Open is not that Open: the construction of "open source" at the epistemological intersection of agency, language and technology

Introduction: Openness at the Intersections

"Titstare is an app where you take photos of yourself staring at tits," (Miller, 2014)

Imagine that you are a new person in a room, trying to figure things out and feeling confused by new technology and new terminology. When you finally figure out what question to ask, the response to your question is "Read the fucking manual." Unfortunately, RTFM[1] is a common response when you ask an "obvious" question in a technical space. A recent variation on RTFM is LMGTFY ("Let me Google that for you"), but the same unwelcoming message underlies it: if you do not know how to find "easy" technical answers yourself, you do not deserve to get help[2].

Free/Libre Open Source Software (F/LOSS)[3] is a category of software that has a few central characteristics – it is generally made by people who are working for free, it is distributed without cost, and the source code (effectively, the innards of the software) is available for anyone to change and improve. The free software movement was started in the 1980s as a reaction against corporations controlling the means of knowledge production and a "moral response to capitalist economies" (Nafus, 2012, p. 669). It was

[1] Commonly said to new developers asking questions. When a person new to a technology joins an online space (most commonly a chat room) and asks a question, a few things happen: 1. The reputation of the new person is evaluated. This reputation includes gender, links to other well-known community members, etc. 2. The recentness??? of the question - did someone else just ask this? If so, the response is "Read the fucking scroll back." And 3. The "obviousness" of the question.

[2] Please note that there is no cultural precedent for politely telling someone to RTFM. "Please go scan the docs and come back if you have questions" is a rare response.

[3] Free/Libre Open Source Software is the full name. Many scholars and community members strongly identify with the Free and Libre terms, although "OSS" is a more common abbreviation. It is the abbreviation that Nafus uses, and so I will also use it throughout.

a fundamental shift to insist that the knowledge we produce via code should be free as in speech and free as in beer.

Responses like “RTFM” signal is a clear divide – between the people who “know” and the people who are trying to know. They signal a boundary and, I will argue, on one side is the seeker and on the other knowledge, culture, community, and opportunity; a place where people deeply share a culture and vernacular.

TODO: This is where I’m going to insert my cool thesis statement.

While FL/OSS projects have some numerable characteristics, the concept of open source

Were we to map the geography of concepts, the concept of open source would exist at the intersection of agency, language and technology. In this paper I’m going to explore the first two via the lens of the third. WHAT?

The boundaries between who and who cannot ask questions are a discursive reinforcement of the boundaries between

As sociotechnical communities grow and develop, they naturally create their own ways of speaking and interacting with each other, like regional dialects of the same language. Both Carol Cohn's “Sex and Death in the Rational World of Defense Intellectuals” and Dawn Nafus's “'Patches don't have gender': What is not open in open source software” describe cultural and disciplinary boundary work through manipulation of agency and legitimation of language. The addition of Thomas Gieryn's “Boundary Work and the Demarcation of Science from Non-Science: Strains and Interests In Professional Ideologies of Scientists” will help to illustrate how boundaries are formed, destroyed and constantly negotiated.

create an "open culture" by purifying what counts as valid contributions. In many ways, this serves as a unifying concept between Gieryn and Cohn. Both illustrate a way that a container is fortified by purification and distillation of its contents;

Gieryn's boundary work

"It was clear that this was a man's world. Women could come, but only if they followed dude rules. It was only cool if you could roll with the bros."(Kornblum, 2013)

Boundary work is, quite simply, the act of drawing boundaries between things like academic disciplines or political parties. Boundary work can be used to help define what "counts as" a particular area of work (boundaries as inclusion) and define what does not "measure up" to standards (boundaries as exclusion). Gieryn presents boundary work not just as in/out sorting mechanisms but as ideological battles in which the weapons that both sides use are ever-evolving and context-sensitive.

Gieryn's

Because Gieryn's conception of boundary work is central to exploring the epistemological monoculture that is present in Cohn and explicitly named in Nafus, [TODO: EF say and then show why it is central] let us first explore his foundation in depth.

Boundary work has existed for as long as there have been disciplines. In Victorian England, John Tyndall helped to draw boundaries around science to distinguish it from religion and mechanics. According to Tyndall, the aspects of science that make it scientific include its empirical nature, skepticism, objectivity, and practical usefulness. In contrast to religion, says Tyndall, science is *useful* (Gieryn, 1983, p. 785) In a different context, Tyndall described an entirely different picture of science's intrinsic science-ness:

science is knowledge that is grounded in experimentation, valuing discovery for its own sake. Science doesn't have to worry about being *useful*, he says, it has nobler concerns (Gieryn, 1983, p. 787).

