Measuring User Preferences for Web Presentations Using Open-Source Tools and a Holistic Approach

Craig Earley
Earlham College Computer Science
801 National Rd W
Richmond, Indiana - 47374
cjearley13@earlham.edu

ABSTRACT

This paper explores human-computer interaction with an emphasis on how an interface affects the human user's behavior. It describes a history of academic and popular research about the topic, which borrows from psychology but occupies a growing body of research in computer science. It then articulates the motivations and design of a software application currently under development that will allow a web developer to easily test these principles with their own websites in a straightforward, rigorous way without the need for a large-scale usability test. It concludes with a look to the future and a few thoughts on the impact of this knowledge for both social and computational sciences.

CCS Concepts

•**Human-centered computing** → **HCI design and evaluation methods;** *User interface programming;* Web-based interaction; HCI theory, concepts and models;

Keywords

human-computer interaction; user interfaces

1. INTRODUCTION

Computer science is about how to structure and process information. The computer is a tool by which humans execute those processes in some way, be it representing mathematical data, encoding visual information for display, or exchanging information with some other computer. In practice, this necessitates a clear understanding how a computer affects its user and an ability to use that understanding to make decisions about design. This subdiscipline is human-computer interaction (HCI), and it is an important and challenging component of the field, and a practical application of HCI is software application design.

At their most basic level, software applications and websites facilitate interactions between a user and some data, which might otherwise be too complex or technical for a user to carry out for themselves. Research into human-computer interaction has shown that changes to the presentation of objects in a user interface (UI) affects how users respond to the information - in other words, design is not value-neutral from the perspective of user behavior.

This concern is modern. Early computers were large, bulky, mechanical, and heavy, relying on users to input punch cards rather than keystrokes and mouse clicks. Smaller computers and command-line interfaces substantially improved the ability to navigate a computer, but only with the widespread distribution of graphical user interfaces (GUI's) did they become a technology for the public. Now the strongest market is mobile devices, which do not use physical keyboards at all and rely instead on touching on-screen buttons, gestures and forces, and voice input.

This paper describes some of the central insights in HCI about the effect of UI's on user behavior. It surveys both computational theory research and the more well-developed literature body of the private sector. It asks questions about whether a UI is well-designed: that is, whether the user behavior it encourages maps well to the needs of its producer and distributor. It also emphasizes the importance of choosing those needs carefully. It describe a piece of software now in development to make programming a UI easier for independent developers. It concludes with a summary and a short analysis of the implications.

2. RELATED WORKS

Consider three layers for understanding this topic. First, theoretical background informs the reasons for its importance and a vocabulary with which to discuss it. Second, the structure of the data underlying an interface, including the basic visual shape in which it is displayed, allows a user to handle it at all. Finally, an interface gives subtle clues to users, often at an intuitive level rather than a logical one, about how they should behave and what choices they should make. It will be shown that, while some application developers aspire to free the user to do as they wish, doing this without imposing any influence at all is likely impossible in practice.

HCI has been examined from a range of perspectives in the academic literature. The following is a summary of the relevant research, including those portions relevant to the project itself. It first discusses the basic theoretical views that have guided researchers in this topic. It then proceeds to show a few brief case studies also conducted in academia. Finally it draws on the popular research in the topic, which is in some cases more extensive and more attuned to the

DOI: 10.1145/1235

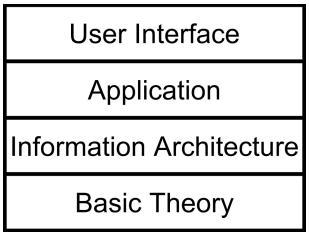

Figure 1: The basic structure of this paper and a way to consider how an application relates to a user.

most modern technology given its commercial applications.

Before continuing, a brief aside is necessary about the discipline of HCI. It is a field that, at the most fundamental level, emphasizes communication. Haynes et al., for example, conducted a series of interviews with HCI design researchers about how design facilitates understanding. They described several established and emerging social theories that affect HCI. Of particular concern in this context is their description of the established, fundamental frames of reference cited by their interviewees with regard to what an explanation is. Most important are causality, instrumentality, and prediction, all of which are relevant to considering user responses within a given interface.[4]

Note that few of these papers carefully investigate specific cases of UI's outside of an academic context, where much of the work is done. Therefore a more thorough discussion of these is the subject of the subsequent section.

