Eight Observations and 24 Research Questions About Open Source Projects: Illuminating New Realities

MATT GERMONPREZ, University of Nebraska at Omaha, USA
GEORG J.P. LINK, University of Nebraska at Omaha, USA
KEVIN LUMBARD, University of Nebraska at Omaha, USA
SEAN GOGGINS, University of Missouri, USA

The rapid acceleration of corporate engagement with open source projects is drawing out new ways for CSCW researchers to consider the dynamics of these projects. Research must now consider the complex ecosystems within which open source projects are situated, including issues of for-profit motivations, brokering foundations, and corporate collaboration. Localized project considerations cannot reveal broader workings of an open source ecosystem, yet much empirical work is constrained to a local context. In response, we present eight observations from our eight-year engaged field study about the changing nature of open source projects. We ground these observations through 24 research questions that serve as primers to spark research ideas in this new reality of open source projects. This paper contributes to CSCW in social and crowd computing by delivering a rich and fresh look at corporately-engaged open source projects with a call for renewed focus and research into newly emergent areas of interest.

CCS Concepts: • **General and reference** → **Surveys and overviews**; • **Information systems** → *Open source software*; • **Human-centered computing** → Field studies; *Open source software*; Computer supported cooperative work; • **Software and its engineering** → **Open source model**; • **Social and professional topics** → Socio-technical systems;

Keywords: Open Source; Corporate-Communal Engagement; Tales from the Field

ACM Reference Format:
Matt Germonprez, Georg J.P. Link, Kevin Lumbard, and Sean Goggins. 2018. Eight Observations and 24 Research Questions About Open Source Projects: Illuminating New Realities. In *Proceedings of the ACM on Human-Computer Interaction*, Vol. 2, CSCW, Article 57 (November 2018). ACM, New York, NY. 22 pages. https://doi.org/10.1145/3274326

1 INTRODUCTION

The amount of work contributed to open source projects is exploding. These contributions include open source software[1] ("the code"), documentation, testing, community participation, and issue logging. The production of open source software is the principle aim of any open source project. Sometimes, open source projects are grouped together in open source ecosystems organized

[1] "Free and Open Source Software (FOSS)", "Free/Libre Open Source Software (FLOSS)" and "Open Source Software (OSS)" are all referred to as 'open source software' in this paper.

Authors' addresses: Matt Germonprez, University of Nebraska at Omaha, 6001 Dodge Street, Omaha, NE, 68182, USA, germonprez@gmail.com; Georg J.P. Link, University of Nebraska at Omaha, 6001 Dodge Street, Omaha, NE, 68182, USA, glink@unomaha.edu; Kevin Lumbard, University of Nebraska at Omaha, 6001 Dodge Street, Omaha, NE, 68182, USA, klumbard@unomaha.edu; Sean Goggins, University of Missouri, 113 Naka Hall, Columbia, MO, 65211, USA, outdoors@acm.org.

2573-0142/2018/11-ART57
https://doi.org/10.1145/3274326

around shared interests and resources. For example, the Linux Foundation ecosystem, valued at $25 billion in 2008 [42] (and likely worth substantially more now), is organized around open source software projects that enhance or advance technologies adjacent to or built on top of the Linux operating system, and today includes 85 projects and over 1000 corporate members. Open source ecosystems often collect and advance the interests of mostly for-profit organizations, and through this engagement are reshaping open source software and eroding the salience of previous descriptions and research. Our understanding of open source projects and the role of for-profit engagement in open source ecosystems is largely anchored in earlier studies focused on subsets of projects or emerging, new project types like front office vertical applications. The acceleration of corporate investment in open source projects and ecosystems is altering open source in dramatic ways, just as great accelerations in growth reshape countless other dynamic, human systems throughout history [55].

Corporate engagement in open source projects is a kind of corporate-communalism, a construct that combines the profit values corporations carry and their integration with more communal structures common to open source projects [27]. Substantial prior work examining open source projects is spread across literature in the various disciplines engaged in the multi-disciplinary social computing and CSCW research. The cacophony of interdisciplinary findings about open source projects suggests that: a) discipline specific theories and methods draw out different, sometimes conflicting insights and b) open source software may not be a "single thing" at all. Prior work is sometimes in conflict across studies, and increasingly not representative of how corporate-communalism is reshaping open source projects. The great acceleration of open source and corporate engagement in open source projects may, at least partly, underly tensions in the literature. One possibility is that our ontology is too limited. If "open source" is, in fact, a complex collection of interrelated phenomena, contemporary studies must lead us to a richer language for distinguishing the categories of and connections within this sociotechnical space.

In this paper, we focus on a longitudinal study of the Linux Foundation and its open source project ecosystem, explicating a set of observations and research questions for open source software that recognizes the complexity and possible ontological limitations of work to date. We illustrate how the world of open source is, in all likelihood, not a single phenomena, by focusing on the role that foundation support and corporate strategy play in the evolution of open source projects. Our ongoing research directly involves us with open source projects supported by the Linux Foundation, such as the Software Package Data Exchange (SPDX) project and the Community Health Analytics Open Source Software (CHAOSS) project. Our engaged field research is presented here in a succinct set of observations, or "tales from the field" [58]. Our findings conceptually frame the open source project phenomena around eight observations which color in the details of how open source projects are operating in a rapidly changing ecosystem. Our eight observations become an organizer for 24 research questions intended to spark ideas for further investigation. Our conclusion reinforces the notion that open source is one category and context among many in the accelerating space of technology mediated work.

Our goal is to apply empirical insights from our field work to draw out new ways for social computing and CSCW researchers to consider the dynamics of open source projects as a site for research. CSCW once centered on collaborative systems inside organizational boundaries. The rapid acceleration of individually controlled collaboration tools fundamentally reframes the field as social computing, which includes more specific constructs like open collaboration, peer production, social media, and citizen science. Open source projects are presently in the middle of a similar transformation as they become more complex, interrelated, and a preferred method of professional software development with significant implications for the future of work.

Category	Description	CSCW conference proceedings
Conceptual	Studies that provide a descriptive account of open source, talk about its potential and realized advantages and disadvantages, compare open source with proprietary software, and attempt to provide categorizations and models of existing research as well as agendas to shape future research efforts.	—
Performance Metrics	Studies that investigate quality aspects and efficiencies of open source, performance of open source development teams and determinants of success for open source projects.	[3]
Legal and Regulatory	Studies that talk about licensing of open source software applications, intellectual property rights and implications, various legal issues, and the importance/implications of standardization and regulation.	—
Production	Studies that address various aspects of the production of open source applications. This covers a number of input related topics such as developer motivations and team formation, process topics such as the development process itself, and governance, as well as a few output related topics such as learning and community evolution.	[9, 18, 19, 23, 30, 56, 59, 65]
Diffusion	Studies on the technology outputs of open source development process and that address various aspects of open source adoption and implementation.	—
Beyond Software	Studies that explore implications of OSS over and beyond the software development domain. In general, these categories investigate the possibility of the applicability of OSS style organization and work practices to other domains and their implications for users.	[34, 39, 53]

Table 1. Open source research categories by Aksulu and Wade [1].

2 PRIOR WORK ON OPEN SOURCE

Literature on open source projects span a number of academic disciplines. Open innovation literature from Henry Chesbrough [5], Eric von Hippel [60], and Georg von Krogh [61, 62] is widely cited across disciplines. In Information Systems research literature, workplace and organizational frames for open source research are employed by Kevin Crowston [7], Siobhán O'Mahony [45], and Joseph Feller [16]. In the ACM and CSCW literature, Laura Dabbish [9], Kevin Crowston [8], Paul Dourish [13], and Walt Scacchi [50] take on questions pertaining to teamwork in open source software engineering projects.

Aksulu and Wade [1] provide a framework to categorize the variety of open source research, incorporating other overviews of the open source literature, including Scacchi's [50]. Their framework is summarized in Table 1 and includes six categories (we excluded a seventh category — *applications* — which is a list of industries that use open source software and is of limited interest for the present study). Applying the framework from Aksulu and Wade, our paper falls within the *conceptual* category because we aim to shape future research efforts by clarifying the ontology and diversity of open source projects in organizational contexts, yet our specific observations provide paths to explore additional categories from Aksulu and Wade.

We leverage the Aksulu and Wade framework [1] to situate what we know about organizational engagement with open source projects [12, 16, 21]. A recent study shows that more than 50% of development in open source projects is now done by paid developers [48] and organizations are dominating major open source projects [36]. The acceleration of corporate engagement in open source projects in all likelihood adds layers of complexity or alters the reality of open source as we understand it, given what we know about corporate processes and structure from seminal literature.