Tyndall's contradictory descriptions of science and scientific knowledge give great weight to Gieryn's argument that even as we try to draw boundaries around "science" there is no objective science to which we can point. Science itself is conceptualized as a monolithic ideological concept that we can skew and manipulate for our ideological and rhetorical (and political, and strategic…) goals. We will see later how useful concepts are as weapons and tools of tyranny.

The uses and results of boundary work are not only ideological. Part of boundary making involves creating a technically-knowledgeable elite separate from the general population. This separation means that only those with adequate training can evaluate technical claims - there is no need to poll an uneducated populace to see if a claim is scientifically valid. By being a scientist doing work in a boundaried area, the scientist's claim is given respect and is thus unchallengeable by non-scientists. As a result, scientific knowledge makers are insulated from public interrogation and from the impacts and implications of their knowledge. [TODO SM: Are there advantages to this? What does Gieryn say about the advantages?]

When groups of thinkers are insulated and isolated, an inbreeding of ideas can result and although Gieryn did not explicitly name it, he is describing Nafus's epistemological monoculture. [TODO EF: Support with examples here. Exclamation point]

In the next section, we will see three ways that boundary work is evident in Nafus

and Cohn and how this manifests specifically in open source software communities.

The manipulation and transformation of agency

First, we will explore the the manipulation and transformation of agency – who (or what) has agency, how agency is transferred. –TODOX. As boundaries change, agency can shift to support this-TODO

Gieryn describes the way that science carries its own intellectual authority – science is valid because it is science[4]. This is consistent with Nafus and Cohn. In Nafus, code has agency and "technology [has] its own moral imperative." (Nafus, 2012, p. 676). Discussing projects, engineers talk about how "code runs," as though the code itself can choose to do so and they "grant code its own agency" (Nafus, 2012, p. 678) The defense intellectuals also use language as a manifestation of agency/positionality when "the speakers' technostrategic language are positionally allowed, even forced, to escape" the idea of themselves as victims of nuclear war (Cohn, 1987, p. 706). Code produces "transformative knowledge of self" (Nafus, 2012, p. 679) in the same way that technostrategic language use produces a great sense of agency. All speakers of the defense language share a transformed sense of agency and their detached positionality as the only ones who can control the bombs. Similarly, F/LOSS programmers are unified by their control of code and ability to produce technological artifacts.

In F/LOSS contexts, programmers[5] self-identity with their ability to "write code that runs"[6] and set themselves apart from the individuals who do other important software work like QA[7] or UX[8]. In this way, there is boundary organization of programmers with

[4] A similarly invalid statement happens in technology linking openness to freedom to liberation.
[5] I use the term programmers, coders, and software engineers interchangeably
[6] This is common terminology in technology
[7] Quality Assurance
[8] User Experience Design – the study of the best way to present screen interfaces to end users.

a distorted perception of their power and importance. Because their code "runs," because they are the creators of pseudo-sentient agents, they often conceptualize themselves as having a greater sense of control and centrality. [CITE - bitcoin]. Their boundary organization also, as Gieryn theorized, separates them from the general public and insulates them from non-technical interrogators. Programmers, insulated from the users, the implications, and the long-term impacts of their work, are free to maintain their technological elitism and superiority because of this culture. From a societal standpoint, non-technical individuals are being controlled by the technology they need; programmers, however, are situated as being in control of it. Note the parallel between these programmers controlling code and the defense intellectuals controlling weapons.

One of the ways that agency manipulation is seen in technology contexts is the obfuscation of agency of individuals with their jobs TODO In Nafus, the boundaries between which workers performed development tasks and which workers performed "support" tasks like documentation or QA[9] was divided almost perfectly along gender lines (Nafus, 2012). However, the women were hesitant to "acknowledge the materially obvious ways in which their participation had been socially shaped" (Nafus, 2012, p. 676). What we see here is an example of the women wanting to retain agency over their participation and their roles in the technological space (another example of the manipulation of agency common in F/LOSS contexts). It is disempowering to believe that you chose to do QA work because you were socially funneled into that area[10].

The positionality resulting from the illusion of control and manipulation of

[9]Quality Assurance. Typically code moves from development (where it is created by programmers) to QA. QA work is feminized as "support" work in which women check the men's technological production.
[10] A lot of work is being done, systemically, to give programmers (especially those just starting out) more agency and choice in their careers. TODO – Black girls code.

agency reinforces the boundary between those with and those without control of technology and technological artifacts.

