‘Patches Don’t Have Gender’: Epistemic Cultures of Open Software Production
Abstract

As technology changes, the work that gender does in and for technology production also changes. Gender discourse in open source programming reveals broader issues that tell us something beyond the question of why there are so few women. Participants’ claims that gender is irrelevant reveal deeper questions about knowledge production that become increasingly important as openness becomes a constitutive model of technology production. The inclusion of women is a threatening prospect because it questions the means by which software developers put themselves at technology’s ‘edge.’ By selectively relinquishing their agency to the technology itself, developers create the social and cultural resources with which to sever ties with other developers and institutions. Gender shows how the cutting of social relations is equally productive for new media knowledge production as collaboration and ‘open’ knowledge sharing.

Open source, gender, epistemic culture, knowledge economy
Introduction: Research Problem and Methods

Free/libre/open source software (F/LOSS) developers are a loosely-knit group of programmers who forego traditional intellectual property rights in favor of what they see as better collaboration, knowledge exchange, and ultimately improved software technology. Open source is a more common term in the press and in commercial circles, but “free” or “libre” are also important terms for community members, hence we use the inclusive form, F/LOSS. Linux is the most famous example of software built this way, but there are numerous other projects, usually smaller pieces of software rather than entire operating systems. F/LOSS is widely considered a significant transformation in cultures of technology production (Kelty, 2004, Weber 2004, Lessig 1999), and is taken seriously by some of the most influential quarters of industry and the public sector.

In 2005 [omitted for purposes of blind review] a team of three anthropologists conducted an ethnographic study to examine why it was that F/LOSS had such a poor record of gender-based inclusion. An earlier study showed that just 1.5% of F/LOSS participants were women, compared with 28% of women in the proprietary software industry (Ghosh et al 2002). Through the course of this research, and the subsequent fallout from the [omitted] report, there was no evidence that women’s participation levels, or the fundamental issues behind them, had changed. What had changed was the growth in large commercial enterprises that contributed to F/LOSS, but this did not seem to budge women’s exclusion, despite more women working in these organizations. Our central research problem was twofold. First, what is it about F/LOSS that made it so rife with gender-based exclusion? Second, to the extent that F/LOSS is an influential shift in how technology is produced more widely, do the underlying issues that generate
this exclusion tell us something more fundamental about how technology production is changing? That is, what is it that gender reveals, beyond gender \textit{per se}?

We conducted participant observation intermittently across two years in the UK, France, and Germany across multiple software projects. Some are large, such as Linux, and some smaller. This entailed becoming involved in groups’ day-to-day affairs, participating in informal meetups and formal conferences like OSCON, observing and participating in online communication, and learning to code. We used a combination of semi-structured and open-ended interviews alongside participant observation. We spoke with individual contributors and those that occupy organizational and community building roles. As this was primarily ethnographic research there was no single interview protocol, but largely we were looking for evidence of perceptions of gender, social organization, forms of formal and informal power, how people understood code to work, and their motivations for contributing. In addition, we conducted an online survey of 1500 F/LOSS contributors, although for the purposes of this paper we focus on the ethnographic findings.

\textbf{Why Study Gender?}

There is an extensive gender and technology literature that provides convincing reasons for why women are less likely to be found in communities of technology production, such as this one. For example, Lin (2005), following Wajcman (2000) discovered that in the F/LOSS communities she examined, the way programming environments were designed assumed a history with obsolete technologies which new entrants to the field did not have. Women tended to be newer to programming, and thus were subtly but powerfully excluded through design choices. Wajcman’s work, alongside other like Sorensen (2002), Lie (2003), Gill and Grint (1995), Green and Adam
(2001), show women’s exclusion to be a broader problem of socio-technical construction, where both the material aspects of computing as well as the social identities that people create for themselves through engaging with programming are cultures that are made by and for men. This literature shows how programming cultures are about creating and sustaining certain forms of masculinity, which comes at the price of women’s exclusion.

While culture is not easy to change, Margolis and Fisher (2002) showed that the absence of women in technology is not nearly as unchangeable as one might assume. Similarly, Corneliussen (2009) has identified the false impression of stability in the gender and technology literature: if women are ‘still’ not in ICTs in equal numbers, this is falsely taken as indicative of the same, unchanging set of issues, and obscures the transformations that gender has a good deal to tell us about. Like Corneliussen, this work takes the view that as technology changes, the work that gender does in and for technology production also changes.

