WEBSITE MAINTENANCE: MANAGING AND IMPLEMENTING USER REQUESTS

THE RELATIONSHIP BETWEEN USER-DEVELOPER COMMUNICATION AND SOFTWARE BIAS

A Thesis Prospectus
In STS 4500
Presented to
The Faculty of the
School of Engineering and Applied Science
University of Virginia
In Partial Fulfillment of the Requirements for the Degree
Bachelor of Science in Computer Science

By
Anna Williamson

November 1, 2022

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.

ADVISORS

Catherine D. Baritaud, Department of Engineering and Society

Briana Morrison, Department of Computer Science

The general topic of this project involves examining user-developer communication during the software development cycle, with a focus on the requirements elicitation process and maintenance. The technical report will discuss methods for streamlining user-developer communication and how to apply user feedback to software development. The STS report will discuss the biases present in users and developers and how this impacts the collection and analysis of user feedback. There are many obstacles preventing complete and useful communication between developers and users, including user and developer bias, lack of communication channels, indirect communication that leads to misunderstood feedback, differing levels of system understanding between users and developers, and various social factors that prevent honest feedback. All of these obstacles prevent user satisfaction with the final software product. In the industry of software development, understanding the desired product is critical since maintenance and future changes can comprise 90% of the total software cost (Dehaghani & Hajrahimi, 2013, p. 63). Misinterpreting or assuming prior knowledge of user needs can lead to the development of a product that is not used by the target group and ultimately fails. This can also allow bias to enter the system; since software is so ingrained in society, subtle biases in everyday software can significantly harm certain groups and perpetuate inequity. While more user influence in the development process is generally positive, it is essential for developers to examine how this influence should be exerted.

The idea of including users in the software development process is not new; developers understand that users play an important role in the development cycle and should be consulted. However, not much research has been dedicated to understanding how developers and users communicate and exactly what impact users can have on the final software product. Gallivan and Keil (2003) analyze the system CONFIG, created by an anonymous company in 1980, and reveal

that more user participation in the development process does not necessarily lead to a better product (p. 45). Developers worked to include users throughout the process by holding periodic meetings with a subset of the user group, integrating a feedback system into the software, and frequently conducting telephone surveys to gauge opinion on the new system (Gallivan & Keil, 2003, p. 46). However, they still found that users interacted very little with the finished system. Users were pressured by the company to accept the product and therefore moderated their feedback. They also assumed that developers were already aware of "obvious" problems with the software. The use of indirect feedback, i.e., collected through the telephone surveyor before reaching developers, served to warp original feedback collected from the user. Finally, developers were affected by overconfidence bias and did not consider that the system was not actually useful for users. CONFIG was abandoned in 1992 after an expensive redeployment initiative that did not improve its usage numbers (Gallivan & Keil, 2003, p. 45). This is an extreme example to motivate continued research into the communication pipeline between users and developers, focusing on how user needs are interpreted and applied to the developing software.

The technical and STS topics are tightly coupled because they both examine user-developer communication. The technical paper focuses on an internship experience with Dominion Energy in which the lack of official channels for collecting and implementing user feedback could be improved. It details a system for promoting user-developer communication and describes how this could improve internal operations for Dominion Energy. The STS paper focuses on the ethical concerns behind requirements elicitation, including the biases introduced into software through user-developer communication. The technical paper for this project will be completed by December 2nd, 2022 and the STS paper for this project will be completed by May

2nd, 2023. The technical paper will be overseen by Professor Briana Morrison in the Department of Computer Science and the STS paper will be overseen by Professor Catherine Baritaud in the Engineering and Society department.

DEVELOPING AN EFFECTIVE USER FEEDBACK SYSTEM

The technical report focuses on the lack of a user feedback system in Dominion Energy, an energy and power company centered in Virginia. HR employees at Dominion Energy desire changes to the internal HR website that will allow them to more efficiently search and filter employee information. The technical report relies on current research to present an appropriate user feedback collection system as well as techniques for effectively interpreting feedback in order to implement the desired changes. The system developed in this report will allow Dominion Energy to conduct internal affairs more productively and may prompt other companies to examine their internal systems for similar deficiencies and possible improvements.

The motivation for developing a method of user-developer communication regarding the company's HR website can be applied to other software products as well. Understanding and implementing user feedback will improve overall efficacy of the product and ensure that users are satisfied with the software. A more formal channel of communication works to ensure that developers can clearly understand and analyze the needs of the users. Figure 1 on page 4 shows the current system employed by Dominion Energy, which relies on direct email communication between users and developers. This method would not be suitable for a large number of user requests and so is not scalable. There is also a lack of clarity inherent in this method of communication. Non-technical users make requests that are significantly difficult or do not make logical sense based on the site setup. This makes it difficult to obtain a clear understanding of what would satisfy the user.