Prior work has explored the emerging effects of corporate engagement in open source projects. These studies have an important focus on the *production* of open source software and show that open source projects are complex design engagements [27, 33] consisting of individual participants, corporate employees, foundations, and universities organized around a shared focus and practice [15, 35] and set against a backdrop of innovation [5]. As corporations engage with open source projects, prior studies describe how corporations make use of open source software [10–12, 15, 50] (*diffusion*), the organizational structures that emerge around open source projects with corporate engagement [16], and open source business models [21] (*beyond software*). Complexities and differences between open source projects with and without organizational engagement are also evident in research on *legal and regulatory* issues [24], highlighting licensing decisions from individual and organizational views [57]. From this literature, we have a base from which to see that the practices within open source projects are evolving in significant ways that are altering the sociotechnical dynamics and mechanisms that underlie our prior understanding of "how open source projects work".

Fitzgerald [21] describes a shift away from horizontal, infrastructural focus of early open source software. Corporate involvement in Fitzgerald's frame centers on providing services to augment open source projects applied in vertical domains like front office applications. In parallel, Elliott and Scacchi [15] frame the evolution of open source projects as a profession that draws interest partly from a technological, Utopian vision of the future of work and partly from the pragmatic evolution of technology focused professional communities. These studies, in turn, take up the corporate and developer perspectives on how open source projects operate and evolve. Scacchi [50] goes a step further and contrasts open source project practices with more organizationally bound software engineering work. The collected perspectives of prior work to understand open source projects as a social computing phenomenon centers on constructs like cooperation, coordination, social movements, and more broadly "sociotechnical systems". Each perspective characterizes some form of distributed social structure emerging around open source projects, which essentially is a form of "networked teamwork" [20].

Howison and Crowston [33] suggest the networked, organizational lens for open source development no longer provides an empirically verifiable representation of how open source project work is accomplished. They show, instead, that the "overwhelming majority" (their words) of open source software development is performed by individuals. Further, they illustrate how large tasks, those so complex that teamwork is an a priori necessity for a solution, are usually postponed until they appear easier; and are implemented mostly by individuals. Howison and Crowston's [33] organizational view, Fitzgerald's [21] description of how open source projects are changing, and Scacchi's [50] synthesis of observed open source project and corporate development show the literature to house well-executed studies that, viewed as a whole, can produce an initial set of valuable conclusions that inform a corporate-communal perspective of open source projects. However, as open source projects are ever-evolving, research must stay attuned to this perpetually shifting landscape.

Software engineering and CSCW conference literature add an up-close perspective on open source project dynamics. Software engineering-focused CSCW research identifies hidden coordination requirements evident from the analysis of artifact (code) access [3]. The transformation of some open source projects into more social, less organizationally centered models of engagement is also shifting the metrics individual contributors look at from raw metrics like "commits" to more socially engaged metrics like "outside press attention" [41]. Mens [43] likens contemporary open source software to an ecosystem of complexly intertwined projects and products. Social computing research connects to the software engineering and organizational science perspectives by showing how open source projects are becoming, at least partly, social places and how, specifically, "social

coding" on sites like GitHub is transforming some open source projects and making them more accessible for a wider audience [40].

One important field of research where we would expect this new norm of how open source projects work to be reflected is in the CSCW conference proceedings. CSCW has a history of studying social computing phenomena in open source projects, which are among the longest running exemplars of open collaboration. We explored the CSCW conference proceedings[2] using the Aksulu and Wade framework (Table 1) to ground our search results and demonstrate the variety of open source research in CSCW. Most of the research in CSCW explored collaboration and coordination in open source projects from a software engineering perspective and falls in the *production* category, specifically as it examines collaborative design engagements [65], conflict resolution [18, 19], group awareness in development teams [30], and the transparent nature of open source software development [9]. Further work has explored how open source projects have social barriers for newcomers [56], share knowledge [59], and build shared understanding [23] in the development of open source software. Additionally, research in CSCW looked at *performance metrics* with a focus on methods for identifying coordination requirements in complex open source projects [3]. Regarding *beyond software*, CSCW explored interactions of open source software user groups [53], incentives for producing scientific open source software [34], and how activity traces are used as signals for recruiting and hiring developers [39]. We found no papers in the CSCW conference proceedings that fall clearly in the categories *conceptual*, *legal and regulatory*, and *diffusion*.

Our review of the CSCW conference proceedings is not meant to be definitive but it does reveal that the new realities of corporate-communal engagements are not fully reflected in CSCW research on open source projects — where the conceptual view of open source projects with "largely voluntary nature of participation" [18, p. 1393] is still prevalent. This view, while accurate for some projects, fails to account for organizational coordination and collaboration mechanisms that have been introduced into many projects. Not fully representing the implications of corporate-communal engagements for the social computing aspects of open source software limits our ability to connect from it and inform a larger literature base. To advance the CSCW literature and "generalize the open-source approach to ordinary organizations" [65, p. 337], we have to overcome acknowledged limitations such as not considering how "organizational sponsorship of open source projects might affect their social dynamics" [18, p. 1402].

Updating our understanding of open source projects as a social computing phenomena is critical not only for its own sake, but because previous studies of open source projects are sometimes operationalized in literature as a conceptual anchor that informs work in other collaborative settings [e.g., 17, 32, 37, 40, 51] or serves as a site of study to inform concepts that are not unique to open source [e.g., 30, 31, 63, 66]. Those studies do not advance our understanding of open source projects specifically, but they build their conceptual cases using a potentially outdated understanding of "how open source works". Our findings emerge from this foundation and both shape an updated conceptual understanding of open source projects and highlight areas of research that could benefit from a renewed focus derived from our observations and questions, framed across the categories in the Aksulu and Wade framework [1].

3 METHODS AND DATA

Our observations are the result of an ongoing, eight-year study exploring organizational engagement with open source projects. To explore this engagement, our efforts are localized in open source

[2]We searched full paper CSCW conference proceedings in the ACM digital library without date constraint for "open source", excluding posters, workshops, panels, doctoral consortia, videos, and demos.

projects that include heavy organizational engagement, particularly projects brokered by the Linux Foundation, a 501(c)(6) trade association. The Linux Foundation has helped "establish, build, and sustain some of the most critical open source technologies fostering innovation in every layer of the software stack. The Linux Foundation hosts projects spanning enterprise IT, embedded systems, consumer electronics, cloud, and networking".[3] By exploring open source projects at the Linux Foundation, we explicitly positioned ourselves at the junction of organizational engagement with open source projects, building the "interpretive nerve" that are our observations and questions [58, p. 23].

Within this context and over the prior eight years, we employed a variety of approaches including participant observation, group informatics, direct engagement, and critical reflection. Participant observation was used as a field-based approach when members of our research team were directly engaged in the practices we sought to understand [54, 58]. Group informatics was used to reflexively make meaning from the intersection of our field work and the significant amounts of digital trace data that emerges within open source projects; thus ensuring coherence of research constructs in the outcomes produced in our participant observation [2, 28, 29]. Finally, to ground our findings from participant observation and group informatics, we used direct engagement and critical reflection as our "process of learning from experience" [22, p. 56].

3.1 Data Gathering and Interpretation

Specific research methods employed included approximately 200 interviews, 200 survey responses, 10 focus groups, 1,000 pages of field notes, constant comparison, content analysis, trace analysis, social network analysis, and computational linguistic analysis. With respect to participant observation, we relied on interviews, surveys, focus groups, field notes, constant comparison, and content analysis. Our unit of analysis was most often the individual with abstraction to larger organizational settings to retain the fidelity of informant reports. With respect to group informatics, trace ethnographic methods were integrated with social network analysis, and computational linguistics to explore how engagement, as understood by informants, was represented in project practices. The unit of observation for these digital channels was most often the interaction contextualized by time, project, people, and organization. Collectively, these methods integrate field engagement, trace ethnography, and computational analysis [26] to scale our view of open source project to include corporate engagement. Through the use of participant observation and group informatics, members of our research team have produced numerous research papers aimed at specific aspects of organizational engagement with open source projects.

Our participant observation and group informatics enabled direct engagement and critical reflection to provide continual points of grounding to ensure that our experiences and research contributions were viable in practice. Further, this enabled us to practice reciprocity in order to understand the culture of the open source projects we sought to understand [64]. In this regard, we actively shared our work at open source conferences including the Community Leadership Summit (2018), CHAOSScon Europe (2018), CHAOSScon North America (2018), Open Source Leadership Summit (2018 and 2017), Open Source Summit North America (2018 and 2017), Open Source Summit Europe (2017), Mozilla Festival (2017), Open Source Collaboration Summit (2014 and 2013), Open Compliance Summit (2013), and LinuxCon (2012). Members of our research team are currently board members on a Linux Foundation project. We have performed site visits at leading open source organizations including Red Hat, Hewlett Packard, NVIDIA, Twitter, Microsoft, Yahoo, and IBM. Finally, we have served as active contributors and maintainers to open source projects including

[3] https://www.linuxfoundation.org/about/

Augur, CHAOSS, SPDX, Yocto, Bugmark, Drupal, OpenOffice.org, Toolkit for YNAB, and the Core Infrastructure Initiative Best Practices Badge.

