("Monstrous class sizes unavoidable at colleges", 2007), the number of students seeking help at office hours may overwhelm the availability of the course staff. To provide order to what could otherwise turn into a chaotic scene, a queueing system may be used to efficiently keep track of how many students need help and which students to help next. At the same time, it is important to keep in mind that office hour queues, just as with any technology, are more than just a piece of software and has many social-technical aspects to consider.

The lack of extensive research on office hours queues was noted by MacWilliam & Malan in 2011 and remains an issue today, but anecdotal evidence suggests that queues have become a common staple of computer science courses and are a critical aspect of course administration (MacWilliam & Malan, 2013; Juett, n.d.; "Office Hours", n.d.; "CS 61B Queue", n.d.; Banerji, n.d.; Tychonievich, Brunelle, & Pettit, n.d.; etc.). While queues themselves have been studied extensively as part of "queueing theory" (Cooper, 1981), much of the field of study has focused on impersonal applications of queues, rather than investigating the impact a queue can have when applied in a social context. Indeed, *Queuing Systems*, a journal dedicated to queueing theory, is "primarily interested in probabilistic and statistical problems in [queueing theory]" ("Queuing Systems", n.d.). Consequently, many of the aforementioned queues have been created to improve office hours from a purely administrative standpoint or were a natural result of technological improvement. At Stanford, "challenges have encouraged professors and TAs to create new methods aimed at ensuring each student receives the necessary help…including online office hours [queues]" (Park, 2018). The University of Pennsylvania makes their motivation for using office hours even more clear: "an online office hours queue is in the works [to address wait times]" (Basu, 2018). Looking at it from a different perspective, Malan (2009) writes that "fifteen years ago, students signed up for such help by writing their name on a sheet of paper...eventually, that sheet of paper evolved into a whiteboard, but the process today remains largely the same. It works, and it is simple". This viewpoint that treats queues as a simple process is largely misleading and does not adequately address social interactions. Additionally, the emphasis on providing the means to receive help, rather than focusing on the process itself, leaves an incomplete picture of how a queue impacts its environment.

By looking at queues through a socio-technical lens and applying the Social Construction of Technology framework, it becomes apparent that there are many nuances to queues that have important implications beyond course administration that extend to students, course staff, and more. In particular, a queue is just not a supplementary tool for office hours. It is a full-fledged application, with multiple considerations to be made to optimize its usefulness for any given course.

Background

In their most basic form, queues model a system in which customers arrive and are processed by order of priority by any number of servers. This model is a common occurrence, playing a part in supermarket lines, customer service helpdesks, internet traffic, automobile traffic, etc. In the context of office hours, students are asked to join a queue and are then helped by the course staff, typically on a first-come-first-serve basis. Unlike the emergence of major learning management systems, academia has as-of-yet been unable to consolidate office hour queueing systems. As a result, a plethora of queues have been created to suit the needs of individual professors, with minimum interaction or information sharing between the different authors. A prime example of this fragmentation can be seen at the University of Virginia, where at the time of this writing there are no less than four separate queue systems that are actively used within just the Computer Science department.

Conceptual Framework

Social Construction of Technology (SCOT) can be applied to office hour queues to comprehensively analyze how queues impact students, course staff, and various other stakeholders. Importantly, SCOT emphasizes that each technology can be used differently by distinct groups and independently adapted to best suit the needs of each group. Additionally, SCOT recognizes that technologies are shaped by their *stakeholders*, and that it is essential for any successful technology to identify its stakeholders and keep them in mind when making decisions.

Analysis

Across the world, university professors have transitioned to using online queues to administrate office hours. While every queue is based on the same underlying model, each one has developed independently to support the separate needs and use cases of each course. These queues have defined their own concepts of fairness, usability, and efficiency, metrics which may be analyzed from the unique perspective of each different stakeholder.