In the course of fieldwork, we found that how people talked about and thought about gender did include all the problems of identity and socio-technical construction that the gender and technology literature has robustly explored. But it also took a form which revealed broader issues at stake in open models of technology production that are worth exploring beyond the question of why there are so few women. The central argument is that participants find it difficult to talk about, let alone address, women’s exclusion because to do so questions the means by which one gains both status and personal satisfaction with technical work. There is a precarious relationship between a radical positivism on one hand and the social strategies on the other that enable people to constitute themselves as boundary pushers. The paper will conclude by suggesting that this opens up grounds for interrogating the ways that F/LOSS, and perhaps too the wider forms of open innovation (Chesbrough 2003) of which F/LOSS is but one instantiation,
does not encourage unconstrained open participation as the movement claims, but may in fact also make certain kinds of relationships less possible. That is, this particular form of exclusion may be telling us something about wider turns new media appears to be taking.

Gender exclusion takes an extreme form both qualitatively and quantitatively. As mentioned earlier, mere 1.5% are women, whereas there are far more in proprietary software and hardware firms. Extremes can also be found in the harsh treatment that women face, which include widespread sexualization, and women finding it necessary to remind men not to stare and point at them. It also can be seen in the willful inattention to offensive talk. From a female participant:

“One IRC channel I used frequently made jokes about rapes… At one stage he ended up telling me, if I couldn’t “stand the heat I should get out of the kitchen” and then I said, “What would you say… if I had been raped and I took exception to be used as a subject of humor?” and he said: “that’s too bad, but you need to learn to live with it.”

Even the merest mention of gender touched an apparently raw nerve. For example, there is no surer way to begin a flame war, that is, a series of inflammatory messages posted on an Internet discussion board, than to raise the issue of women’s inclusion. Word had spread quickly that our research was taking place, sparking irate tirades and occasional outright vitriol about how we were the sexist ones by even studying gender. Once we published our [omitted for purposes of blind review] report, we became known as authors of the “infamous” [name omitted] report—a description that featured in our introduction onto stage at a 2500-person F/LOSS conference. Years later, the report was still used by advocates within the community as sorely needed proof
that a gender bias exists. This was puzzling: why was proof that women were being excluded even necessary with such thorough male dominance?

The puzzle deepened still through the sincerity with which F/LOSS participants committed themselves to openness. In daily F/LOSS activities developers are constantly reminded about how increasing the number and diversity of heads around a problem produces better software. Participants say they reveal the source code of software to others to open up problems to scrutiny by a wide group, and thus to produce better software in the end. The whole point is to traverse boundaries, not rigidly maintain them. Inclusion of any and all interested parties is a central, constitutive part of F/LOSS social organization, which would suggest more opportunities for women to participate, not less.

The strength of the gender skew becomes even more puzzling when we take into account the relative newness of F/LOSS. Kotamraju (2004) documents how, as in the earliest days of computing, it was only with increased specialization and job insecurity that web development became male dominated. At the time of study, F/LOSS was very much in a similar position as early web design, yet even at this relatively early stage, F/LOSS excluded women. In sum, women’s exclusion becomes puzzling when we consider the timing of F/LOSS’s evolution, the extreme form the exclusion takes, and, perhaps most intriguingly, the genuinely held values surround inclusion. Put together, it suggests gender does something for F/LOSS that it does not for other communities.

Making Gender Out of the Question

In this section the paper explores the nature of the nerve that was touched by the suggestion that steps might be taken to include women. There are near universal refrains from both men and women within F/LOSS that gender ought to be irrelevant. As in Silva’s account of “color-blind racism,” (Silva 2003) where racism continues
through less overt means, few F/LOSS participants see themselves as sexist or hostile to women, and believe that if gender is not acknowledged, there is no sexism. For example, in a widely read text taken to be foundational, a ‘typical’ F/LOSS member is described by a well-known F/LOSS founding father (Raymond 2005). There is an explicit denial not just of sexism, but of the salience of gender at all. After asserting that the percentage of women in F/LOSS was high compared to other technical professions—an outright falsehood—Raymond claims:

the ties many hackers have to AI research and SF literature may have helped them to develop an idea of personhood that is inclusive rather than exclusive -- after all, if one's imagination readily grants full human rights to future AI programs, robots, dolphins, and extraterrestrial aliens, mere color and gender can't seem very important any more (Raymond 2005).