2.1 Theory: Nudges and Behavioral Change Support Systems

In the last decade, some psychologists and social scientists have developed the notion of a "nudge" in the design of a system. The most popular example came from Richard Thaler and Cass Sunstein in their book of the same name. A nudge is any change to the design or layout of a system or its environment that changes the behavior of an individual but does not reduce the number of options available to them in that system.[11] Liu et al.'s examination of behavioral change is an example in computer science. They found that, when experiencing a visual warning nudge, users were less likely to click through a link and more likely to employ a "dismiss" feature than when no warning was activated.[7]

Oinas-Kukkonen produced a theoretical framework about how interfaces on a variety of platforms change behavior, with an explicit emphasis on the human's side of the interaction. This research in particular draws on his heuristic series of specific changes an interface may want to produce, and emphasized that this is a cross-platform phenomenon.[9]

Oinas-Kukkonen's paper is worth elaborating in slightly greater length than other sources, as it is among the most detailed academic descriptions of these questions. He defines the notion of a "behavioral change support system (BCCS)" as "[an] information system designed to form, alter, or reinforce attitudes or behaviors or both without using coercion or deception." Note that the definition is not limited to a particular platform or even to computer systems themselves, though this the context he explores.[9]

He specifies three types of changes: changes in attitude, in behavior, or in compliance. Which of these a system is meant to solicit is the "intent" of the system. He notes that the context of the event matters as well, and that designing such a system inevitably requires a strategy that takes into account the computational and social contexts of a potential user. Over time using the system, a user's attitude may also be subject to "reinforcement, alteration or formation."[9]

In his summary, he emphasizes perhaps the most important point of research in this topic: those designing these systems, for whatever platform, are exercising control over their users whether they intend to or not. This is not a harmful process; as Oinas-Kukkonen himself describes, it is inevitable and can produce desirable outcomes for both the designer and the user.[9]

Thus a piece of software has, as one of its functions, behavioral change. Different changes may be part of the design considerations of that software, and they may affect attitude as well as action. This is not a pitfall to be avoided, but a reality to be recognized and considered in the design and construction of a given program.

2.2 Information Architecture and Applications

The first layer up from theory is the structure of the information an application handles. Here information architecture can be understood as both the content and purpose of the data on the backend of a given application. Several researchers have considered these topics in depth. One of these relates to the structure of the information presented to the user: how much should be presented and what form it should take given the needs and constraints of the project.

Because space on a screen is fixed within a given platform, it is necessary to control the factors presented to the user. Cockburn and Gutwin consider cases in which input is constrained - that is, it is not through mouse clicks and typing, where options are virtually limitless and interfaces may be designed as such, but through one factor such as touch in a mobile application. They find that there is a measurable difference between novice and expert navigation within an interface, and - crucially - that different interfaces cause people to respond differently ("intra-interface" and "inter-interface" differences, as it were). Their experiments examine three different interface types to ensure generality. They considered a simple scrolling list, a binary search list, and a binary search grid in which items in the list are placed in boxes in a grid (visually, the first two are one-dimensional, the last is two-dimensional). For each they predicted the time a user would take to complete a given task through them. They made two broad theoretical conclusions:

- Different interface types produce different user behaviors.
- Novices and experts navigate an interface differently.

Thus their model is a useful analytical tool for the structure of information.[3]

Architecture must be an important consideration when designing for both computers and users. A properly-designed data structure provides functionality for a software application that can facilitate some kind of user interaction. When placed in an interface that is accessible, the right information architecture enables users to complete tasks more effectively and quickly. However, the proper architecture is a minimum: a fundamental requirement to a useful website or application, but not one that ultimately determines user choices.

2.3 Interface Design

The next layer of this topic is a brief overview of the design research from the private sector. This is because the incentive for companies to produce good UI's that enable their customers to interact with them creates a strong profit motive to invest in examining interfaces and people's reactions to them.

For example, the classic *The Design of Everyday Things* by Donald Norman approaches product design, digital and physical, from the perspective of usability. He emphasizes the responsibility of the designer, e.g., that "human error" is often a consequence of design flaws instead of clumsiness, that product designers have an obligation to minimize these problems rather than shifting blame to users. Early in the book he describes in some detail the effect of design on the psychology of the user.[8] His insights have in some respects filtered into the more general knowledge of people concerned with design, and it affected the type of research included here.