Stakeholders

There are two primary stakeholders that drive the success of an office hours queue – the course staff and the students. The *course staff*, who are generally teaching assistants or other individuals who provide a supporting role in the course, can be identified as one of the primary stakeholders as they interact directly with the queue to hold office hours. The extent to which they are able to perform their staff duties hinges on the efficacy of the queue. At the same time, it would be remiss to not also name the *students* as primary stakeholders. As students attending office hours, they require a functioning system that they are able to easily navigate in order to receive the help they need. These two groups are the ones that are not only the most impacted by any implementation of a queue, but also the most dependent on having a working queue available. While their needs are oftentimes similar, there are also occasions when their priorities are at odds with each other. Course staff, for example, prioritize the ability to know what students need help with and tools to help them minimize the amount of time each student is kept waiting, which may involve helping groups of students at the same time. Students, on the other hand, want detailed and individualized help as quickly as possible, even if that means not providing detailed information on what they have questions on.

In addition to the primary stakeholders, there are also a variety of secondary stakeholders that play a part in the development of office hour queues. Queues provide multiple benefits to *professors*, such as allowing them to devote more of their time to other aspects of course administration, or enabling the collection of various statistics that can be used to identify potential concerns or places for improvement. They are also the ones that are directly responsible for the performance and success of the queue. The *software developers* are the ones who create and maintain the queue. Oftentimes, they may be a part of another stakeholder group as well, such as the course staff or professors, but they may also be otherwise completely detached from the course, and for the purposes of this paper, they will be treated as a separate stakeholder group. Looking more broadly, *university administrators* are indirectly involved with office hour queues through the handling of room reservations. While these stakeholders are more detached from the queue, they remain as voices that have an important say in the final design of the queue. Just as with the primary stakeholders, the values of secondary stakeholders sometimes seem to be mutually incompatible. While professors and administrators want to be able to use stable software releases, software developers continuously tweak and improve their product, occasionally leading to compatibility issues with the professor's or university's computing framework.

SCOT allows us to examine office hour queues from a multi-faceted point of view, keeping in mind that for every decision, there are multiple other possibilities that may be valid for other queue implementations. In the end, the final vision of the queue depends on the priorities of the various stakeholders, and no implementation is necessarily superior to any other because of its design decisions.

Fairness

Receiving help during office hours should be an impartial process – there should be no bias on the part of the course staff when selecting which students to help. In a traditional office hours system, students wait to receive help in the order that they arrive at office hours. Similarly, most queues, as the name itself implies, model the traditional way of holding office hours by utilizing a first-in-first-out (FIFO) strategy. This is a familiar and ingrained model to students and course staff, and needs practically no introduction or training to be used. However, it would be a naïve mistake to assume that FIFO is the best option simply because that is the model that most office hours follow.

Consider an office hours session that runs for two hours, say from 2-4pm. If every student had equal opportunity to attend this office hours session, then perhaps a FIFO algorithm might appear to be fair. However, the reality is often much murkier. Students have classes throughout the day that may prevent them from being present when office hours begin. Some students may pre-emptively join the queue, even before they have any questions to ask. Different students will be working on different assignments, some due later that night, some due far into the future. Given all these considerations, FIFO may not be as fair as it initially appears to be.

To increase fairness, developers and professors may look to change the priority algorithm to take into account factors other than time-of-entry. If the goal is to prevent preemptive help requests and allow students with scheduling conflicts to receive similar amounts of help, a useful priority metric would be number-of-entries, rather than time-of-entry. Under this system, the more times a student has requested help, the lower their priority is. Thus, there is an active incentive for students to not enter the queue until they need help from course staff; likewise, students with less access to the queue are still able to receive help without being disadvantaged by their lesser availability. However, this is a double-edged sword: students may feel as if they are actively being punished for taking the initiative to seek out help. Course staff may also become more inclined to stay with a student longer to answer all the questions they have, rather than only guiding the student to a starting point so that the student does not suffer the decreased priority penalty if the starting point was not enough for them to complete the assignment on their own. A similar problem exists when tuning priority to account for assignment deadlines. Students working on assignments with closer deadlines would be prioritized over students working on later assignments. While this greatly helps students who need some extra help in order to complete the assignment, it also disincentivizes starting assignments early and going to office hours to receive advance guidance, as the queue will assign them a low priority, leading to longer wait times.