Our direct engagement and critical reflection helped produce a collection of models and stories that we could share with others. The models and stories represented our understanding of open source compliance, open source health, or open source leadership — whatever our specific focus was at the time. The models and stories were presented, published, and discussed with open source project participants, producing real impact in organizational engagement with open source projects. They reflected our team's values in how we filtered experience, selected knowledge, and built interpretation [38, pp. 140-150]. From the models and stories, we were able to pursue one of our goals of getting as close to our setting as possible and then staying there [58, p. 20].

This process of moving from participant observation and group informatics to direct engagement and critical reflection was not always clean but it was consistent. Our research team met, on average, twice weekly over the last three years, notwithstanding open source project meetings. The aim of these meetings was to not only produce our models and stories for sharing with open source participants but to keep our narrative rolling at the risk of losing project continuity — "staying close to the sequential, immediate, and tightly linked flow of events as it is a function of the substance of the tale itself" [58, p. 103]. From this long term commitment and continuity, we present our findings to generate not just knowledge about open source projects but to foster a new acquaintance with such projects [58]. The findings are descriptions and related questions, temporarily stepping away from our more generalized theoretical work. While we recognize and support the balance between generating theory against building description, for this paper, we specifically leverage our methods not into theoretical problem solving but into descriptive problem setting [38].

4 FINDINGS AND OBSERVATIONS

Open source projects are, with fair reason, usually considered by researchers one project at a time. The classic example is the Linux kernel, a vast open source project, containing over 15 million lines of code, accepting contributions from some of the largest technology organizations in the world, and instantiated across a variety of distributions including Ubuntu, Slackware, and Red Hat.[4] However, open source ecosystems include not just a view of projects themselves, but a view that includes the organizations that rely on this corporate-communal relationship as critical part of work. The great acceleration of corporate engagement in open source projects, exemplified by the Linux Foundation's growth from a handful of projects in 2008 to 85 projects today is both amplifying and diversifying the dynamics in open source ecosystems.

> "There is no software company that does not have an open source strategy."[5] — Executive of an Open Source Foundation

Localized project considerations do not reveal the broader workings of the open source ecosystem. From a local perspective of most empirical work on open source projects, findings and local theories of open source projects produce specific knowledge with claims constrained to that context. When project specific studies are subsequently reframed as providing insight on a more general phenomenon of "open source software", the leap from specific to general could be an exemplar of challenges of generalizability considered by philosophers of science [46] and the useful frame of systems theory [4]. Consider, for example, the acceleration of our understanding of the universe over the past 100 years. The acceleration in knowledge of the universe mirrors the present

[4] A fantastic visual look of the timeline of distributions is hosted on Wikipedia: http://upload.wikimedia.org/wikipedia/commons/1/1b/Linux_Distribution_Timeline.svg

[5] Quotes are selected from the interviews we conducted with members of open source projects.

acceleration of knowledge of open source, with the result including specific ontologies we use to describe parts of the universe and its system.

Metaphorically, consider the Linux kernel as one star in a complex galaxy of open source stars, planets, and celestial bodies. Perhaps a few take the shape of very large stars, exemplified by their enormous mass and influence throughout adjacent parts of the galaxy. As the Linux kernel project moves, so do surrounding projects and organizations, and researchers can understand open source projects better by also understanding their position within the galaxy. By knowing the tilt and patterns of stars, we now infer planetary associations. By knowing how open source container orchestration projects (i.e., Kubernetes) align with open source operating system projects (i.e., Linux), we can understand how one project can receive and provide influence in the galaxy of open source projects and across the universe of social computing and collaborative work. For example, the Kubernetes-Linux relationship reveals that Kubernetes and the Linux kernel jointly influence each other. A closer look at Kubernetes, reveals that the project is part of the Cloud Native Computing Foundation (CNCF), founded in 2015, and another exemplar of the acceleration of corporate engagement with open source. Looking closer we observe that the CNCF is now home to 25 other open source projects that are not Kubernetes. Continuing the metaphor, Kubernetes is one star system within the CNCF region of the galaxy. A broader observation reveals that CNCF is actually a supported project at the Linux Foundation (to which the Linux kernel is also a project). Such a constellation of objects creates this massively complex and interconnected open source project galaxy. How does all of this fit and how does all of this work?

> "One company can't effectively do the work that ideally would be done by the whole open source project ecosystem." — Lawyer at a Large Software Company

Open source projects now constitute a position or node within an open source ecosystem with a complex array of partner open source projects, foundations, conservancies, and organizations. The open source ecosystem is so vast and ever-changing that to declare it understood is tantamount to declaring the galaxy fully researched after 20 years of peering through the Hubble Space Telescope. Such a complex ecosystem provides an endless set of questions and during our project, we continue to actively question if we, as researchers, can ever fully understand the complexities of open source projects and ecosystems. Yet to move this understanding forward, we present select observations and questions from corporate-communal engagements that frame what we know, and shape concise notions of what we do not yet know, with the general aim of informing our understanding of the complexities that define "a new normal" for open source projects.

4.1 Observation #1: Open Source Software Lives in Complex Supply Chains

Historically, open source engagement is viewed as an organizational-communal relationship, but open source project engagement can just as easily be found in organizational-organizational relationships. Software is like any component in an innovation supply chain. Organizations must make upstream and downstream procurement and distribution decisions. They can contract the product through external sources, develop internally, and leverage open source projects. Organizational-organizational relationships can be informal or formal. Informally, an organization is not necessarily engaging with an open source project, instead engaging with software that is comprised of both closed source and open source software components. Indeed, much of the supply chain includes open source software embedded in larger software products. Further, organizational-organizational relationships may be structured formally in the exchange of open source software. In these relationships, the upstream and downstream distribution of open source software (e.g., the Linux kernel) is a formal engagement. All partners work together, aware of each other, within the same supply chain. In these arrangements, organizations in the supply chain must be aware of the open

source software in an exchanged product, for reasons of technical dependency and software license awareness.

> "My company is a distributor for Linux and provides a toolkit to customers to build a custom Linux for embedded devices. We view ourselves as an extension of the engineering departments of our customers. We are in the middle of the supply chain." — Director of Open Source at a Medium-Sized Software Company

Any supply chain that includes computing equipment is likely to include software, and in this, open source software. The dynamics between that supply chain and the open source software may take a variety of forms. OpenSSL is an example of an informal software relationship between organizations. It is embedded and often enters an organization via its presence in other pieces of software (e.g., a distribution of the Apache Web Server). The same is true for open source software libraries, classes, and scripts — they enter an organization as part of a larger piece of software, often unbeknownst to that organization. The development of the Qualcomm DragonBoard (a product similar to the Raspberry Pi) is an example of a more formal organizational relationship. In this case, Qualcomm works with Linaro for upstream support in the Linux kernel for the DragonBoard platform. Qualcomm also works with Arrow Electronics for downstream distribution of the DragonBoard.

These complex supply chains have implications such as exposing organizations to new opportunities and risks; organizational engagement with open source projects is an accepted reality, and organizations, projects, and foundations play an important role in both realizing opportunity and mitigating risks. Corporate lawyers worry about open source licensing and risks associated with internal intellectual property as it comes into contact with open source. Managers and developers leverage new technologies that exist in relation to their upstream and downstream code bases. As open source software enters and leaves organizations in a variety of new ways, this new complexity requires attention in practice and research.

> "[Dependencies are] what you depend on and who depends on you, both ways. Your project is positioned in a graph of relationships with other projects. The difficulty here is that there is a lot of hidden stuff which takes place in proprietary software and which are nonetheless important. [Consider, for example, that] the firmware installed on every Intel CPU is MINIX, as we've learned from a Google researcher. And therefore MINIX is the most popular operating system in the world now. Windows runs on top of MINIX. MacOS runs on top of MINIX. And Linux runs on top of MINIX. And nobody knew it before." — CEO and Co-Founder of a Small Software and Service Company

Three Research Questions Regarding Open Source in Supply Chains:

- How are open source software irregularities, such as vulnerabilities and licenses, fixed in supply chains?
- How are organizations tracking their intake of open source software?
- What roles do open source foundations play in regularizing the open source software within a supply chain?