In interface design as well as in any science discipline, theory and intuition are not sufficient for complete understanding. Experiment is necessary, and for this reason the discipline of quality analysis (QA) and usability testing emerged. Over time designers have discovered the best protocols for carrying out such tests, and their benefits and challenges form the motivation for the software component of this research project.

The combined academic and popular research emphasize some common principles: ease of use, simplicity, and clarity. These design principles guide the general project. Further research will examine more carefully the details of particular features, but for current purposes this general review is sufficient.

3. EXAMPLES

These three layers of the topic can be supplemented by a fourth, which is less academic but demonstrates the insights of computer science on some important use cases. We consider here the iOS design guidelines and how they instruct developers to think about users, and the social network Nextdoor's attempt to solve their site's racial profiling problem.

3.1 iOS Design Guidelines

The hardware-software giant Apple Computer has occasionally published new guidelines for the design of UI'ss within applications. Originally they published them for their Macintosh desktop computer[2], but the most modern version concerns the design of software applications for their operating system iOS for iPhones, iPads, and other mobile devices.

Most sections of the iOS user experience guidelines contain tips about the proven approaches to making an application functional, easy to learn, and pleasant to use. The key themes, repeated throughout, include offering user feedback on actions that make a change to the application, being simple and consistent, and making responses to gestures as intuitive as possible.

While behavioral change is not explicitly described in the guidelines, many of them imply a consideration for impact on user behavior. Mostly, the guidelines hope designers will avoid such impacts, emphasizing that, as a rule, decision-making within a software application belong to the user. From this principle they derive such notions as letting a user cancel a decision, designing with consistency so users know what each of the options presented to them mean (another section is titled "WYSIWYG [What You See Is What You Get]"), and using suggestions rather than requirements to guide behavior. [1]

This is in keeping with the value of a user-centered experience, the running theme of the guidelines (and Apple's public relations). It is a call for simplicity, clarity, and user control. This value and the principles which follow from it also feature in the research literature: the findings of Kalnikaite et. al., who found that an overwhelming UI presenting too many options or indicators is excessive and slows users down.[5]

3.2 Nextdoor

In general, design emphasizes users making a series of decisions quickly and effortlessly. Given millions of customers, shifts that change the behavior of even one percent of the customer base's purchasing decisions can substantially affect corporate incomes. However, speed and likelihood of transaction are by no means the only factor design affects.

Consider, for example, the case of Nextdoor, a social networking site for neighborhoods. They faced negative coverage when some users noticed that the platform hosted racial profiling: specifically, reports of threatening or suspicious behavior observed in a neighborhood overwhelmingly described suspects as "dark-skinned" or similar, and often included little or no additional information. Pushback threatened the reputation of the site and they sought to solve the problem.[10]

To solve the problem, they made posting such messages slower and more difficult rather than easier. A user posting a message that specified race (see Figure 2) were required to also fill out some subset of the other fields in the post. In this case, the appearance of their interface did not change, but it required the user to provide additional information. As Shanani explained for National Public radio, this is extremely uncommon for social media companies, who prefer to make posting as quick, easy, and effortless as possible.[10] (The iOS design guidelines also emphasize this principle in a section titled "Minimize the Effort Required for User Input"[1]).

This approach is atypical for such tools, and it is notable in that it explicitly constraints user choice in a way that most interfaces do not. However, it is in keeping with other important design principles. The iOS guidelines, for example, emphasize feedback, and one function of Nextdoor's change was to provide feedback about a user's action. It is also a particularly strong instance of a recurring theme in design: because design decisions affect user decisions, they must be made consciously, and when systems do not work it is fundamentally the responsibility of the designer to improve their design. Information architecture as well as interface design must be carefully considered, their initial use tested, and their future results monitored to ensure that such mistakes are avoided or corrected.

4. SOFTWARE APPLICATION

4.1 Usability Testing

Ultimately, examples and guidelines are not sufficient to determine the efficacy of a UI. This requires a usability test.