As the professor is the one running the course, it is up to the professor to determine what they believe is most fair for both the students and course staff. Any decision they make has a direct impact on those stakeholders, who must go with the professor's solution. Of course, a professor is also limited by the options that the developer has integrated into the queue.

Usability

For a queue to be useful, students, course staff, and professors must be able to easily manipulate the queue to perform any number of tasks, from joining the queue as a student, to viewing the number of students in the queue as a member of the course staff, to differentiating students and course staff as a professor. A queue that is difficult to use will passively discourage students and course staff from using the queue and lower the likelihood of positive office hour experiences.

One way to increase usability is for the developer to take advantage of modern browser features. Allowing the queue to display properly on phones or other form factors gives more flexibility to course staff and students to choose what is the most comfortable for them. Showing a notification to course staff whenever a student enters an empty queue improves response speeds and means that course staff won't accidentally keep students waiting. Additionally, if the queue is able to update automatically without having to manually refresh the application, both course staff and students will be given a better idea of the current status of the queue. However, integrating newer features into the queue also necessitates raising the minimum computer requirements needed to use the queue (Deveria, n.d.). The developer must be aware of these tradeoffs and clearly communicate them to the professor, who will then also be tasked with deciding what a reasonable technological baseline is for the students and course staff.

Usability often depends on the number of features available in a queue. Allowing course staff to "requeue" students, for example, reduces the work that needs to be done to hand off one student to another member of the course staff. While it may appear harmless to add on features to improve usability, any extra features should be carefully validated by the relevant stakeholders to ensure that the desired outcome will actually be achieved. MacWilliam & Malan observed that their attempt to direct students with similar questions to the same course staff member did not end up working, as they did not consider that there was no perceivable incentive for students to be detailed in their problem description, nor did they effectively communicate to course staff that students would be providing a description of what they needed help on, and thus most course staff "did not have [the queue] open…and were unaware of the students' questions until they arrived in person".

In order to use the queue, members of the course must not only be introduced to it but also be able to find and utilize it without assistance later on. Thus, terminology plays an important role in the usability of a queue. The very notion of a "queue" inserts additional complexity into the very general concept of office hours. A queue must have a strict ordering; office hours do not have such a requirement. The term "queue" dehumanizes the experience and introduces both technical vocabulary and unnecessary implementation details into an abstract concept. In American English, "queue" is also seldomly used – it is not one of the 5,000 most common words ("Word frequency data", n.d.). Considering that the most common 3,000 words comprise 90% of English conversations ("3000 most common words in English", n.d.), the correlation between queues and office hours may not be immediately obvious. Thus, the developer may want to limit using “queue” in user-facing contexts, resorting instead to more generic terms such as “office hours” or “ask for help”, while professors and course staff should do the same and abstract away the details of the queue when discussing office hours with students.

At every step of the process, care must be taken to consider what impact certain design decisions may have on the final usability of the queue. Students and course staff, as the primary stakeholders, are a crucial voice and should be consulted during the development of a queue so that no unexpected surprises arise later on.

Efficiency

Queues are typically introduced as a part of the course structure to allow the course to run more efficiently. Some of the benefits are very clear – queues automatically keep track of who needs to be helped next, allow many course staff to be present at office hours without interfering with each other, and clearly identify course staff as such. Some of the benefits may be less obvious, such as allowing office hours to occur in large common areas, perhaps concurrently with other courses’ office hours, which simplifies or even reduces administrative logistics for reserving rooms, and providing professors with a wealth of data that can be analyzed to identify trends and points of concern. As one example of the increased efficiency queues can bring, Harvard University courses’ decisions to move office hours into dining halls were facilitated by the development of digital queues and resulted in “more than linear” office hours attendance as the new locations were “more convenient and social for students, [and] motivated higher attendance” (MacWilliam & Malan, 2013). Furthermore, MacWilliam & Malan, the professors of the aforementioned courses, found that moving to an online queue allowed them to understand students' work habits and "quantify students' confusion on each [assignment]". These benefits are all strong arguments for backing office hours with digital queues.