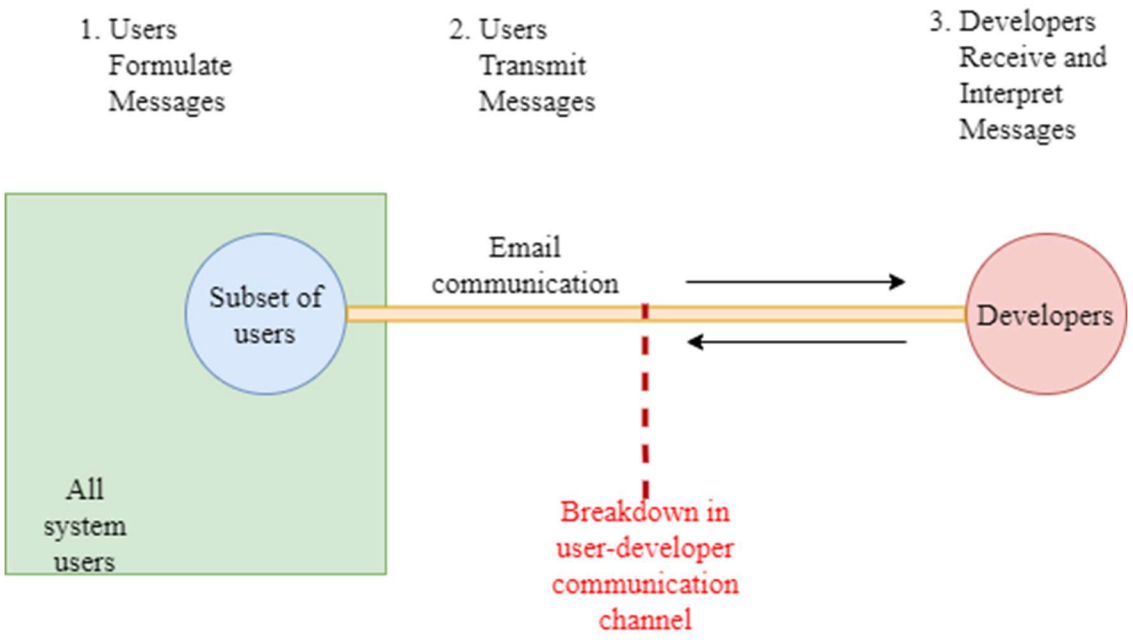


Figure 1: Current System of User-Developer Communication: There is only a single channel of user-developer communication in use. An error in any of the three given steps for users to transmit messages to developers will result in an unsatisfactory final product. (Adapted by Anna Williamson from M. Gallivan and M. Keil, 2003).

In the technical paper, a new user feedback system is developed by examining the current research surrounding user-developer communication channels. The frequency of feedback collection, the quality of communication via different mediums, and each user's understanding of the system (Johanssen et al., 2019) are all important to take into account when developing this new user feedback system. This new system is planned by examining case studies such as CONFIG, reviewing survey data about how other software companies handle user feedback collection and analysis, and understanding published research about where and how breakdowns in the user-developer communication pipeline occur. An effective system can be described for Dominion Energy by using prior research to support an argument for the adoption of new feedback collection techniques.

The technical paper discusses observed issues with Dominion Energy's current method of collecting user feedback and presents a system for improving the quality and results of this

feedback. Based on past research and an understanding of the internal workings of the company, anticipated outcomes are described. However, the system cannot at this time be implemented within Dominion Energy and therefore the paper cannot provide proof that the proposed system will improve the function of Dominion Energy's HR department. Based on analysis, proof of concept is provided and the overall benefits of improving user-developer communication is discussed. Finally, the technical paper is a technical report that describes a CS-related learning experience that may be useful to some outside group or individual. Specifically, this project discusses how the user feedback collection system could be redesigned and improved upon.