4.2 Observation #2: Open Source License Compliance is a Really Big Deal and It's Pretty Young

Open source software compliance is important for organizations engaged with open source projects. At its core, open source software is a licensing designation on top of a piece of copy-written code. In the case of open source software, without a designated license, no person has been given explicit rights to use that software. Yet, open source software carries licenses that define the rights and requirements associated with a piece of software. Commonly used open source software licenses

include the GPLv2, Apache2.0, MIT, and BSD.[6] Some of the more peculiar open source software licenses include the Death and Repudiation and the Beer-ware licenses:

> *D&R (Death and Repudiation) License*[7]
> This software may not be used directly by any living being. ANY use of this software (even perfectly legitimate and non-commercial uses) until after death is explicitly restricted. Any living being using (or attempting to use) this software will be punished to the fullest extent of the law.

> *THE BEER-WARE LICENSE (Revision 42)*[8]
> <phkFreeBSD.ORG> wrote this file. As long as you retain this notice you can do whatever you want with this stuff. If we meet some day, and you think this stuff is worth it, you can buy me a beer in return. Poul-Henning Kamp

As open source projects become an integral part of corporate innovation practices, licenses play a critical role in knowing how pieces of found open source software can be used and what obligations are necessary when using that software. Our knowledge about open source software licenses is well documented. What is unique is the standardization of practices associated with open source software license compliance to help normalize the processes by which licenses are identified, expressed, and shared [25].

> "Many companies, including many of our customers, are heavily investing in compliance, tooling, and processes and have several people working on it." — Head of Open Source Program Office at a Medium-Sized Open Source Software Company

To date, organizations often work on license identification and expression independently from one another, resulting in practice inconsistencies and greater collective uncertainty. In isolation, this is not a problem. However, as organizations share software through supply chains, tracing open source software and respective licenses is a challenge. Initiatives like the Linux Foundation's SPDX and OpenChain are aimed at reducing the inconsistencies by which organizations identify and express open source software licenses. Further, a service industry has developed around managing open compliance. Source code auditing organizations such as Black Duck, Source Auditor, and nexB provide license compliance-related consulting and support services for just such needs.

> "We sponsor the development of open source code tools, which are free and open source to deal with open source-relative issues and we sell a commercial subscription to a tool that is a dashboard for tracking the origin, provenance, licensing and other things that you need to track." — Co-founder and CEO of an Open Source Software Company

Three Research Questions Regarding Open Source License Compliance:

- How do supply chains accommodate sharing of open source license compliance information?
- What impact does an open source license compliance service industry have in the ongoing evolution of open source project engagement?
- What role do organizations like the Linux Foundation play in developing shared open source license compliance strategies?

4.3 Observation #3: Open Source Projects are Professional

Free software — software as a basic right, shared freely among people — is an idea with shifting influence in open source project engagements, partially as a byproduct of the growing complexity of open source supply chains. Christopher Kelty suggests open source projects are becoming domesticated [36]. Quite simply, open source projects, in the sense of Richard Stallman's free

[6] The Open Source Initiative maintains a list of OSI-approved open source software licenses: https://opensource.org/licenses
[7] Full license text: https://github.com/indeyets/syck/blob/master/COPYING
[8] https://people.freebsd.org/~phk/

software ideal, constitute a decreasing proportion of the open source world as corporate engagement accelerates, requiring clear and defined forms of engagement.

Professional open source projects are increasingly commercial endeavors, deliberately managed by many large, for-profit organizations that bring values that serve the supply chain more than social constructions of freedom to the forefront.[9] We do not deem this observation negative. It is simply what some open source projects have become. The way that London no longer fully represents the natural environments of the Thames River upon which it was established, open source projects no longer fully represent software as a basic right. It is simply a new norm. It is different. For better or worse, open source projects have evolved into a critical business endeavor.

> "Open source contribution has normalized and so it's not such a strange thing, isolated to these strange people who are really involved in X and really passionate about it. We have moved on from that and as it becomes more normalized you do lose a little bit of that history and the ethos and where that philosophy came from." — Marketing Manager at an Open Source Foundation

Within this new norm, structures that help open source projects support professional engagement emerge. These come from the organizations themselves in the formation of internal open source review boards. They come from the supporting foundations in the form of governance mechanisms that define many professional open source projects. In both, stability provides a recognizable and approachable space for organizations to engage with open source projects. Upstreaming key components of R&D to an unregulated open source project may be unappealing, but creating professional, or domesticated open source projects gives organizations (developers, managers, and lawyers) a recognizable point of entry when participating. In some cases, open source projects even require membership fees to be on the technical steering committee, which may only be feasible for large organizations interested in guiding the strategic direction of select open source projects.

> "I worked with [another project member] and some others and we tried to change the governance to make it more of a funded project but there was a lot of resistance to that so it didn't change. That was the only other thing we thought about in terms of [our open source project] that it should be more like some of the other collaboration projects. " — CEO and Co-Founder of a Small Software and Service Company

Three Research Questions Regarding Professional Open Source:

- How is open source project professionalism understood by organizations?
- How is work generated and distributed in professional open source project engagements?
- What mechanisms do organizations employ at the interface between private-public work?

4.4 Observation #4: In Professional Open Source Projects, the Distinction Between Employees & Volunteers is Diminishing

Open source project members have been historically considered volunteers, donating free time in the advancement of a communal good. However, with the influx of corporate members engaging with open source projects, the distinction between corporate employee and community volunteer is altered. Community members are both — open source project members and corporate employees — at the same time. Often, there are no volunteer community members without corporate affiliation. This reshapes prior experiences in open source that create a more singularly communal experience. The altering of roles is important for knowing that open source projects can, at times, carry little egalitarianism — a premise of early open source project research. Professional open source often continues to embrace volunteer contributors but is attuned to their corporate affiliation.

[9] Mozilla described prevalent open source project archetypes — some specifically designed to benefit individual organizations: https://blog.mozilla.org/wp-content/uploads/2018/05/MZOTS_OS_Archetypes_report_ext_scr.pdf

> "We try to determine the rough diversity of employment in our projects. The rough metric we have is about two-thirds of our contributors do not work for our primary corporate sponsor. That is extraordinarily critical when you look at both the objectives of our primary sponsor as well as the health of the community too." — Lead Program Manager for a Large Open Source Software Company

Open source project members often work for companies like AT&T, Cisco, Fujitsu, Google, Hitachi, Huawei, IBM, Intel, Microsoft, NEC, Oracle, Qualcomm, Samsung, Tencent,and VMWare (all Linux Foundation Platinum Members[10]). Employees of these and similar for-profit companies comprise open source project boards, represent leading contributors within the open source projects, and serve as stewards for open source projects at international conferences. At times, there can be only a small group of unaffiliated contributors who help constitute the open source project and their free agent status may limit their influence.

As open source project engagement becomes increasingly domesticated and stable — it becomes more closely aligned with corporate innovation. While there may be free agents within open source projects, they exist on a considerably more limited scale when open source project engagement is a means for getting corporate work done. This does not mean that these engagements are not open source project engagements, but it does mean that the membership of these open source projects is changing. In short, the licensing of the project remains open, but the member practices around the work on these projects are materially distinct from earlier characterizations of open source project research.

> "On projects that I worked on previously, there was always someone who was being paid to work on those projects. I don't think I've ever worked on an open source project where there wasn't people or a small group of people who were getting paid." — Co-Founder of a Startup

Three Research Questions Regarding Employee and Volunteer Roles:

- How can volunteers benefit from professional open source?
- How do people manage joint roles of being corporate employees and community members?
- What are new forms of volunteer work that are needed within an open source project comprised primarily of corporate members?

4.5 Observation #5: Open Source Software Procurement Decisions Require Trained Developers

To satisfy feature requirements, developers may substitute open source projects and code for writing the code themselves (e.g., a library for encryption which fulfills part of a product requirement). This reuse of software is a well-known aim of software engineering practice and a welcome change that boosts solution options and velocity for developers working in software supply chains.

The day-to-day effect of this is that developers now have to maintain some degree of awareness of software procurement values and risks — the process of finding external resources and using them internally. Procurement of software involves considerations such as software licenses, vulnerability reports, and open source project health. In a proprietary software setting, a developer would coordinate with a corporate procurement department for securing software in their work. In an open source software setting, the developer has autonomy and authority to make the procurement decision and thereby subsumes that function for the organization on a day-to-day basis.