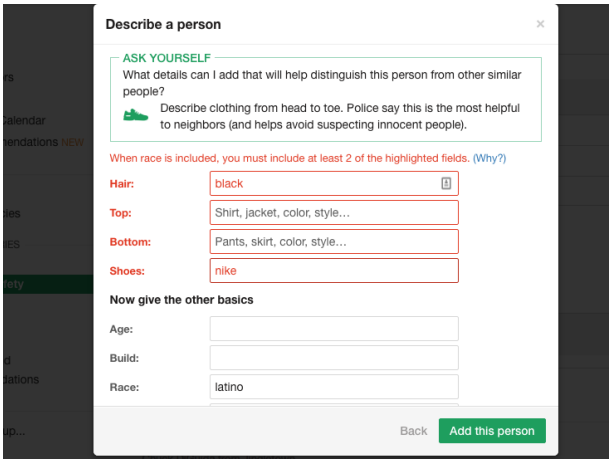

Figure 2: Screenshot from NPR. Note that, other than additional text, the interface is the same before and after the failed action.

In his concise popular book *Rocket Surgery Made Easy*, Krug described the idea and execution of a usability test. In broad strokes, the designer of an interface authors a test case, recruits several volunteers, hosts them in a single room, and observes them as they navigate. They ask questions, observe decisions, and take careful. The authors also set up the space and often provide refreshments.[6].

The procedures for such a test are well-defined and consistent, but in aggregate they are cumbersome and potentially expensive. A large company will dedicate funding to both time and money to do it, because they have a direct financial stake in its success. However, for an independent programmer lacking a large organization or an easily-mobilized social network, it may be impractical. Academics and developers at a smaller scale may prefer less friction in return for less specificity, particularly early in the development stages. The tool described in the following section is designed to fill that niche.

4.2 The Problem

The research supports the basic assumption of the software package currently in development: design of interfaces is of critical importance. This is true for the private sector for obvious reasons: increased click-through rates, more pleasant experiences, and greater command of user behavior are critical objectives to increasing the financial success of companies. However, it is of value in academia as well: academics in numerous disciplines wish to guide their students, research participants, and others to succeed in a given set of goals. Academics lack access to the often proprietary resources of the private sector. This is an even more acute problem for students and independent developers, who lack the resources for usability testing of any scale.

As stated, major firms have application programming interfaces (API's), internal guidelines, the credibility to organize testing and focus groups, and other advantages that those not employed by them do not enjoy. The application of the principles of human-computer interaction, thus, require extensive setup work. This can slow the progress of research. Furthermore, many academics outside computer science will find the maze of resources impenetrable without dedicating substantial extra time to them.

This section describes a framework in which independent developers may do some basic, small-scale testing independent of their geographic and institutional constraints.

4.3 Motivations for the Chosen Approach

Technology increasingly disconnects work from geography and institutional constraints. Every academic now has access to the Internet and make frequent use of it in other contexts through a browser, making it an approachable technology for them. In many cases, at extremely local scales, simply referring to an extension, sending a little bit of information out via email, and waiting twenty-four hours to collect anonymized data may serve their purpose.

Software currently under design and early implementation can channel both the strengths and the constraints of being an independent developer and tester. Having demonstrated the significance of allowing developers to investigate interface design quickly and easily, parts of the project are straightforward to construct.

I propose a framework for conveniently choosing, displaying, and analyzing interface options for comparison, for use by smaller organizations, academics, and other developers not affiliated with major technology firms. For now, this will focus on speed, the feature most easily checked using built-in system features and accessible software.

It should be emphasized that this solution is not preferable to a full-scale usability test or A/B test. Given the proper resources, such tests are superior and can produce substantially greater insights. This project also focuses exclusively on compliance, to use Oinas-Kukkonen's language,[9] though as will be described later this software could easily examine behavior change as well. The vision for this project is to allow projects on an individual or small-group scale to be quickly and easily tested. As they scale, more robust tests will still be necessary.

4.4 Hardware

Developing this software requires minimal hardware. It will make use of the Earlham College Computer Science Department's Linux cluster computing system, as well as a local Apple MacBook Air laptop, to design and implement this tool. These computers include an apache web server and other system tools in the event that later stages of the project require custom web pages for testing. It also features the PostgreSQL database infrastructure in which this data will be stored (for more about this, please see the Software and Data Collection sections).

4.5 Software

The solution under development is a browser extension for Google Chrome. It will enable A/B testing on a small scale using open-source tools, programming languages most developers know or have been exposed to, and an accessible platform.

An extension was chosen for several reasons. Perhaps the most important is that designing and building them (relative to developing an entirely new, platform-agnostic application) is straightforward. Chrome (and Firefox) extensions have a robust API ecosystem and an active developer community.

Second, every computer user has web access and can easily download it, so such disparities as iOS/Android in mo-

bile applications do not apply. As a result, an extension is portable, easily deployed, and (upon completion) easily discovered by others.