BIAS IN SOFTWARE SYSTEMS

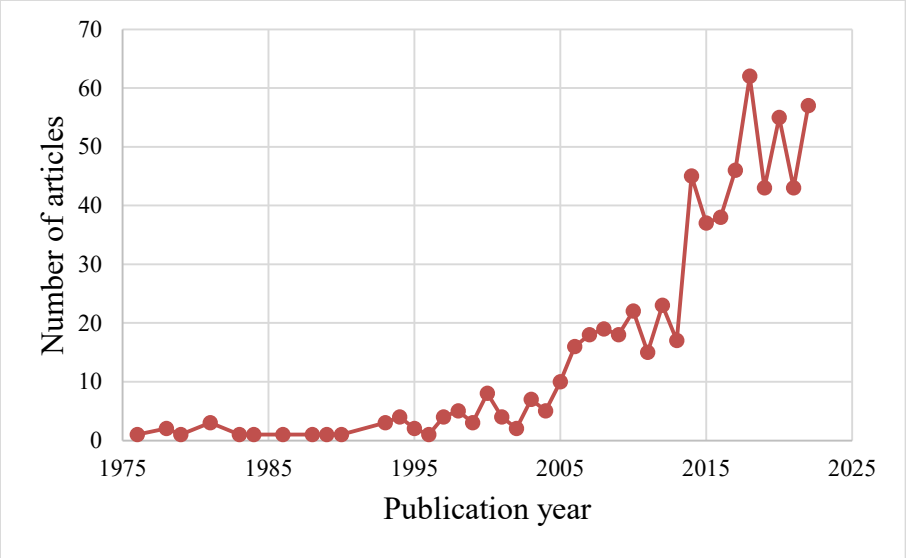
The STS report focuses on bias in software systems and specifically biases introduced through user-developer communication. The goal of the report is to answer the question: does user-developer communication foster bias in software and how can this be minimized? The presence of bias in artificial intelligence and facial identification is often discussed in modern media and it is important to realize that this bias also permeates everyday software. Friedman et al. (1996) examine some of the more subtle biases in graphical user interfaces (GUIs), educational software, and other less cutting-edge software systems (p. 49). Software is ingrained in society and inherent biases in software work to reinforce and perpetuate societal inequities, therefore creating a bias-free and fair society necessitates developing software systems that are bias-free. This topic is tightly-coupled to the topic chosen for the technical report since it considers user-developer communication. It specifically handles the developer's role in applying their own biases to interpreting user needs.

It is often difficult to identify subtle biases in more conventional or every-day software systems. For example, Friedman et al. (1996) point out the mainly male characters in computer

games, the GUI designs that have become increasingly difficult for the visually-impaired to navigate, and the assumption of prior knowledge in understanding software systems (p. 49). One of the ways that bias can infiltrate software systems is through the user-developer communication process. This paper references various case studies, psychology research, data collected on the type and frequency of feedback submitted by various user groups, and proposals for more ethical software development to understand how this occurs. Developers should be deliberate and aware of which users they collect feedback from, how many users from various social groups offer feedback, how situation and social pressures affect user feedback, how they interpret and apply user feedback, and how both developer and user biases can carry into the final software product. For example, Gallivan and Keil (2003) state that "…minority group members… are often reluctant to share their views publicly…" (p. 63), which affects what user needs are incorporated into the final software product. While more user influence in the development process is generally positive, it is essential for developers to examine how this influence should be exerted.

Identifying and measuring bias can be difficult and there is a significant lack of research into how user-developer communication throughout the requirements elicitation process influences the biases in software. As shown in Figure 2, the amount of published research in this area has increased but overall remains low. Research in this field is increasing, however the link between requirements engineering and bias in software systems is still not well understood or analyzed on a large scale.

Figure 2: Published Articles in the Association for Computing Machinery (ACM) Digital Library Related to Requirements Engineering and Bias: Filtering articles in the ACM Digital Library by



"requirements engineering" AND "bias" yields evidence of the amount of research in this field (Created by Williamson, 2022).

This STS paper examines the current research related to this topic and works to motivate future research about how software development processes allow subtle biases into the final product. It also discusses the current understanding of user-developer communication through the lens of Pinch and Bijker's Social Construction of Technology (SCOT) (1984), with an emphasis on the simultaneous influence of both users and developers on the software system. This relationship is shown in Figure 3, where the financial sponsor group denotes a general entity that commissions the development of the software product. The financial sponsor has an initial strong influence on the developers and the resulting software product as they commission the software product and define its function. Despite this initial strong influence, the focus of this paper is not on the power of the sponsor over development but rather on the presence of the user-developer communication pipeline. Users are impacted by various factors, such as personal experience and social pressures, that influence how they view the software, while developers are influenced by personal experience and bias. All of these elements combine to influence the final software product. In turn, the software product influences users, developers, and society.