At the same time, queues also make some tasks harder to perform. A queue by itself is inherently focused on one student at a time and has no way to help multiple students at once. This can reduce efficiency when multiple students are having the same problem, or when there is a difficult concept that the course staff would like to reinforce with everyone. In the case of MacWilliam & Malan, they found that "it was not uncommon for multiple teaching assistants to answer the same question independently and concurrently", even after developing a queue that tried to reduce the number of duplicate questions. Furthermore, while queues can indeed allow office hours to take place in common areas, that same advantage can also make it harder for the course staff to find the student that they are looking for. Finally, care must be taken by the developer to ensure that the queue performance does not deteriorate as time goes on and that no matter how many students are in, or have been in, the queue, course staff are able to find out which student they are helping next in real time.

Once again, it is up to the judgement of the professor to determine whether the benefits outweigh the potential disadvantages. Is this a course that is large enough to take full advantage of the benefits that a queue provides? Is the professor willing to dedicate the time to analyzing queue statistics to improve the course? Did the developer even provide a way to view those statistics? Is there an established procedure to help groups of students at a time? All of these questions and more should be considered before committing to using queues.

Conclusion

On the surface, office hour queues appear to be a simple solution to a straightforward problem. Once examined using Social Construction of Technology, though, many considerations from the stakeholders arise that must be addressed to adequately provide fairness, a good experience, and an efficient workflow. Queues cannot be viewed as just a way to return order to office hours. Rather, they should be seen as complex technologies that have the ability to alter multiple aspects of course administration as well as students' overall performance. By changing the lens through which we examine not just office hours queues, but the use of various technologies in general, many lessons can be learned for how to better integrate technology into the lives of those around us.

References

3000 most common words in English. (n.d.). Retrieved March 26, 2020, from https://www.ef.com/wwen/english-resources/english-vocabulary/top-3000-words/

Banerji, D. K. (n.d.). debkbanerji/office-hours-queue. Retrieved March 26, 2020, from https://github.com/debkbanerji/office-hours-queue

Basu, K. (2018, December 5). Some CIS courses are so overloaded that students wait more than an hour for homework help. Retrieved from https://www.thedp.com/article/2018/12/cis-120-office-hours-wait-time-penn-upenn-philadelphia

Cooper, R. B. (1981). Queueing theory. *Proceedings of the ACM '81 Conference*. doi: 10.1145/800175.809851

CS 61B Queue. (n.d.). Retrieved March 26, 2020, from https://oh.datastructur.es/

Deveria, A. (n.d.). Can I use... Support tables for HTML5, CSS3, etc. Retrieved March 27, 2020, from https://caniuse.com/#info_about

Juett, J. (n.d.). EECS Office Hours. Retrieved from https://lobster.eecs.umich.edu/eecsoh/

MacWilliam, T., & Malan, D. (2013). Scaling office hours: managing live Q&A in large courses. Journal of Computing Sciences in Colleges, 28(3). doi: 10.5555/2400161.2400179

Malan, D. J. (2009). Virtualizing office hours in CS 50. ACM SIGCSE Bulletin, 41(3), 303. doi: 10.1145/1595496.1562969

Monstrous class sizes unavoidable at colleges. (2007, November 24). Retrieved from http://www.nbcnews.com/id/21951104/ns/us_news-education/t/monstrous-class-sizes-unavoidable-colleges/

Office Hours. (n.d.). Retrieved March 26, 2020, from https://cmu.ohqueue.com/#/

Park, E. (2018, January 29). Honor code cases, office hour queues among challenges of expanding CS class sizes. Retrieved from https://www.stanforddaily.com/2018/01/28/honor-code-cases-office-hour-queues-among-challenges-of-expanding-cs-class-sizes/

Queueing Systems. (n.d.). Retrieved from https://www.springer.com/journal/11134

Tychonievich, L., Brunelle, N., & Pettit, R. (n.d.). CS 1110/1111 – Office Hours. Retrieved March 26, 2020, from https://cs1110.cs.virginia.edu/oh.html

Word frequency data. (n.d.). Retrieved March 26, 2020, from https://www.wordfrequency.info/free.asp?s=y