The legitimation of language

The other part of this intersection is the legitimation of language. There are three ways we can see this happening: by narrowing the kinds of responses that are acceptable to a certain question, by specifying kinds of language that can be contained in those responses, and by overloading meanings onto words.

To illustrate the first, Cohn describes the ways that defense "discourse has become virtually the only legitimate form of response to the question of how to achieve security" (Cohn, 1987, p. 712). Cohn discusses how using common language to respond to a defense topic was unacceptable and the defense intellectuals would not consider her an equal in conversation. In F/LOSS communities, the de-facto (read: legitimate) way to get the code you wrote incorporated[11] into the main project, is to advocate for it in an issue queue flame war. This may sound like a dated "Hacker" movie concept, but is still a very real and very prevalent method of communication about technical matters. Nafus argues that language is also used to draw boundaries between those who "can take it" (which is also a form of masculinized boundary) and those who cannot. This is inherent in Cohn's analysis of defense language - if you cannot handle the sexual overtones, then you are not serious enough to be a part of the conversation. Similarly, if you cannot handle the issue queue flame wars, you are not serious enough to be an open source programmer. This is a legitimation of the types of responses that monocultures are reinforcing.

[11] Also known as "merging" when you take TODODO

The second form that the legitimation of langue takes is the regulation of the types of language that should be contained in a response. As Cohn finds that she is required to use traditional defense discourse as her type of response, she also finds that she is required to use specific defense words and phrases. She had to use "escalation dominance" rather than an a layperson version like "controlling the pace of the battle.[12]" (Cohn, 1987, p. 708)

There is an unexpected side-effect to this legitimation. Becoming subject to the "tyranny of concepts" means that "language shapes your categories of thought" (Cohn, 1987, p. 714). Concepts that were previously charged become neutralized by your participation in this context. This is an example of language as weapon (language as thief of agency) – the defense intellectuals' culture was so immersive and the language so powerful that Cohn found herself unable to access her own concepts of peace and justice after her enculturation (Cohn, 1987, p. 706).

The third legitimation of language is what I am calling "overloading of meaning." In programming, engineers have the ability to "overload" a function[13]. This means that you can have two functions with the same name but different contents, producing different results. Similarly, overloading a word is the ability to give two or more meanings to the same word, typically beyond those one would find in a dictionary.

An example of overloading of meaning is the technological use of the terms "master" and "slave." In computing terminology, you implement a master/slave database

[12] I made this layperson version up. A google search for alternative phrases for "escalation dominance" revealed that the term is so prevalent that it's difficult to even find a definition. The closest I found was one defining it as "dominating the speed of escalation."

[13] A function is a named set of instructions to do a specific task. For example, a function named `sum` might add two numbers together and return the result.

configuration most often for performance or security reasons. The "master" is the primary source for information, and can have an infinite number of "slaves" that are available for servers to be read from or written to[14]. Technologists use master/slave terminology readily, and without thinking about the origin of the words or their connotation[15]. Companies (Reuters, 2003) have requested that terminology be changed, and open source communities continue to debate whether their "freedom" to create software extends to their "freedom" to use master/slave (Drupal.org, 2014)[16].

Defense intellectuals also overload meaning. In their case, they use innocuous words as stand-ins for weapons terminology: "crew members call the part of the submarine where the missiles are lined up in their silos ready for launching 'the Christmas tree farm.' What could be more bucolic-farms, silos, Christmas trees?" (Cohn, 1987, p. 698).

Note that while Cohn's defense intellectuals describe violent technology with innocent abstractions, technologists describe (relatively) innocent technology with historically violent terms. This is a manifestation of the same mechanic – enculturating group members to reinforce group membership by using appropriate language.

Lang: Construction of Openness & access to concepts? There are many connections between the construction of openness in open source, and the lack of women[17] (and other URMs). As Nafus says, "openness relies on a steadfastly closed epistemological frame" (Nafus, 2012, p. 681) and this is true of any context in which a

[14] See Figure 1 for a master/slave diagram.
[15] A common use would be in a sentence like "If there's too much traffic, we'll just add slaves as needed."
[16] In many projects, the master/slave language has been replaced with "primary/replicant" and this has the advantage of also being technically correct.

new boundary is being defined. In order to maintain the boundary, in this case between "open source" and "everything else," ideological purity is required so that people can determine what side of the boundary they want to be on. Cohn experienced the clear delineation between the defense intellectuals' world and the outside world in an embodied way – she lost access to concepts that she had before entering the community. She found herself engaging with them in ways that were foreign to who she was.