> "We don't have a single overarching system, but we have very well trained developers who are empowered to make decisions about what software they use in their projects." — Senior Commercial Counsel at a Large Software Company

[10]Full list of Linux Foundation corporate members: https://www.linuxfoundation.org/membership/members/

The implication is that procurement decisions are moving to the edge of the organization. As such, employees have to be trained on how to make procurement decisions to limit the open source risks introduced to the organization. Open source program offices have formed within organizations to assist with the training necessary to educate developers, manage the many points of entry for open source software risk, and standardize how open source software enters a corporate innovation process.

One reaction could be to implement processes by which developers report all open source software procurement decisions. Adding this overhead certainly satisfies a desire to control and to be in-the-know. However, in an organization heavily involved in open source projects, the value of such a process is a net-negative. Nearly all organizations, whether they know it or not, engage in the procurement of open source software (e.g., by developers using Stack Overflow and GitHub). The recognition of this new reality of procurement is working its way through organizations and represents a change that is both welcome and uncertain.

> "I'm running our working group on open source readiness, which is producing a reference open source policy for [companies in a specific industry. It will describe] at a high level what a firm's policy toward open source use and contribution and engagement with the community is. Every large company that engages significantly in open source has a policy like this." — Lawyer at an Open Source Foundation

Three Research Questions Regarding Open Source Software Procurement:

- How is an organizational strategic focus maintained when procurement decisions are decentralized?
- How can organizations manage risk associated with decentralized procurement decisions?
- How can developer work be understood and rewarded when effectively engaging with open source projects?

4.6 Observation #6: The Health of Open Source Projects is a Hot-Topic

The health of open source projects is important in facilitating the cooperation between contributors, foundations, projects, and organizations within an open source ecosystem. Open source project health is important to all stakeholders because it signals that a project will continue to produce quality software. Organizations and individuals engaged with an open source project often share the same concerns, albeit for different reasons. Both want to engage in stable and successful projects where their contributions are valued and acknowledged. Measuring project health can help organizations and individual developers determine where to best focus their resources to maximize their desired outcomes.

> "[People] are looking at [health metrics] with different perspectives, with different reasoning. Foundations try to maintain a community and ensure that the community will be a long-living one. They look at all the progress and everything that's happening from the perspective of the ecosystem, and how to make the ecosystem healthy, and how to make the environment itself healthy, and a pleasant experience for [participants]. As a developer, I'm looking at it from a different point of view because, while I like a good environment and a nice and healthy environment around me, obviously, I have my personal goals to fulfill. If I'm contributing on behalf of a company, then I have goals that my employers set for me or we set together. I might also get a question from the employer about my view on a community, whether or not I think it's doing better than some other communities, and whether or not they should invest in it." — Community Manager at an Open Source Foundation

Open source project health can be informed by data from many different sources. Companies such as Red Hat and Bitergia have developed software analytics tools to measure metrics. They gather data from many systems supporting open source development (e.g., git, GitHub, Jira, Bugzilla, Gerrit, Jenkins, Slack, Discourse, Confluence, StackOverflow, and mailing lists) to provide analytics

and data visualization. Research efforts to analyze such data and correlate it with open source project success [e.g., 6, 52] have been openly discussed by practitioners as open source project health is a complex and evolving topic. For example, measurement of open source project health is confounding because health metrics may have different meanings for different projects and there is little consensus about the correct way to calculate metrics across projects. Despite the universal recognition that metrics for project health are helpful, there are also concerns about misinterpretation, abuse, or misuse of metrics. In response to this complexity, the Linux Foundation's Community Health Analytics Open Source Software (CHAOSS) project was created in 2017 to define both trace-data and socially-defined metrics, create tooling, advance methods, and develop standards for expressing open source project health.

> "I know about tool sets like Elasticsearch, GrimoireLabs, also companies like Bitergia. Our own company had [name of tool], that was built as a metrics tool. My interest in metrics really started when I came on board here, so I've been hearing about tools left and right. I will say that right now CHAOSS is probably the most focused project that I've seen on metrics itself, and the why behind the metrics." — Community Analyst at a Large Software Company

Three Research Questions Regarding Open Source Project Health:

- What actions can be informed by metrics to improve open source project health?
- How can undesirable outcomes from the use of metrics be prevented?
- How can open source project health inform the health of an open source ecosystem?

4.7 Observation #7: Sustainable Funding for Open Source Projects is Needed but Difficult to Establish

Despite the professionalism of open source projects, they are still built from the traditions and ideals of the free software movement. As a result, some projects find themselves stuck between professional and egalitarian ideals - in part corporate innovation and in part free software. This has contributed to very public coordination and maintenance failures such as Heartbleed [14] and the Struts/Equifax [49] data breach. As free software projects become strategically important, measures are being taken to ensure that these critically important projects are maintained and supported in the long-term.

In the past, supporting open source projects was viewed as charity; today, organizations may see open source project support as a necessity, just like paying electricity bills is important to keep the lights on. The Civil Infrastructure Platform, Core Infrastructure Initiative, and TideLift address the maintenance and funding issue by identifying central open source projects and their maintainers to provide funding for their work from member contributions. Several funding mechanisms to coordinate tasks across and between projects exist, including donations (e.g., non-profit foundations), bug bounty programs (e.g., Hackerone), open source bounty programs (e.g., Bountysource), crowd sourcing contests (e.g., Topcoder), and crowd funding (e.g., Open Collective).

> "When I first joined [company], one of the developers said, 'Hey, we need to open an Open Collective account and start giving to [project].' I'm like, 'Oh, great, cool. Why? Just out of curiosity.' He's like, 'Because then we can vote for features that we need,' so there's the value." — Director of Developer Relations at a Small Commercial Printing Company

However, providing sustainable funding for developers who maintain critically important open source projects remains an issue. The long-term maintainers of critical and widely used open source projects, especially when they are not employed by an organization, face financial challenges to sustain their work. Yet, there is a problem of identifying central projects and allocating appropriate funds one-by-one because of the reality that open source ecosystems have become so complex. Existing funding mechanisms may require project maintainers to use skills (e.g., marketing a crowd

funding campaign) to secure sufficient funding that are very different from what is required to develop quality software. Sometimes, funding mechanisms introduce incentives (e.g., keeping a bug fix concealed until a bug bounty payout is awarded) that run counter to the ideals of the open source software development model (e.g., share partial solutions and invite early feedback to negotiate the best solution with the community). Professional open source projects continue to need new and innovative market mechanisms for coordinating development and financing maintenance across projects while maintaining the loose coupling that has allowed the open source ecosystem to grow into what it is now. Such market mechanisms [e.g., 47] could channel some of the $100 billion that companies spend yearly on cybersecurity [44] and help lower the risk of major vulnerabilities and improve the sustainability of critical open source projects.

> "It concerns me when I see a struggling project that a lot of people use and that adds value to multiple companies. There are some really, really amazing sustainers out there like [maintainer name]. The things that he has done for [his project] is unbelievable. The money he has raised, the awareness that he has. He has given value to [company] and many other big sponsors. If every popular platform had [such a maintainer], there would be no issues in the open source community around sustainability issues, because he knows how to really bring value to his project sponsors." — Director of Developer Relations at a Small Commercial Printing Company

Three Research Questions Regarding Open Source Market Mechanisms:

- What are the effective market mechanisms for the sustainability of open source projects?
- What incentives and failures are introduced with new market mechanisms?
- How will stakeholders in open source projects respond to new market mechanisms?

4.8 Observation #8: Open Source Projects are Political

We see that open source projects as a political movement are changing. As corporate interests have become heavily involved and are actively shaping the development of open source software, an ongoing tension between interests of organizations and interests of the open source movement becomes manifest. It could be that the political open source movement loses its focus because much of our core open source development is now funded by large organizations [36], but we observe some evidence that the political nature of open source lives on.

Developers and organizations make a difference when they decide to work on open source projects such as healthcare projects (e.g., GNUHealth) that allow people in resource low environments to be better served, decide to work on packet sniffers (e.g., Wireshark) that can be used to build a firewall, or decide to work on container projects (e.g., Kubernetes) that orchestrate container management at scale. Each open source contribution becomes a vote in that project. The political power of developers in open source projects extends beyond their immediate choice of work. Developers who work in open source projects control what functionality their open source software provides and thereby influence what software and devices can do, how they are used in other projects, and what can be built from them.

> "You do have the ability to influence where a project goes, and this goes right back into community engagement. If you're engaging with [the community], if you know them, if you're interacting, if you're contributing, you're going to have a lot more ability to influence the direction of that project than somebody who's just sitting back and taking the code and being pretty quiet." — Developer at a Large Hardware Company

At times, organizations limit the freedoms postulated by open source. In 2017, for example, Facebook included a patent clause in the license of its open source React software. The developer community pushed back and Facebook reverted to the permissive MIT open source license.[11] In

[11] https://qz.com/1087865/outraged-programmers-stood-up-to-facebook-fb-over-open-source-licensing-and-won-sort-of/

this case, proponents of the open source movement pushed back to maintain some sense of the roots of open source software. Further, when a company develops a proprietary extension to an open source software, we often see that the open source community subsequently creates an open source alternative, again casting a vote for the future of a code base.