In this comment, gender is made to be ‘mere’ gender. In our ethnography, both men and women interpret the barest acknowledgement of gender as somehow ‘reiterating’ the ‘artificial’ differences between the sexes, again reducing it to ‘mere’ gender. From an extended online debate about a group that helps women contribute to Debian, a version of Linux:

I think the whole idea of ‘Debian Women’ is flawed. All it does is give / reiterate to people the idea that women are somehow different to men when it comes to computers and should be treated differently.”
And from the same discussion…

Instead of saying “Linux geeks should be nicer to newbies,” [the article] says “Guys should be nicer to women.” No… To ask for different treatment for different genders is SEXIST, and when the stated goal is to minimize sexism, it becomes counter-productive.

The threads of online discussion quoted above reveals a clear ontology. They echo a longer-standing discourse of liberal feminism, a form of early feminism premised on the idea that equality would be had once formal equality of opportunity was achieved, such that gender might no longer matter. Both men and women explained the lack of women in the field in terms of a “misperception” that computers were for men, that there was nothing necessary and biological about the affinity between men and computers. In calling it a misperception, they seemed to suggest that computers already provided a kind of equality, and rejected the idea that a perception could be a social fact carrying ontological weight.

For F/LOSS members, however, gender’s ‘artificiality’ is grounds for dismissal of it. The quote that we have borrowed for the title of this paper is taken from an online F/LOSS discussion of whether to take gender into account in thinking about the future of the community. In it a coder asserts that software patches (that is, lines of computer instructions that fix problems in previously written software) do not have gender. The technological, in other words, is orthogonal to the social. The pressing need is not to consider such artificial constraints, but to make patches which work, whatever their social origin.
Concerns are not, in fact, always about getting down to the business of software creation. Entire conferences are devoted to how to organize and maintain vibrant F/LOSS communities, which debate legal issues and engagement with non-technical publics. This would not be possible if there were a strict social/technical divide. It is only when it comes to gender that this ‘pressing need’ emerges. For example, we held a lengthy discussion with a well-placed leader in one of the largest and most successful F/LOSS projects, in which he glowingly enthused about the international diversity of his members, and how it strengthened their code. He was proud this achievement, and observed that it was “unfortunate” that only 2% of the membership in his organization were women. We raised a number of techniques that could be used to reach out to them, including building on the formal mentoring structures within the community that already helped newcomers. His tone turned. He told us he did not believe in special help for women. He was genuinely concerned about the absence of women, and genuinely valued diversity, and yet was dead set against doing anything that might improve the situation. This was not because he believed our suggestions were likely to be ineffective; his obvious discomfort revealed something much more problematic, such that women’s absence posed less problems than the method to change it.

Women in the community expressed similar sentiments. Some expressed embarrassment with being connected to inclusion efforts through their gender, and others took it as an occasion to assert that they were the only ones responsible for their own careers. The inclusion strategies that have gained ground made explicit that “special help” was not desirable. Debian Women, a group of women and men gathered together to support and mentor programmers, made clear at every opportunity that their group was open to male participation. The gendered name was constructed as a “mere” acknowledgement that women tended to be newcomers, or else “happen” to not enjoy the
aggressive tone of online conversation alongside many men. By downplaying their target audience, they drew on the scope of social concerns that people saw as legitimate to manage in a deliberate fashion, which was how to help newcomers.

The [name omitted] report, too, is used in very specific ways. Advocates for women’s inclusion point to the quantitative elements that showed, for example, just how rampant sexual harassment of women is. By referring to these particular passages of the report, advocates for women’s inclusion are making the claim that there is a very real culture hostile to women, not their “misperception.” It is here where the difficulty starts. To have to argue that gender is both something utterly artificial and meaningless and yet very real puts these advocates in an interpretive bind around what is and is not real. As will become central later, this is a bind that vexes in other ways.

