Introduction
Software

Carl DiSalvo

Software is not a new object of inquiry for STS, but it is fair to say that our relationship with software, as both users and researchers, is changing. This handbook itself is a prime example. Software played a significant role in producing and organizing these chapters. We deployed php code for open online peer review to produce a collaborative space for our community's early proposals and essay drafts. Later, coeditor Yanni Loukissas, working together with Ben Sugar (then a graduate student at the Georgia Institute of Technology), analyzed the text of all of the essays using topic modeling algorithms to identify themes and form groupings, aiding in the decision of how to organize the essays in the volume. These software deployments were not rote processes, but involved experimentation, discussion, a bit of hesitancy, and finally some decisions. This is emblematic of our contemporary engagement with software as both commonplace and a social practice.

The ordinariness of software, its social construction, and its practices are themes that span the chapters in this section. These essays attend to the vernacular aspects of software. Although we live, work, and play with software, despite its many fantastic capacities, it is a rather mundane thing. That software is commonplace, however, does not mean it is not a worthy object of study. Rather, its ordinariness opens us up to new encounters and themes of inquiry. Building upon foundational work understanding how software shapes and is shaped by works practices in science and technology (Star 1995; Bowker and Star 2000), we now see an increasing number of investigations into the material characteristics of software (Rosner 2018; Dourish 2018), the roles of software throughout society, including financial markets (MacKenzie and Spears 2014) and governance (Introna 2016; Ziewitz 2016), and, increasingly, the political character of algorithms (Gillespie 2014; Crawford 2015).

The chapters in this section share a commitment to analyzing the practices that compose software: that is, how software works through associations, interactions, and performances between and among individuals, groups, organizations, and code. This line of inquiry connects to scholarship that examines the variety of ways that software is made and the cultures of its making, from historical inquiries (Turner 2010) to studies of free and open software (Kelty 2008; Coleman 2013) to global do-it-yourself and so-called "maker" communities (Lindtner 2015). It is perhaps the attention to the diverse practices of making, using, and maintaining software that distinguishes STS work in this area from other contemporary studies of software as cultural objects.

In "From Affordances to Accomplishments: PowerPoint and Excel at NASA," Janet Vertesi introduces readers to the use of everyday software packages in the context of space science. As Vertesi notes, although there is significant work examining the custom and expert tools of science, less attention has been directed to studying the use of the run-of-the-mill suites of software that permeate much of the work of science and technology research and development. In particular, Vertesi analyzes the concept of affordances in software design. She argues that discussions of affordances are "logically problematic" and their "explanatory capabilities are thin" because, in fact, affordances are not universal but specific to and differentiated among varied communities of users and contexts. Vertesi suggests a renewed commitment to attending to the situatedness of working with software, interrogating how software is used to order work, without giving precedence to any particular functionality or pattern of use.

"The Role of 'Misuses' and 'Misusers' in Digital Communication Technologies," by Guillaume Latzko-Toth, Johan Söderberg, Florence Millerand, and Steve Jones, returns to a critical theme in STS: how the actual use of technologies often extends the normative notions of its "proper use." The authors trace misuse and misusers through the three historical cases of PLATO (an educational platform), Bitnet (an international network for liberal arts faculty), and Internet Relay Chat (IRC). From across these cases, the authors proceed to identify the character of a "misuser" as an active, performative figure in the innovations process. In addition, the misuser can operate as an analytic category. Situating this figure and category of the "misuser" in STS literature, the authors go on to describe how the actions of a misusers are relational to what they term as the "plasticity" of a technology. The result of this inquiry, then, is a new perspective on innovation, situated in regard to digital communications technologies, that highlights misuses and misusers who resist the normative framings of "proper use."