> "Somebody within the community becomes sufficiently motivated to do the work that they felt should have been done [open source, not proprietary] in the first place, and then produce an open source plug-in which competes with the proprietary one. In the end, the proprietary one loses because it always does. In every scenario." — Open Source Officer at a Large Software Company

Three Research Questions Regarding Open Source as Political:

- How does professional open source influence votes cast in the future of open source projects?
- How are politics manifest in professional open source projects?
- How are political views of open source developers and their employers influenced by corporate goals?

5 DISCUSSION AND CONCLUSION

Open source projects are an ever-evolving, complex array of phenomena. Corporate-communal open source ecosystems are one exemplar of social computing and work that, in present form, challenge existing models of collaboration. Through our engaged field work, we observed an opportunity for expanding existing open source research with the new reality. The work of previous researchers provides a necessary but insufficient framework for understanding today's open source project engagement and open source ecosystems. Engaged field work and narrative characterizations of contemporary open source project stories are needed to help adapt the trajectory of CSCW and social computing research. We affirm that open source project research is not new, but we think we bring new light to bear on its continued emergence.

To make the scope of our work tractable for other researchers and open source ecosystem stakeholders, we chose to shape our findings as eight observations that are surprising and novel when cast against prior examinations of open source projects. Further, we share what we are curious about and our thoughts on where research on open source projects, software, and ecosystems might evolve to through our 24 research questions. Both — our observations and questions — are aimed at CSCW researchers to underscore the evolving nature of this form of open collaboration, and to illustrate the opportunities and unique potential of open source project research. We have merely provided a primer, many more research questions exist. Through our observations and research questions we show that CSCW researchers can contribute to all categories of open source project research (Table 2), framing a set of implications for open source research and practice specifically, and social computing more generally. While implications can be drawn across a number of theoretical domains in the ongoing examination of open source projects, design and privacy/ethics are presented as domains to which the observations and questions can provide insight.

5.1 Implications for Design Research in Social Computing

Community and network focused lenses for understanding open collaboration provide an important theoretical foundation for design research. Given the growing significance of corporate engagement in open source projects, organization science and small group theories are increasingly relevant components for making sense of and explaining observed phenomena. There are new sociotechnical, organizational, and political realities around open source projects that, like the studies of freedom of earlier open source project research, likely foretell a new future of design research in social computing.

Category	Observation
Conceptual	*#1:* Open Source Lives in Complex Supply Chains
Performance Metrics	*#6:* The Health of Open Source Projects is a Hot-Topic
Legal and Regulatory	*#2:* Open Source Compliance is a Really Big Deal and It's Pretty Young
Production	*#3:* Open Source Projects are Professional *#4:* In Professional Open Source Projects, the Distinction Between Employees & Volunteers is Diminishing *#7:* Sustainable Funding for Open Source Projects is Needed but Difficult to Establish
Diffusion	*#5:* Open Source Software Procurement Decisions Require Trained Developers
Beyond Software	*#8:* Open Source Projects are Political

Table 2. Matching our observations to the open source research categories by Aksulu and Wade [1].

Design research that pays close attention to the experience of people directly engaged in open source projects is essential for making sense of an ecosystem that is evolving significantly and is likely to impact the values, ethics, and practices of substantial parts of the social computing research ecosystem. Many of the organizations that facilitate social computing in practice also contribute to design within open source projects. These inward facing efforts will influence the experience of open collaboration and social computing platform users over time. This influence will, in all likelihood, grow and reshape the design of technologies that underly much of the human experience online.

Design in open source projects is a dynamic, shared, and responsive activity. It often eschews linear design methodologies and frameworks in favor of an approach where all members, from developers to users, are active participants in the process. As design concepts are witnessed and understood, open source projects provide a unique and accessible view into the social and technical parts of the design process. Further, much open source software exists as infrastructure upon which more extensive platforms are built, and implications for design can emerge subtly, and sometimes shrewdly over time. As such, design research can not only be used to revisit the premises of open source software but also ask how design is manifest within the interoperability and complexity that now exists within open source software supply chains.

5.2 Implications for Privacy and Ethics Research in Social Computing

Corporations invest in open source projects that do not differentiate them from their marketplace competitors and include technologies essential for basic software operations. Investments in the Linux kernel, for example, are focused on the operating system for physical computers and how those physical computers are optimally utilized. Extending that example, Kubernetes makes it possible to support isolated virtual machines "on demand", instead of requiring a corporation to maintain an active set of systems constantly available to handle the largest speculative user load. This corporation spanning shared interest in open source is likely to extend into the domain of privacy and ethics in similar ways.

In a hopeful light, as demands for more transparent disclosures of user data privacy and use grow from the public, few corporations are likely to view those demands as distinct for themselves in reference to the open source software marketplace. For example, developing software features to comply with policies such as the European General Data Protection Regulation (GDPR) is a requirement all companies face. The development of solutions in open source projects for complying with regulatory demands has the added benefit that compliance can be demonstrated through code

audits. Companies reduce their risk by relying on audited and trusted open source software. At the same time, the open source software development facilitates public discussion and learning that is available not only to companies but to research as well.

As we discuss privacy and ethics considerations for user data in the social computing research community, it is advisable to maintain awareness of what is happening in open source projects with privacy and ethics touch points and to seek opportunities to influence the direction. Open source projects, originating from a value system founded on openness (a kind of transparency) are well positioned to play a role in assuring people that the corporate community is attending to privacy and ethics concerns responsibly. The principle corporate aim for engagement in open source is resource sharing and leverage, which has the effect of shaping culture across projects in ways not previously witnessed in open source. As such, there is opportunity to engage the open source community in discussions and consideration of privacy and ethics concerns across a range of social computing topics. And, through this engagement, the potential to amplify the effects of work on privacy and ethics in data centric fields especially is unprecedented. Privacy and ethics will likely be challenged as the "open" part of open source remains as a leverage point for organizations influencing the evolution of the technology and work upon which the "computing" part of social computing rests.

5.3 Conclusions

Practitioners often acknowledge aspects of the new realities in open source projects but few examine their full scope of consequence. Ideals from the free software movement persist and continue to guide individual engagement in open source projects, which is noble but can be out of step with the emerging open source project reality. Project maintainers must understand the significance of the supply chains their projects become part of to avoid exposing not only immediate users but national infrastructure projects to failures that can stem from interconnectedness. Organizations seeking to balance the use of open source software in their internal innovation streams must adjust internal practices to account for open source project health risks and how its engagement impacts a larger software ecosystem.

Open source projects and their vast ecosystems, as a whole, are facing a collective action challenge born of the tensions between its past and the emerging future. The new realities of open source projects are shifting who benefits from the wealth generated in these projects. We observed many new realities (e.g., changes of membership from volunteers to paid employees to a lack of effective market mechanisms). These new realities are interrelated and cannot be solved by a single open source project or organization, but provide new forms of engagement in the design and distribution of software. New mechanisms of practice and research can be developed to incentivize the development of healthy projects that produce secure software, incentivize the engagement of volunteers who are not affiliated with an organization, and continue to leverage the traditional and successful open source development model. This paper is intended to serve as a starting point for these new conversations.

ACKNOWLEDGMENTS

This project received funding from the National Science Foundation's Virtual Organizations as Sociotechnical Systems and the Innovation and Organizational Sciences Programs under Grant Num: 1122642 (https://www.nsf.gov/awardsearch/showAward?AWD_ID=1122642).

This project received funding from the Alfred P. Sloan Foundation Digital Technology grant on Open Source Health and Sustainability, Num: 8434 (https://sloan.org/grant-detail/8434).

This project received funding from Mozilla.

ORCID IDs

- Matt Germonprez https://orcid.org/0000-0003-2326-5901
- Georg J.P. Link https://orcid.org/0000-0001-6769-7867
- Kevin Lumbard https://orcid.org/0000-0001-9306-3040
- Sean P. Goggins https://orcid.org/0000-0002-4331-147X

REFERENCES

[1] Altay Aksulu and Michael Wade. 2010. A comprehensive review and synthesis of open source research. *Journal of the Association for Information Systems* 11, 11 (2010).