Ways of legitimately dealing with gender, without inviting invective, are informed by the adoption of liberal notions of social organization. Early commentary from both scholarly and media sources has focused on F/LOSS as a moral response to capitalist economies (Kelty 2004: 498) and progressive transformation in orthodox systems of capitalist production. According to Stephen Weber, “by experimenting with fundamental notions of what constituted property, this community has reframed and recast some of the most basic problems of governance” (Weber 2004: vii). F/LOSS has attracted comparisons with classical anthropological tropes of economic organization, such as a gift economy (Raymond 1998), a barter economy (Ghosh 1998) and a kinship system (Zeitlyn 2003). Such a seemingly unorthodox system has left economists asking questions about how F/LOSS can be economically viable (Rossi and Bonaccorsì 2005, David 2006) Many F/LOSS contributors use these tropes to evoke the promise of alternative, fairer, and more democratic social organization.
Still, there is much here that is familiar. Coleman and Golub (2008) have described hacker morality in general, and F/LOSS specifically, as based around a moral liberalism. They argue that much of the F/LOSS cultural ethos relies on early notions of liberty formulated by John Stewart Mill (2008 (1859) and other classical Western liberal thinkers who posit that freedom lies in individuality being as unfettered by others as possible. They show how talk about F/LOSS, either in terms of community making, or intellectual property, or ethics remain are grounded in a liberal intellectual tradition—drawing on notions free speech, meritocracy, privacy, the power of the individual, self-cultivation of knowledge, etc..

In practice, this means that female achievement appears unthreatening as long as it is a matter of ‘individual’ choice. Few were prepared to question whether it was possible for an individual woman, through heroic efforts, to become successful in F/LOSS. For example, one leader publicly calls herself an “open source diva.” The label does two important things. It renders gender an eccentricity, in the sense that ‘diva’—a term loaded with gender connotations—is a rare acknowledgement that she even has a gender at all. Yet it is immediately made absurd by the unseriousness of the job title. It also individualizes her: try being another diva with one already in circulation! This strategy works because it preserves herself as having freely chosen to be eccentric, and makes it difficult to acknowledge that the way she enacts her gender might have come from somewhere other than her personal inventiveness. In turn, in one prominent public venue she called those who acknowledge gender as a problem “whiners” and not “doers,” accusing “whiners” of being “feminazis” who improperly make their complains social by sharing them with others.

We similarly observed in one small firm that the work space was literally divided down the middle along gender lines.' The more technical' developers sat on one side of
the room and the staff who did the documentation and testing work, sat on the other. Yet all fifteen staff members insisted that this was mere coincidence, and that no one had been “discriminated against” (their words). The women insisted that it was their individual choice to work in support rather than development, and that the office geography therefore constituted an accident of multiple individual decisions. Their commitment to locating agency in the individual, a commitment shaped by the broader liberal cultural moorings Coleman and Golub point to, left them unwilling to see the materially obvious ways their occupation had been socially shaped.

In this firm, workers are neither ignorant nor foolish; rather, something is gained in not acknowledging gender, and only acknowledging individual choice. It allows them to see that they could have done the developers’ job if they chose to do so. By claiming that all comers are welcome regardless of social origin, they keep intact the possibility that through sheer individual zeal, nearly all paths are open. The consequence, however, is that suggesting that there is anything systemic about the exclusion of women invites vitriol. This seeming 'blindness' to social categories is at work in other domains, such as online gaming. Nakamura (2009) describes this 'blindness' as neoliberal, reflecting the broader re-entrenchment of liberal ideas about and social relations that began in the 1970s (see Harvey 2005). It does particular, important work for F/LOSS.

The gender politics that we examined throws the liberal ethos into a particular kind of relief. While Coleman and Golub show the shifting ways hackers deploy liberal ideas, the angry discussions that shut down the question of gender inclusion reveal that they are nevertheless still in the service of better software code, and to that end, not open for debate. Intellectual property, maintaining a contributor base, debating how and when code is agreed to be incorporated into the larger project: all these can be negotiated as a normal part of F/LOSS activity to the extent that support the goal of making software.
But they cannot threaten the ‘real business’ of technology production, which, judging from criticisms like “patches don’t have gender,” it seems people believed was at stake.

**From Coding as Craft to Coding as Science**

This section shows how the notion that individual choice keeps all paths open is central not just to what people believe about themselves, but to what they believe about the technology itself. Institutionally and geographically F/LOSS is highly heterogeneous. Nevertheless, ideas about where knowledge comes from, and how one should be situated in relation to it are largely shared and uncontested. F/LOSS members understand technology as having its own moral imperative. Above all, code must evolve. To participants, this does not require any further justification in the way that property rights do. Particularly valued, evolved, pieces of code are called ‘Good Things’. A Good Thing is written with capitals and pronounced with corresponding moral emphasis. Good Things are believed to be as true as they are beautiful, again reflecting longstanding liberal intellectual heritage.