Nick Seaver's chapter addresses a fundamental component of software: the algorithm. He begins "Knowing Algorithms" with the insight that algorithms should be (but often are not) treated as social constructions. Those who study algorithms regularly claim that lack of "access" and "expertise" hinders our understanding and evaluation algorithms, but these claims are dubious, "where facts are simple objects to be retrieved from obscurity and analyzed by ever more sophisticated experts." Through the chapter, Seaver carefully builds an argument for an ever more nuanced appreciation of algorithms, addressing key themes of experimentation and transparency. This leads Seaver to the conclusion that, from an ethnographic perspective, our focus should be on what he terms "algorithmic systems—intricate, dynamic arrangements of people and code." Pushing this idea further still, Seaver argues against the notion of a "cultural/technical distinction as a ground for analysis." In many ways, Seaver's insights into studying algorithms recall the features of a multisited ethnography (Marcus 1995), in which we are charged to follow and interpret the varied sites and practices of humans and nonhumans together in the construction and expression of algorithmic systems.

How do we study software as a thing that changes over time? This is the question that motivates Marisa Leavitt Cohn's chapter "Keeping Software Present: Software as a Timely Object for STS Studies of the Digital." Though we commonly think of software as new, as Cohn points out, much of the software that we rely upon for our most needed work are legacy systems, sometimes decades old. For Cohn, one implication of studying the temporalities of software is that we attend to "its forms of duration, the entangled lifetimes of careers, professional identities, program-

ming languages and paradigms, all of which come and go." The phrase "keeping software present" speaks to this notion of duration, of "bringing old code together with new." Throughout the chapter Cohn describes the ways in which code is *lived*, the ways that engineers, managers, users, and others, in multiple and varied ways, endeavor to keep software working. For Cohn, these endeavors are part of the materiality of software, and like other materialities, possess distinctive temporalities: "its lifetimes, histories, and changes as drag, drift, and forgetting."

Taken together, these inquiries into software, its analysis, and its discontents offer opportunities for scholarship that examines the everydayness of software systems and their associated practices, while at the same time troubling our everyday distinctions between use and misuse, culture and technology, social and material, the novel and the aged.

Works Cited

Bowker, G. C., and S. L. Star. 2000. *Sorting Things Out: Classification and Its Consequences*. Cambridge, MA: MIT Press.

Coleman, E. G. 2013. *Coding Freedom: The Ethics and Aesthetics of Hacking*. Princeton, NJ: Princeton University Press.

Crawford, K. 2015. "Can an Algorithm Be Agonistic? Ten Scenes from Life in Calculated Publics." *Science, Technology, & Human Values* 41:77–92.

Dourish, P. 2018. *The Stuff of Bits: An Essay on the Materialities of Information*. Cambridge, MA: MIT Press.

Gillespie, Tarleton. 2014. "The Relevance of Algorithms." In *Media Technologies: Essays on Communication, Materiality, and Society*, edited by Tarleton Gillespie, Pablo Boczkowski, and Kirsten Foot, 167–95. Cambridge, MA: MIT Press.

Introna, L. D. 2016. "Algorithms, Governance, and Governmentality: On Governing Academic Writing." *Science, Technology, & Human Values* 41 (1): 17–49.

Kelty, C. M. 2008. *Two Bits: The Cultural Significance of Free Software*. Durham, NC: Duke University Press.

Lindtner, S. 2015. "Hacking with Chinese Characteristics: The Promises of the Maker Movement against China's Manufacturing Culture." *Science, Technology, & Human Values* 40 (5): 854–79.

MacKenzie, D., and T. Spears. 2014. "'The Formula That Killed Wall Street': The Gaussian Copula and Modelling Practices in Investment Banking." *Social Studies of Science* 44 (3): 393–417.

Marcus, G. E. 1995. "Ethnography In/Of the World System: The Emergence of Multi-sited Ethnography." *Annual Review of Anthropology* 24 (1): 95–117.

Rosner, D. 2018. *Critical Fabulations: Reworking the Methods and Margins of Design*. Cambridge, MA: MIT Press.

Star, Susan Leigh. 1995. "The Politics of Formal Representations: Wizards, Gurus, and Organizational Complexity." In *Ecologies of Knowledge*, edited by Susan Leigh Star, 88–118. Albany: State University of New York Press.

Turner, F. 2010. *From Counterculture to Cyberculture: Stewart Brand, the Whole Earth Network, and the Rise of Digital Utopianism*. Chicago: University of Chicago Press.

Ziewitz, Malte. 2016. "Governing Algorithms: Myth, Mess, and Methods." *Science, Technology, & Human Values* 41 (1): 3–16.