[2] Kelly Blincoe, Jyoti Sheoran, Sean Goggins, Eva Petakovic, and Daniela Damian. 2016. Understanding the popular users: following, affiliation influence and leadership on GitHub. *Information and Software Technology* 70 (Feb. 2016), 30–39. https://doi.org/10.1016/j.infsof.2015.10.002

[3] Kelly Blincoe, Giuseppe Valetto, and Sean Goggins. 2012. Proximity: a measure to quantify the need for developers' coordination. In *Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work (CSCW '12)*. ACM, New York, NY, USA, 1351–1360. https://doi.org/10.1145/2145204.2145406

[4] Kenneth E. Boulding. 1956. General systems theory-the skeleton of science. *Management Science* 2, 3 (1956), 197–208. http://www.jstor.org/stable/2627132

[5] Henry Chesbrough. 2006. Open innovation: a new paradigm for understanding industrial innovation. In *Open Innovation: Researching a New Paradigm*, Henry Chesbrough, Wim Vanhaverbeke, and Joel West (Eds.). Oxford University Press, Oxford, 1–12.

[6] Kevin Crowston, James Howison, and Hala Annabi. 2006. Information systems success in free and open source software development: theory and measures. *Software Process: Improvement and Practice* 11, 2 (March 2006), 123–148. https://doi.org/10.1002/spip.259

[7] Kevin Crowston, Qing Li, Kangning Wei, U. Yeliz Eseryel, and James Howison. 2007. Self-organization of teams for free/libre open source software development. *Information and Software Technology* 49, 6 (2007), 564 – 575. https://doi.org/10.1016/j.infsof.2007.02.004

[8] Kevin Crowston, Kangning Wei, James Howison, and Andrea Wiggins. 2012. Free/libre open-source software development: what we know and what we do not know. *Comput. Surveys* 44, 2 (Feb. 2012), 7:1–7:35. https://doi.org/10.1145/2089125.2089127

[9] Laura Dabbish, Colleen Stuart, Jason Tsay, and Jim Herbsleb. 2012. Social coding in GitHub: transparency and collaboration in an open software repository. In *Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work (CSCW '12)*. ACM, New York, NY, USA, 1277–1286. https://doi.org/10.1145/2145204.2145396

[10] Linus Dahlander. 2007. Penguin in a new suit: a tale of how de novo entrants emerged to harness free and open source software communities. *Industrial and Corporate Change* 16, 5 (May 2007), 913–943. https://doi.org/10.1093/icc/dtm026

[11] Linus Dahlander, Lars Frederiksen, and Francesco Rullani. 2008. Online communities and open innovation. *Industry and Innovation* 15, 2 (April 2008), 115–123. https://doi.org/10.1080/13662710801970076

[12] Linus Dahlander and Mats G. Magnusson. 2005. Relationships between open source software companies and communities: observations from Nordic firms. *Research Policy* 34, 4 (May 2005), 481–493. https://doi.org/10.1016/j.respol.2005.02.003

[13] Paul Dourish. 2001. *Where the action is: the foundations of embodied interaction.* MIT Press, Cambridge, Mass.

[14] Zakir Durumeric, James Kasten, David Adrian, J. Alex Halderman, Michael Bailey, Frank Li, Nicolas Weaver, Johanna Amann, Jethro Beekman, Mathias Payer, and Vern Paxson. 2014. The matter of Heartbleed. In *Proceedings of the 2014 Conference on Internet Measurement Conference (IMC '14)*. ACM, Vancouver, BC, Canada, 475–488. https://doi.org/10.1145/2663716.2663755

[15] Margaret S. Elliott and Walt Scacchi. 2008. Mobilization of software developers: the free software movement. *Information Technology & People* 21, 1 (2008), 4–33.

[16] Joseph Feller, Patrick Finnegan, Brian Fitzgerald, and Jeremy Hayes. 2008. From peer production to productization: a study of socially enabled business exchanges in open source service networks. *Information Systems Research* 19, 4 (2008), 475 – 493. https://doi.org/10.1287/isre.1080.0207

[17] Casey Fiesler, Shannon Morrison, R. Benjamin Shapiro, and Amy S. Bruckman. 2017. Growing their own: legitimate peripheral participation for computational learning in an online fandom community. In *Proceedings of the 2017 ACM*

Conference on Computer Supported Cooperative Work and Social Computing (CSCW '17). ACM, New York, NY, USA, 1375–1386. https://doi.org/10.1145/2998181.2998210

[18] Anna Filippova and Hichang Cho. 2015. Mudslinging and manners: unpacking conflict in free and open source software. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing (CSCW '15)*. ACM, New York, NY, USA, 1393–1403. https://doi.org/10.1145/2675133.2675254

[19] Anna Filippova and Hichang Cho. 2016. The effects and antecedents of conflict in free and open source software development. In *Proceedings of the 19th ACM Conference on Computer-Supported Cooperative Work & Social Computing (CSCW '16)*. ACM, New York, NY, USA, 705–716. https://doi.org/10.1145/2818048.2820018

[20] Stephen M. Fiore. 2008. Interdisciplinarity as teamwork: how the science of teams can inform team science. *Small Group Research* 39, 3 (June 2008), 251–277. https://doi.org/10.1177/1046496408317797

[21] Brian Fitzgerald. 2006. The transformation of open source software. *MIS Quarterly* 30, 3 (2006), 587 – 598. https://doi.org/10.2307/25148740

[22] Jan Fook. 2011. Developing critical reflection as a research method. In *Creative Spaces for Qualitative Researching*, Joy Higgs, Angie Titchen, Debbie Horsfall, and Donna Bridges (Eds.). Sense Publishers, Rotterdam, 55–64. https://doi.org/10.1007/978-94-6091-761-5_6

[23] Pål Fugelli, Leif C. Lahn, and Anders I. Mørch. 2013. Shared prolepsis and intersubjectivity in open source development: expansive grounding in distributed work. In *Proceedings of the 2013 Conference on Computer Supported Cooperative Work (CSCW '13)*. ACM, New York, NY, USA, 129–144. https://doi.org/10.1145/2441776.2441793

[24] Jonas Gamalielsson and Björn Lundell. 2017. On licensing and other conditions for contributing to widely used open source projects: an exploratory analysis. In *Proceedings of the 13th International Symposium on Open Collaboration (OpenSym '17)*. ACM, New York, NY, USA, 9:1–9:14. https://doi.org/10.1145/3125433.3125456

[25] Robin Gandhi, Matt Germonprez, and Georg J.P. Link. 2018. Open data standards for open source software risk management routines: an examination of SPDX. In *Proceedings of ACM GROUP '18*. ACM, Sanibel Island, Florida, USA, 219–229. https://doi.org/10.1145/3148330.3148333

[26] R. Stuart Geiger and David Ribes. 2011. Trace ethnography: following coordination through documentary practices. In *Proceedings of the 44th Hawaii International Conference on System Sciences*. https://doi.org/10.1109/HICSS.2011.455

[27] Matt Germonprez, Julie E. Kendall, Kenneth E. Kendall, Lars Mathiassen, Brett Young, and Brian Warner. 2016. A theory of responsive design: a field study of corporate engagement with open source communities. *Information Systems Research* 28, 1 (Nov. 2016), 64–83. https://doi.org/10.1287/isre.2016.0662

[28] Sean Goggins, Giuseppe Valetto, Christopher Mascaro, and Kelly Blincoe. 2013. Creating a model of the dynamics of socio-technical groups. *User Modeling & User-Adapted Interaction* 23, 4 (Sept. 2013), 345–379. https://doi.org/10.1007/s11257-012-9122-3

[29] Sean P. Goggins, Christopher Mascaro, and Giuseppe Valetto. 2013. Group informatics: a methodological approach and ontology for sociotechnical group research. *Journal of the American Society for Information Science and Technology* 64, 3 (March 2013), 516–539. https://doi.org/10.1002/asi.22802

[30] Carl Gutwin, Reagan Penner, and Kevin Schneider. 2004. Group awareness in distributed software development. In *Proceedings of the 2004 ACM Conference on Computer Supported Cooperative Work (CSCW '04)*. ACM, Chicago, Illinois, USA, 72–81. https://doi.org/10.1145/1031607.1031621

[31] Christine A. Halverson, Jason B. Ellis, Catalina Danis, and Wendy A. Kellogg. 2006. Designing task visualizations to support the coordination of work in software development. In *Proceedings of the 2006 20th Anniversary Conference on Computer Supported Cooperative Work (CSCW '06)*. ACM, New York, NY, USA, 39–48. https://doi.org/10.1145/1180875.1180883

[32] Mona Haraty, Joanna McGrenere, and Andrea Bunt. 2017. Online customization sharing ecosystems: components, roles, and motivations. In *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing (CSCW '17)*. ACM, New York, NY, USA, 2359–2371. https://doi.org/10.1145/2998181.2998289

[33] James Howison and Kevin Crowston. 2014. Collaboration through open superposition: a theory of the open source way. *MIS Quarterly* 38, 1 (March 2014), 29–50. https://misq.org/collaboration-through-open-superposition.html

[34] James Howison and James D. Herbsleb. 2013. Incentives and integration in scientific software production. In *Proceedings of the 2013 Conference on Computer Supported Cooperative Work (CSCW '13)*. ACM, New York, NY, USA, 459–470. https://doi.org/10.1145/2441776.2441828

[35] Christopher M. Kelty. 2008. *Two bits: The cultural significance of free software*. Duke University Press, Durham.