In this way, the "tyranny of the concepts" [TODO CITE], the concepts become weaponized. [TODO – Say this is why Gieryn is central] They can be used to separate you from your ideas and concepts, as Cohn experienced when she tried to access her previous notions of peace in the defense context. In F/LOSS spaces, the concept of "openness" carries with it a mountain of ideological baggage. There are certain behaviors and perspectives that are crucial to openness and violating them becomes violating the sanctity of openness itself.

Conclusion

One of these perspectives that is relevant to our discussion here is the belief that "the technological […] is believed to be orthogonal to the social" (Nafus, 2012, p. 674). This manifests in many ways, including contributing to the stereotype of the programmer as a solitary individual, creating code alone in a basement.[18] One of its more toxic manifestations is in the notion of technology being a space that is gender and race blind. Coders believe that gender ought to be irrelevant (Nafus, 2012, p. 673) because the agency and focus belongs on the code and how it runs, not on the creators of the code. It is a not-uncommon situation to hear men exclaim that women are sexist for bringing up

[18] See Figure 2

gender, that people of color are racist for bringing up race when "I just want to write code." [TODO]

The objects we create (including the code that we write) are not value-neutral, and is not without context. As a result, the technological is not orthogonal to the social, but they are co-constructed with each new programming language or software project. The concept of openness and its constituent ideology becomes a weapon against individuals who prefer different methods of technology construction, and who want to shape the technosocial context from within.

Nafus, Gieryn and Cohn all describe methods and results of boundary-makings. The implications are different for each, but there is a commonality in motivations and justifications. In sociotechnical contexts, speaking the language is, on one hand, an indicator of technical skill and on the other hand, a way to exclude newcomers. Being able to use technical terms to describe how to roll a patch[19] can signal your technical competence to those around you and can exclude novices from the conversation.

I believe that thinking about F/LOSS politics and barriers as a form of boundary making will help draw them out of background activity - bringing attention to the actions that many participants take for granted as how things are done. There has been a sea change in F/LOSS communities in the last several years. Communities have introduced Codes of Conduct (CoC) for online spaces and in-person events. Communities have created teams to deal with CoC violations and procedures for handling situations of

[19]To "roll a patch" is to prepare a text file that will be applied to a file containing code. The text "patches" a hole in the code file.

sexual and behavioral misconduct. The news is full of prominent men in tech finally being recognized for their cultural mis-contributions to the culture of their organizations.

A recent Harvard Business School study found that each toxic worker costs an employer $12,000 more than a neutral (aka, non-toxic) employee (Shapiro, 2015).

Rather than conceptualizing boundaries as fixed, and the culture within them as static, Gieryn gives us a way to think of boundaries as fluid and strategic. When we are consciously aware of boundary-making activities (as Gieryn asserts we are), we can alter them as our needs for power, control and clarification evolve.

We find

Bibliography

Cohn, C. (1987). Sex and Death in the Rational World of Defense Intellectuals. *Signs*, *12*(4), 687–718.

Drupal.org. (2014, May 28). Replace "master/slave" terminology with "primary/replica." Retrieved November 10, 2017, from https://www.drupal.org/node/2275877

Gieryn, T. (1983). Boundary-Work and the Demarcation of Science from Non-Science: Strains and Interests in Professional Ideologies of Scientists. *American Sociological Review*, *48*(6), 781–795.

Kornblum, J. (2013, September 14). It's time to eliminate bro-culture from the tech industry. Retrieved November 10, 2017, from https://www.salon.com/2013/09/14/how_bro_culture_has_been_hardwired_to_the_tech_industry_partner/