The uncontested way to expand knowledge boundaries is by working openly and sharing the code. Participants believe that code, like a Popperian falsifiable hypothesis, ought to demonstrate its worth only through the scrutiny of others. Other researchers have likened software production to craft production (Coleman 2000) or pleasurable play (Klief and Faulkner 2002). In our study this is indeed part of coders’ imagination. Members often describe their work as “scratching an itch” by producing something tangible and craft-like. Scratching an itch is a common way people account for why they become involved with communities and why they stay. Yet tradition and repetition, key elements of other forms of craft production, have no place other than as building blocks upon which to take one’s own work further. Re-doing work similar to others does not scratch the itch satisfactorily, whereas it generally does amongst craftspeople. In this
way, the craft system looks suspiciously like a system of science, where coding is more about working towards ‘better’ code, not necessarily code that is usable or performs a valued function in society. It is not considered interesting to just make a media player or word processor, but only new kinds of media players or word processors that exemplify some transformation in knowledge. Through the emphasis on subjecting new code to public scrutiny, F/LOSS programmers put themselves on a technological, generative, and creative frontier (Helmreich 1998).

Like a system of science, objects that can be claimed to be novel proliferate more quickly than those deemed repetitive or traditional. New programming languages, for example, proliferate and die off at a rate that confounds everyone involved. No one can claim absolute expertise in such a wide and rapidly developing domain (see also Downey 1998, Born 1996, Ullman 1997). Ullman reports how one has to be quite certain that one’s knowledge at the edge is ‘enough’, even though it never is. We experienced exactly this problem in learning to participate in the F/LOSS community in Paris. What at first was a trade-off between learning the native language of the field and getting on with the ethnography subsequently turned out to be a never-ending spiral of new technical forms that community members were themselves challenged to keep on top of.

With unknowns all around, and an ethos concerned with making perpetually better code, it becomes necessary to establish which code pushes the boundaries and which is mere tradition, or replication. Establishing code as something that pushes the edge becomes a source of lasting legitimacy and reputation. To achieve this kind of legitimacy, participants say the code should be able to speak for itself. That is, the technology itself ought to be the authority, just as the ‘patches don’t have gender’ claim refers authority back to the software. One contributor reported to us an instance where a developer was not able to convince his colleagues about the worth of his code due to his
limited English language skills. However, in his view success was possible because the code could speak for itself. Code can speak for itself to the extent that it ‘works,’ that is, it does have some discernable material effect independent of the coder pressing buttons. That effect is in the form of machine operations that brings something to life on the screen, or other people’s screens. With code that significantly pushes boundaries, however, it is less clear that it can speak for itself through the mere fact that it functions successfully. Edgy code is meant to be unstable, unfunctioning even: no boundaries are being pushed if there are no uncertainties.

There are implications if the code speaks for itself. It obliges other coders to go digging round the lines of text within the source code to identify how it works, and thus the knowledge and techniques that might be brought to other coding problems. In principle, how the code works is right there, embodied in the lines of code which have been made open to other programmers. It is incumbent upon the reader of lines of code to spot it. Producing unstable versions of software, that is, software that does not function very well, in turn affords opportunities for others to become producers through necessitating engagement with those lines of code.

The edge moves forward because the knowledge revealed in code is considered to be transferable programming concepts, allowing anyone to create nearly anything. The ability to function generatively gives it the ability to claim that it has some independent effect on the real world. The knowledge is contained within it as a kind of a subtext—it has no direct effects in the sense of making a machine do something free of the creator, but it nevertheless it is key to make new possibilities come forth. If one were to draw the parallel with science, code is simultaneously the experiment and the account of it to be peer reviewed. Successful code is code that can operate at the edge without the
interventions of its makers. In doing so it ought to reveal something new, laying possibilities for the edge to shift and generate new edges.