[36] Christopher M. Kelty. 2013. There is no free software. *Journal of Peer Production* 1, 3 (2013). http://peerproduction.net/issues/issue-3-free-software-epistemics/debate/there-is-no-free-software/

[37] Kurt Luther and Amy Bruckman. 2008. Leadership in online creative collaboration. In *Proceedings of the 2008 ACM Conference on Computer Supported Cooperative Work (CSCW '08)*. ACM, New York, NY, USA, 343–352. https://doi.org/10.1145/1460563.1460619

[38] Marg Lynn. 1996. Negotiating practice theory (or, I've got the theory, you've got the practice). In *The Reflective Researcher: Social Workers' Theories of Practice Research*, Jan Fook (Ed.). Allen & Unwin, St. Leonards, NSW, Australia, 140–150.

[39] Jennifer Marlow and Laura Dabbish. 2013. Activity traces and signals in software developer recruitment and hiring. In *Proceedings of the 2013 Conference on Computer Supported Cooperative Work (CSCW '13)*. ACM, New York, NY, USA, 145–156. https://doi.org/10.1145/2441776.2441794

[40] Jennifer Marlow and Laura A. Dabbish. 2015. The effects of visualizing activity history on attitudes and behaviors in a peer production context. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing (CSCW '15)*. ACM, New York, NY, USA, 757–764. https://doi.org/10.1145/2675133.2675250

[41] Nora McDonald and Sean Goggins. 2013. Performance and participation in open source software on GitHub. In *CHI '13 Extended Abstracts on Human Factors in Computing Systems (CHI EA '13)*. ACM, New York, NY, USA, 139–144. https://doi.org/10.1145/2468356.2468382

[42] Amanda McPherson, Brian Proffitt, and Ron Hale-Evans. 2008. Estimating the total cost of a linux distribution. https://www.linux.com/publications/estimating-total-cost-linux-distribution

[43] Tom Mens, Maálick Claes, Philippe Grosjean, and Alexander Serebrenik. 2014. Studying evolving software ecosystems based on ecological models. In *Evolving Software Systems*. Springer, Berlin, Heidelberg, 297–326. https://doi.org/10.1007/978-3-642-45398-4_10

[44] Steve Morgan. 2018. 2018 cybersecurity market report. https://cybersecurityventures.com/cybersecurity-market-report/

[45] Siobhán O'Mahony and Fabrizio Ferraro. 2007. The emergence of governance in an open source community. *Academy of Management Journal* 50, 5 (2007), 1079–1106. https://doi.org/10.5465/AMJ.2007.27169153

[46] Karl R. Popper. 1969. *Conjectures and refutations: the growth of scientific knowledge* (3rd ed. (revised) ed.). Routledge & K. Paul, London.

[47] Malvika Rao, Georg J.P. Link, Don Marti, Andy Leak, and Rich Bodo. 2018. A trading market to incentivize secure software. In *17th Annual Workshop on the Economics of Information Security (WEIS) Proceedings*. Innsbruck, Austria. https://weis2018.econinfosec.org/wp-content/uploads/sites/5/2016/09/WEIS_2018_paper_27.pdf

[48] D. Riehle, P. Riemer, C. Kolassa, and M. Schmidt. 2014. Paid vs. volunteer work in open source. In *System Sciences (HICSS), 2014 47th Hawaii International Conference on*. 3286–3295. https://doi.org/10.1109/HICSS.2014.407

[49] Sally. 2017. Apache Struts statement on Equifax security breach. https://blogs.apache.org/foundation/entry/apache-struts-statement-on-equifax

[50] Walt Scacchi. 2007. Free/open source software development: recent research results and emerging opportunities. In *Proceeding ESEC-FSE companion '07*. ACM Press, Dubrovnik, Croatia, 459–468. https://doi.org/10.1145/1295014.1295019

[51] Jodi Schneider, Krystian Samp, Alexandre Passant, and Stefan Decker. 2013. Arguments about deletion: how experience improves the acceptability of arguments in ad-hoc online task groups. In *Proceedings of the 2013 Conference on Computer Supported Cooperative Work (CSCW '13)*. ACM, New York, NY, USA, 1069–1080. https://doi.org/10.1145/2441776.2441897

[52] Charles M. Schweik and Robert C. English. 2012. *Internet success: A study of open-source software commons*. MIT Press, Cambridge, Mass.

[53] Vandana Singh and Michael B. Twidale. 2008. The confusion of crowds: non-dyadic help interactions. In *Proceedings of the 2008 ACM Conference on Computer Supported Cooperative Work (CSCW '08)*. ACM, New York, NY, USA, 699–702. https://doi.org/10.1145/1460563.1460670

[54] James P. Spradley. 1980. *Participant observation*. Holt, Rinehart and Winston, New York.

[55] Will Steffen, Wendy Broadgate, Lisa Deutsch, Owen Gaffney, and Cornelia Ludwig. 2015. The trajectory of the anthropocene: the great acceleration. *The Anthropocene Review* 2, 1 (April 2015), 81–98. https://doi.org/10.1177/2053019614564785

[56] Igor Steinmacher, Tayana Conte, Marco Aurélio Gerosa, and David Redmiles. 2015. Social barriers faced by newcomers placing their first contribution in open source software projects. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing (CSCW '15)*. ACM, New York, NY, USA, 1379–1392. https://doi.org/10.1145/2675133.2675215

[57] Chandrasekar Subramaniam, Ravi Sen, and Matthew L. Nelson. 2009. Determinants of open source software project success: a longitudinal study. *Decision Support Systems* 46, 2 (Jan. 2009), 576–585. https://doi.org/10.1016/j.dss.2008.10.005

[58] John van Maanen. 1988. *Tales of the field: on writing ethnography*. The University of Chicago Press, Chicago, IL.

[59] Bogdan Vasilescu, Alexander Serebrenik, Prem Devanbu, and Vladimir Filkov. 2014. How social Q&A sites are changing knowledge sharing in open source software communities. In *Proceedings of the 17th ACM Conference on Computer Supported Cooperative Work & Social Computing (CSCW '14)*. ACM, New York, NY, USA, 342–354. https://doi.org/10.1145/2531602.2531659

[60] Eric von Hippel and Georg von Krogh. 2003. Open source software and the 'private-collective' innovation model: issues for organization science. *Organization Science* 14, 2 (2003), 209 – 223. https://doi.org/10.1287/orsc.14.2.209.14992

[61] Georg von Krogh, Stefan Haefliger, Sebastian Spaeth, and Martin W. Wallin. 2012. Carrots and rainbows: motivation and social practice in open source software development. *MIS Quarterly* 36, 2 (June 2012), 649–676. http://misq.org/carrots-and-rainbows-motivation-and-social-practice-in-open-source-software-development.html

[62] Georg Von Krogh and Eric Von Hippel. 2006. The promise of research on open source software. *Management Science* 52, 7 (2006), 975–983. http://www.jstor.org/stable/20110574

[63] Yi Wang and David Redmiles. 2016. The diffusion of trust and cooperation in teams with individuals' variations on baseline trust. In *Proceedings of the 19th ACM Conference on Computer-Supported Cooperative Work & Social Computing (CSCW '16)*. ACM, New York, NY, USA, 303–318. https://doi.org/10.1145/2818048.2820064

[64] Harry F. Wolcott. 2005. *The art of fieldwork* (2nd ed ed.). Altamira Press, Walnut Creek, CA.

[65] Yutaka Yamauchi, Makoto Yokozawa, Takeshi Shinohara, and Toru Ishida. 2000. Collaboration with lean media: how open-source software succeeds. In *Proceedings of the 2000 ACM Conference on Computer Supported Cooperative Work (CSCW '00)*. ACM, New York, NY, USA, 329–338. https://doi.org/10.1145/358916.359004

[66] Roshanak Zilouchian Moghaddam, Zane Nicholson, and Brian P. Bailey. 2015. Procid: bridging consensus building theory with the practice of distributed design discussions. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing (CSCW '15)*. ACM, New York, NY, USA, 686–699. https://doi.org/10.1145/2675133.2675272

Received April 2018; revised July 2018; accepted September 2018