Third, user behavior on the web is a ubiquitous case in both commerce and academia. Desktop web users are not the fastest-growing sector in consumer and commercial technology - mobile device users are - but the browser is still the mechanism through which many users access other people's content on a computer. Together, these benefits make this project simple and broadly useful.

The code is stored in a git repository in the Earlham Computer Science Department's GitLab account for version control. The details of the software are continually updated there, including sparse but clear documentation in the `README.md` file.

The extension has two modes. One is a small author-side tool containing two simple text boxes. The first asks for start URL's, the second for stop URL's. In later versions of the project, minimal metadata may also be collected (see Data Collection). Upon a click, the software would output a unique URL that can be shared. It will also store in the entry in a PostgreSQL database. See Figure 3.

The output URL would activate the user-side face (a user could also activate it by clicking the icon and choosing "User Mode" or similar). Upon being opened, the URL would randomly choose one of the input URL's provided by the author, send the user to that URL, and begin tracking user progress. Upon reaching one of the stop URL's, it would prompt the user to approve saving the data to the author's records.

This could then be exported for quick A/B testing of web features of either a large or small scale. Any data collected would be anonymized and stored in a CSV.

It will also employ a small PHP program which connects to a PostgreSQL database, run on the cluster computers, to store the small amounts of data necessary for the project. This is described below.

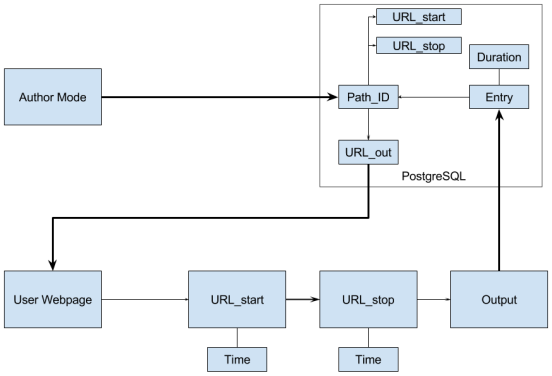

Figure 3: Flow diagram for the extension. Bold lines indicate that the action is taken by the extension.

5. RESULTS AND CURRENT STATE

The software as it currently exists is a base case, in which a test author provides two start URL's and one stop URL, which are written to a text file. On activating the browser extension, a user is navigated to one of the suggested pages and should begin regular web interaction with the goal of reaching the stop URL. The extension begins a timer when the user starts the extension, ends it when they reach the stop site, and saves the information about that transaction to a new file. The code is publicly available on the Earlham CS Gitlab website.

Due to unforeseen complications in development, the project has not produced data that can be analyzed. However, on installing and running the extension, the time required to move from start to stop based on the pre-defined path is displayed.

Please see the Future Work section for details about the necessary improvements to the extension that must be built before it can be deployed for use.

5.1 Data Collection

This project may necessitate a request to the members of the class to examine the software in its base case, where it considers only a small set of start URL's and a single stop URL. They will go through the full process, including anonymous timing and data recording, as they each complete the process for this one circumstance. The anonymized data can then be analyzed using basic statistics to see which interface tended to be completed more quickly, if any.

When the project scales to in-world use, the data collection question must be answered in a way that preserves privacy, collects data in a minimally invasive way, and is transparent about how it stores the data without opening it up to security breaches.

6. CONCLUSION AND FUTURE WORK

6.1 Progress of Human-Computer Interaction

Because of the financial interest in continued development in this area, research into how to use design to affect user behavior will likely continue. Small increases in click-through or purchase rates may translate into substantial added returns at scale, more than compensating for the labor cost of designing and upgrading the interface components of software stacks.

More generally, UI designers and developers have proven adept at adapting to new hardware and operating systems. Companies now employ developers for platforms well beyond the web, especially mobile devices running on Apple's iOS and Google's Android operating systems. The research presented here is general enough to guide choices in all of these contexts. The software, as described below, is substantially more limited but will remain useful as long as websites are a critical mode of interaction for the public with their companies.

6.2 Future Software Development

The conception of the extension is this project is slightly different from its current configuration, in that it was meant to measure not completion rates but rates of user choices - e.g., given a set of starting URL's and a set of stopping URL's, which stopping URL does a user tend to land on? Ideally the start URL's will be different versions of a company's web tool and the URL's will be different webpages within the same site. This data could be collected with similar anonymization. Focusing the project on questions of

choice present opportunity for other design considerations not yet explored.