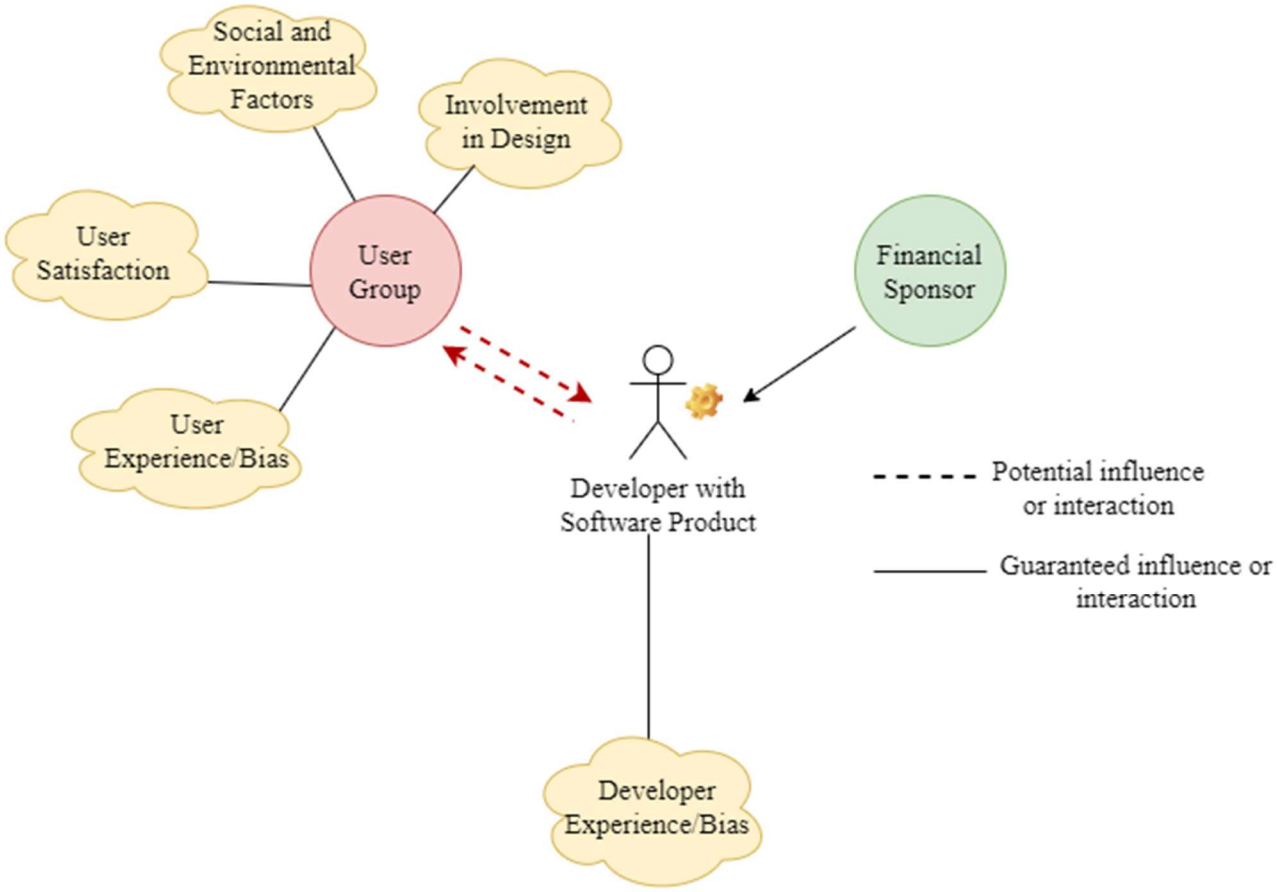


Figure 3: Adapted Social Construction of Technology (SCOT) model for the Influence and Interaction of the User Group in the Development of a Software Product: A depiction of the user group and financial sponsor influencing the final software product. There is a focus on the user-developer communication pipeline, which is not a guaranteed structure in the design system but connects a subset of the final user group with the developers. (Adapted by Anna Williamson from W. Carlson, 2009).

The anticipated outcome of this paper is both to encourage more research into this area as well as bring the issue of subtle software biases to the attention of developers. Software developers may be unaware of the biases they promote in the development of a software product. They may also be unaware of the potential for bias during user interaction. For example, developers who only survey or receive feedback from a certain group of users will be biased to create a system based on the needs of this specific group instead of for the larger target group. While more user influence in the development process is generally positive, it is essential for

developers to examine how this influence should be exerted. The STS paper will take the form of a summary and analysis of the current available research related to the topic of user-developer communication, biased software systems, and requirements engineering. It will also motivate the importance of conducting further research and encourage developers to consider this issue in their own software projects.