**Securing A Place at the Edge: Meritocracy and Pushyocracy**

The fact that it is the human interpretation that makes lines of code discernable and readable is not something F/LOSS participants care to spend a good deal of time considering. Participants grant the technology itself the ability to push the boundaries, not people reading the code. Yet if technology is meant to speak for itself on one hand, and yet be so advanced as to be unable to do so in practice, just who or what pushes technology forward becomes incredibly problematic. F/LOSS communities have ways of mitigating against that uncertainty, enabling people to make claims that they are indeed on technology’s edge. There are two key ways of claiming to work at the edge: by emphasizing the truth status of code, and by externalizing it. Both of these, readers might anticipate, are called into question in discussions about gender.

To take the first, truth claims are central in debates about the worth of software. Computer programming has a rather explicit way of constructing facts (Latour 1987). Truth statements (‘if ‘x’ is true, then y’) are a central feature of programming languages. The software functions, and thus can act upon the world, if the truth statements are true. Often making software ‘work’ is a zero sum situation; it either runs or fails to run, and it members describe with exhilaration moments of finally making something ‘work’. As if this aspect of code served as a wider metaphor, judgments are made on the basis of what counts as ‘good code’ and ‘bad code’ as if all ‘good code’ works and all ‘bad code’ does not. Recall from the previous section that good, knowledge-pushng code rarely works as a stable, usable system. Nevertheless, the imagination that it does, or could, sets the tone of online discussion. A person is not likely to find their way to the edge if the code they write never works. Contributors can point to his own ‘working’ code as ‘true’ to
thwart other possibilities coming forth, to position oneself at the edge, and shift others away from it. This de-emphasizes the multiple ways that someone could get something to work.

There is a telling but necessary slippage between the technology and oneself. Although it is considered ideal that good code would speak for itself, in reality authors must vociferously defend their proposed code in order to demonstrate knowledge and establish what good coding is: that is, they must create the edge through talking. This is especially necessary the more edgy the code is, as there is always more uncertainty there.

The practice of re-using code, and thus the reason for it to be openly readable in the first place, is not necessarily about developing shared understandings of the principles behind them as claimed in discourse about openness. We can see this in the absence of documentation of code that would facilitate shared understandings. What is more important is the way that code cultivates individual agency for oneself and others which, in the end, only legitimately can be expressed through yet more code. The itch that people say they scratch is precisely the moment knowledge comes to be recognized as self-generated, and in personal narratives these moments are constructed as profoundly transformational—people talk about being able to say “I did that” as the moment they become a coder. Many of our research participants began reading manuals and books from a very young age. Yet they recognize it only as an independent autogeneration of knowledge between person and machine, not a communication between author and reader. In this way, code not only crystallizes achievement, and confers technical leadership, but produces a transformative knowledge of self—an “enlightenment” in Osborne’s (1998) sense.

Despite the belief that code reveals knowledge, participation in discussion can be even more important than actual lines of code. We have found cases where people who
do contribute a great deal of code, but do not get involved in flame wars about the worth of code, are perceived as less knowledgeable. If code embodies knowledge as a kind of subtext, flaming asserts that subtext in no uncertain terms. While online communication is known to lend itself to flaming (Winter and Huff 1996), it appears from our research participants that in F/LOSS it is particularly rewarded. For example, one of our participants, known for frequent flaming, explains his behavior as a way to elicit objective truth. He considers that if someone is wrong, not only should they should be told but the point should be insisted upon until they accept it, in much the same way that code can be made to ‘work’ with enough debugging. The important thing is to be seen to get to the bottom of the matter, and for truth to be established, regardless of how disruptive or ‘anti-social’ it may be. Indeed, flames are carried out until one person backs down.

Acrimonious talk is a prime reason women, as well as some men, say they leave. In their leaving, they often cite a central F/LOSS value: that communities insufficiently get down to the real business of technical work, and instead rely on flaming to make decisions about which piece of code is worthy of becoming part of the software. Critics call it a “pushyocracy” as opposed to a meritocracy. This accusation is incredibly undermining to the notion that better code is being produced through open scrutiny. It suggests that merits are not being found in a purely external plane revealed by whether or not the code works.

While the truth status of code secures, people’s places at the edge, so does the specificities of how code gets circulated. Similar to the way truth statements operate, the code either gets incorporated into the community’s software, or it does not. Leadership has either occurred, or it has not, which can be read from the ‘truth’ of whether the technology has been incorporated. The thing-like transaction, as opposed to a
discursive co-creation of knowledge, again supports this view that authors of code are right or wrong, and privileges positivist truth as a resource for securing a place on the edge.