There are less intensive considerations as well. For example, this tool should be developed for browsers other than Chrome. Firefox has a strong development community and many "add-ons" of its own and would be a likely candidate for a followup application. However, to gain any real audience in the future will require some mobile presence, likely in the form of a companion app for each iOS and Android, which would enable small-scale testing of application interfaces as well as webpages. Because of the complexity of working with multiple operating systems and with entirely different navigation methods on mobile, this component is likely more complex than adapting it for different browsers.

Finally, this is a tool that need not be limited to programmers. At a high level, all a test author must provide is a set of start URL's and a set of stop URL's. Academics in other disciplines, such as psychology, often need to perform A/B-style tests to collect data. As a result, they would find such a tool, once built and scaled, easy to work with for purposes otherwise mostly removed from computing.

6.3 Social Significance

This research was inspired by existing research by psychologists and behavioral economics about the large effects of small changes. That research, not explored in this paper, focuses on the social sciences, public policy, and the counter-intuitive behaviors of the human mind. Now that computers are a central, if not defining, feature of the modern world, a critical topic of research in the future must incorporate computer scientists into this research. In turn, computer scientists must recognize that their work has spillover effects in these disciplines and in social life generally.

UI programmers specifically must be mindful that the decisions they make affect the decisions users make. This is not a condemnation but a recognition of it as an almost tautological fact of computer interfaces. The specific changes these programs drive may be positive, negative, or value-neutral, but they must be considered as part of the design and testing processes. (Indeed, if they did not make a difference there would be no need to hire people to build them.) Software such as that proposed here, coupled with the research purpose described above, provide ample opportunity for such discoveries.

This research also highlights the importance of computer scientists interacting with those outside their discipline. The lessons here come from fields as far-reaching as design, behavioral economics, and psychology, in addition to computer science, data modeling, and web development. Technology is designed for people, of whom technologists and computer scientists are one subset. The field of HCI and the specific topic of interface design emphasizes this fact.

7. ABOUT THE AUTHOR

Craig Earley is a Computer Science major at Earlham College and will graduate in December 2016. This is his Senior Capstone paper.

8. ACKNOWLEDGMENTS

The author would like to thank Xunfei Jiang and Charlie Peck, Computer Science Professors at Earlham College, for their advice and instruction in completing this project and his degree.

9. REFERENCES

[1] ios human interface guidelines.

[2] Macintosh human interface guidelines.

[3] A. Cockburn and C. Gutwin. A model of novice and expert navigation performance in constrained-input interfaces. *ACM Trans. Comput.-Hum. Interact.*, 17(3):13:1–13:38, July 2010.

[4] S. R. Haynes, J. M. Carroll, T. G. Kannampallil, L. Xiao, and P. M. Bach. Design research as explanation: Perceptions in the field. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '09, pages 1121–1130, New York, NY, USA, 2009. ACM.

[5] V. Kalnikaitė, J. Bird, and Y. Rogers. Decision-making in the aisles: Informing, overwhelming or nudging supermarket shoppers? *Personal Ubiquitous Comput.*, 17(6):1247–1259, Aug. 2013.

[6] S. Krug. *Rocket Surgery Made Easy: The Do-It-Yourself Guide to Finding and Fixing Usability Problems*. New Riders Publishing, Thousand Oaks, CA, USA, 1st edition, 2009.

[7] J. Liu, S. Ruohomaa, K. Athukorala, G. Jacucci, N. Asokan, and J. Lindqvist. Groupsourcing: Nudging users away from unsafe content. In *Proceedings of the 8th Nordic Conference on Human-Computer Interaction: Fun, Fast, Foundational*, NordiCHI '14, pages 883–886, New York, NY, USA, 2014. ACM.

[8] D. A. Norman. *The design of everyday things*. Basic Books, revised and expanded edition edition.

[9] H. Oinas-Kukkonen. A foundation for the study of behavior change support systems. *Personal Ubiquitous Comput.*, 17(6):1223–1235, Aug. 2013.

[10] A. Shanani. Social network nextdoor moves to block racial profiling online. *National Public Radio*, 2016.

[11] C. Sunstein and R. Thaler. *Nudge: Improving Decisions About Health, Wealth and Happiness*. Penguin Books Limited.