THE INTERSECTION OF REQUIREMENTS ENGINEERING AND BIAS

The technical and STS reports included in this project provide a comprehensive understanding of user-developer communication in software development, both from a technical and social standpoint. User-developer communication is important throughout the development cycle to ensure that software changes and improvements reflect the needs of the user. A software product that does not satisfy end users is a financial and time drain where attempts to improve the product can become increasingly expensive. Avoiding this situation requires that users effectively communicate their needs through a formal user feedback system and for developers to interpret them correctly so that they are able to satisfy user needs appropriately. Thus, improving user-developer communication and understanding the various social and technical factors that affect this communication would be beneficial for many companies.

As society continues to integrate software systems into daily life, developers and users should both be aware of potential biases in these systems and how they can affect different user groups. Bias can be introduced into a software product throughout its development, including during user-developer communication. It is important to examine how user-developer communication occurs so that bias can be recognized and prevented and developers can create fair, unbiased software products.

REFERENCES

Aydemir, F. B., & Dalpiaz, F. (2018). A roadmap for ethics-aware software engineering. *Proceedings of the International Workshop on Software Fairness, 40*, 15-21. https://doi.org/10.1145/3194770.3194778

Dehaghani, S., & Hajrahimi, N. (2013). Which factors affect software projects maintenance cost more?. *Acta Informatica Medica, 21*(1), 63-66. https://doi.org/10.5455/aim.2012.21.63-66

Friedman, B., Brok, E., Roth, S. K., & Thomas, J. (1996). Minimizing bias in computer systems. *SIGCHI Bulletin, 28*(1), 48-51. https://doi.org/10.1145/249170.249184

Friedman, B., & Nissenbaum, H. (1996). Bias in computer systems. *ACM Transactions on Information Systems, 14*(3), 330-347. https://doi.org/10.1145/230538.230561

Gallivan, M. J., & Keil, M. (2002). The user-developer communication process: A critical case study. *Information Systems Journal, 13*(1), 37-68. https://doi.org/10.1046/j.1365-2575.2003.00138.x

Jantunen, S., Dumdum, R., & Gause, D. (2019, May 25-31). *Towards new requirements engineering competencies*. International Conference on Software Engineering, Montreal, QC, Canada.

Johanssen, J. O., Kleebaum, A., Bruegge, B., & Paech, B. (2019). How do practitioners capture and utilize user feedback during continuous software engineering?. *IEEE International Requirements Engineering Conference, 27*, 153-164. https://doi.org/10.1109/RE.2019.00026

Mohanani, R. (2016, May 14-22). *Implications of requirements engineering on software design: A cognitive insight*. International Conference on Software Engineering, New York, NY, United States.

Oudshoorn, N., Rommes, E., & Stienstra, M. (2004). Configuring the user as everybody: Gender and design cultures in information and communication technologies. Science, Technology, & Human Values, 29(1), 30–63. https://doi.org/10.1177/0162243903259190

Pagano, D., & Bruegge, B. (2013). User involvement in software evolution practice: A case study. *International Conference on Software Engineering, 35*, 953-962. https://doi.org/10.1109/ICSE.2013.6606645

Pinch, T., Bijker, W. (1984). The social construction of facts and artefacts: Or how the sociology of science and the sociology of technology might benefit each other. *Social Studies of Science, 14*(3), 399-441. https://doi.org/10.1177/030631284014003004

Saiedian, H., & Dale, R. (2000). Requirements engineering: Making the connection between the software developer and customer. *Information and Software Technology, 42*(6), 419-428. https://doi.org/10.1016/S0950-5849(99)00101-9

Williamson, A. (2022). *Adapted Social Construction of Technology (SCOT) model for the influence and interaction of the user group in the development of a software product.* [Figure 3]. *Prospectus* (Unpublished undergraduate thesis). School of Engineering and Applied Science, University of Virginia. Charlottesville, VA.

Williamson, A. (2022). *Current system of user-developer communication*. [Figure 1]. *Prospectus* (Unpublished undergraduate thesis). School of Engineering and Applied Science, University of Virginia. Charlottesville, VA.

Williamson, A. (2022). *Published articles in the Association for Computing Machinery (ACM) Digital Library related to requirements engineering and bias.* [Figure 2]. *Prospectus* (Unpublished undergraduate thesis). School of Engineering and Applied Science, University of Virginia. Charlottesville, VA.

Zalewski, A., Borowa, K., & Kowalski, D. (2020). On cognitive biases in requirements elicitation. In S. Jarzabek, A. Poniszewska & L. Madeyski (Eds.), *Integrating research and practice in software engineering* (pp. 111-123). Springer Cham.