Communicating via software code further individuates both producer and recipient. In gift economies, gifts are given to create ties and obligations amongst people. But in F/LOSS, code is given as a way of cutting ties. Openness is treated as a way of saying “it’s yours now, you do what you like.” For example, our research participants did not necessarily know, or care, who the receiver is. Whether the code or the knowledge embedded in it gets taken up by subsequent authors is anyone’s guess. To them, free speech does not require any particular audience to be free. Any other communication with users of the code, such as or create manuals that enable others to read and interpret it, do not really count as a part of making code free. This is work that women often do.

When conversation turns sour, the lack of connection between giver and receiver is often wielded as a rhetorical weapon. A common way to construct oneself as more knowledgeable than others is to tell people who are asking questions about how some aspect of the code works is to say “RTFM” (read the f***ing manual). Few feel obliged to write one, however. That is, there are no assumed obligations or ties upon which question-askers might draw if authority only comes from the seeming merits of this or that code. Community members are decidedly not free to build ties that might oblige others to explain themselves. If someone does not like being in one project, the proper course of action is to simply start another project elsewhere, not create an obligation for that community to include you. This particular form of exchange means that others can push the technology along further only as individually willful agents who have taken it upon themselves to read the f***ing manual. Freed of interpersonal stakes, code authors do not have to be certain of what happens once the code is transferred: the satisfaction is
in the fact that it works, and that it was ‘really’ launched out into the ether. Whether or not someone builds on it, or whether or not something like the promised innovation actually happens, is only a diffuse, future concern.

If Patches Don’t Have Gender, They Can’t Fix the Problem: Why Women’s Inclusion Troubles F/LOSS Production

While Western liberal notions of freedom and individual self-sufficiency are resources for F/LOSS members to think about themselves and their actions, it is the specificities of ‘free’ exchange that give the liberal rootings their practical teeth, and make them far less multivalent than in Coleman and Golub’s account. Here only one kind of knowledge is acceptable: externalized and truthful. That good code, in practice, has to be asserted and circulated by extra-technological means is undermining to the belief that technology is external to the self and self-evidently true as the right way of doing things. The act of flaming can be acknowledged, indeed glorified, but is believed to merely help along technology’s own progress. There is a profound need to misrecognize the means by which truths come into being if technology’s edge is to be a real place. Addressing the exclusion of women is troubling because it upsets this necessary misrecognition. Women’s response to the hostile talk by leaving cannot matter because it if it did, participants would have to confront the edge as a construct made by social force rather than revealed ontological fact.

The cost of this acknowledgement would have to be paid in three ways: in people’s reputations, in notions of the self and its relation to technology, and in notions of freedom. To take the first, being explicit about the work of interpretation in F/LOSS means that people risk losing a key way of establishing reputation: referrals to a technology-given truth. If reputations are to be preserved as having technical integrity,
technical progress cannot be just an ‘artificial’ outcome of verbal prowess. A person cannot verbally beat the other into submission if an externally given truth does not give the ‘right’ (and rightness). That is, the temporary abeyance of agency (Miazaki 2000) to the technology itself is broken if women are to be deliberately included.

For both men and women, critics of the pushyocracy and advocates for it, true technological progress is not subject to the winds of declaration. To concede that code is in fact an outcome of verbal prowess would risk losing the ability to “scratch an itch,” through one’s effects upon an external world. Recall that communities value the notion that ‘anyone’ can start tinkering about with bug fixes. If gender is real, more than just arbitrary artificiality to be overcome through individual volition, it suggests there are in fact attachments and ways of doing things that were not necessarily chosen, as in the firm with the gender divided room. Perhaps if gender was real and not an 'artificial' difference between men and women, maybe people did not, after all, just RTFM, but established the 'truth' amongst themselves. Acknowledging this, as the Debian Women project obliquely does, invites accusations of divisive wrongheadedness because it reveals the precariousness of the figure-it-out-for-yourself story that matters significantly to both men and women.