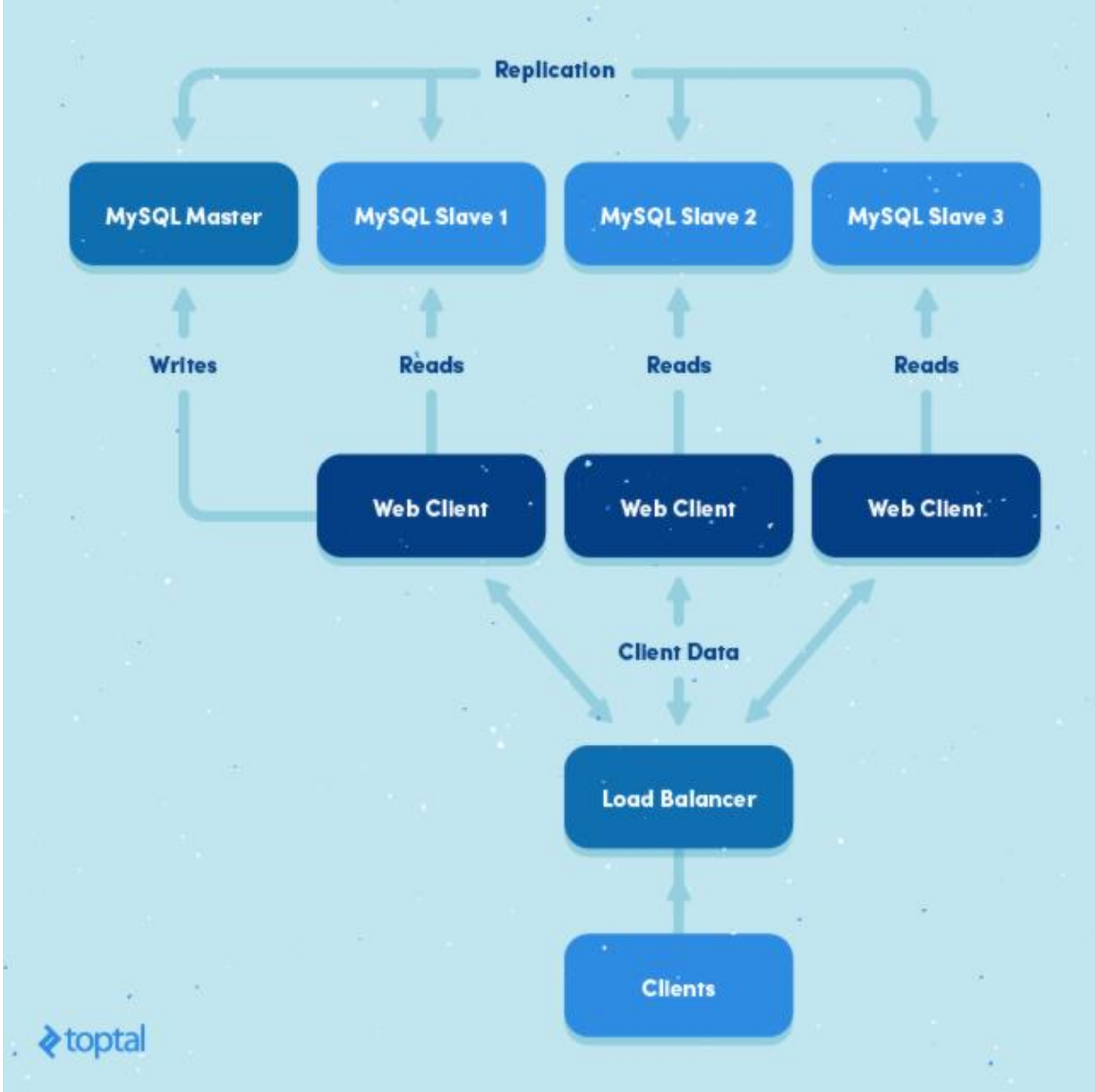
Miller, C. C. (2014, April 5). Technology's Man Problem. *The New York Times*. Retrieved from https://www.nytimes.com/2014/04/06/technology/technologys-man-problem.html

Nafus, D. (2012). "Patches don't have gender": What is not open in open source software. *New Media & Society*, *14*(4), 669–683. https://doi.org/10.1177/1461444811422887

Reuters. (2003, November 26). CNN.com - "Master" and "slave" computer labels unacceptable, officials say. Retrieved November 10, 2017, from http://www.cnn.com/2003/TECH/ptech/11/26/master.term.reut/

Shapiro, A. (2015, December 16). Harvard Business School Study Highlights Costs Of Toxic Workers. *All Things Considered*. NPR. Retrieved from https://www.npr.org/2015/12/16/460024322/harvard-business-school-study-highlights-costs-of-toxic-workers

Figure 1



https://uploads.toptal.io/blog/image/91937/toptal-blog-image-1452523127725-dddcc82ac412d0d1d8dd2cb33c5f7084.jpg

Figure 2

http://theshelternetwork.com/wp-content/uploads/2014/08/south-park.jpg