Finally, social ties evoke a closed system from which developers feel they have broken free through the very way technology circulates. ‘Open’ is believed to mean that technology circulates on its own accord, set free to have its own impacts on the world. In order to open the doors to ‘anyone,’ the possibility that there are social loops creating knowledge and passing it along—the very mechanism that both excludes women and could serve to include them—threatens the basis on which it is possible to claim that the door is open. That ‘free’ technology is not in fact freed of social pattern implies that such social unfetteredness did not in fact take place. These are issues that go well beyond
recruitment and retention questions to the core of what participants themselves believe makes F/LOSS so innovative. The edge is a troubling place indeed.

**Conclusion: Does Women’s Exclusion Suggest “Free” as in Market?**

While F/LOSS has been hailed as a progressive, transformative development that questions traditional forms of technology production, this paper has shown that, when we look at it from the perspective of gender, there is much that re-entrenches old ways of doing things, well beyond rampant sexism. The research found that the inclusion of women is a worrisome prospect because F/LOSS places such emphasis on cutting rather than building the ties between people. What is ‘social’ and what is not is indeed being constructed here as the gender and technology literature teaches us to expect. Yet gender further shows us just how central notions of individual agency are in F/LOSS. F/LOSS technology is not just outside the realm of the social but downright *freed* from it by making code exchangeable without knowing to whom it is being exchanged, what they will do with it, and avoiding creating any obligation for coders to explain themselves to anyone unlikely to help them gain reputation. This emphasis on a particular view of freedom, grounded in a combination of classical liberal thinking and a culture of positivism, is the source of the radical forms of gender exclusion that are particularly strong in F/LOSS.

As a way of concluding, I suggest that perhaps this very freedom raises the prospect of F/LOSS being a wider re-entrenchment of capitalist social relations, and not necessarily something that calls them into question as previous commentary has suggested. Throughout the paper we showed how longstanding liberal tropes informed how F/LOSS culture works and puts the individual at the center. To the extent we are now in an era of neoliberalism, and a re-entrenchment of those values and the systems
built upon them (Harvey 2005), it may be possible to interpret F/LOSS as part of a more general rise of neoliberalism. This is a promising line of enquiry for a future study; here it is only possible to suggest grounds on which such a study would be warranted.

While developers like to say that the ‘free’ in free software refers to the ‘free’ as in speech, not as in beer, the commitments revealed by gender practices show that it is also not so far from ‘free’ as in market, where neoliberal discourses of the ‘free’ market treat individual actors as unencumbered by social obligation. For example, Mackenzie (2006) notes the importance of decontextualization in the widely used Java Virtual Machine, such that, like F/LOSS, producers and users have only thin connections to one another: ‘virtual’ in Mackenzie’s argument. Similarly, the current prominence of open APIs (application programming interfaces) suggests a further unencumbrance by social relations. Open APIs reveal enough about a computer system, whether it be Facebook or a mobile phone, to allow other programmers to write smaller pieces of software for these larger platforms, just as open source licensing allows for building one technology on the basis of another. Just as in F/LOSS discourses, commercial companies now consider that “anyone” can create software for an increasing range of technologies, such as Google’s Android, or Apple’s Apps Store. By opening one’s own technical systems to other developers, businesses like Google have put themselves in a position to be freed of the obligation to deal with people who otherwise would be software providers. They no longer have to be encumbered and ‘bogged down’ with contracts and lengthy negotiations about what features to include and what not to include. There need not be a shared vision or understanding of what should or should not be on a phone. This is an ideal situation in a neoliberal system that values the ability to hire and fire anyone or any company at will. It saves costs not just because much of it is unpaid (Terranova 2000), but avoids the associated costs of having to build a longer standing system of intra-firm
relations. Perhaps most importantly, however, it preserves Google as an actor autonomous from its suppliers. Open platforms mean that companies can now legitimately say to their own suppliers, RTFM, just as F/LOSS men say to F/LOSS women. In this larger scenario, who might leave as a result of these new arrangements, and what the cost of such openness might be, is not yet clear.

These comments are, of course, speculative. The point to making them is to show the continued work gender might do as an analytic lens. The language of openness in F/LOSS, and open innovation more generally, invites us to account for new forms of collaboration and sharing, while gender teaches us to look for where and why ties are broken down, and what these breakdowns do, whether for new economic forms as in the Android example, or for pushing technology’s edge for its own sake, as in the case of F/LOSS. In this way, gender is never a problem of ‘mere’ gender.

Notes
This is the line-by-line set of instructions that programmers manipulate to produce software.

\(^{1}\) See authors, 2009.
References


