Bots, bespoke code, and the materiality of software platforms

by R. Stuart Geiger, UC-Berkeley School of Information

Published in *Information, Communication, and Society* (2014)

Recommended citation: Geiger, R. Stuart (2014). Bots, bespoke, code and the materiality of software platforms. *Information, Communication & Society.* DOI: 10.1080/1369118X.2013.873069

Abstract

This article introduces and discusses the role of *bespoke code* in Wikipedia, which is code that runs alongside a platform or system, rather than being integrated into server-side codebases by individuals with privileged access to the server. Bespoke code complicates the common metaphors of platforms and sovereignty that we typically use to discuss the governance and regulation of software systems through code. Specifically, the work of automated software agents (bots) in the operation and administration of Wikipedia is examined, with a focus on the materiality of code. As bots extend and modify the functionality of sites like Wikipedia, but must be continuously operated on computers that are independent from the servers hosting the site, they involve alternative relations of power and code. Instead of taking for granted the pre-existing stability of Wikipedia as a platform, bots and other bespoke code require that we examine not only the software code itself, but also the concrete, historically contingent material conditions under which this code is run. To this end, this article weaves a series of autobiographical vignettes about the author's experiences as a bot developer alongside more traditional academic discourse.

Keywords

code, materiality, software, bots, Wikipedia, algorithms

*"*Most people think that to understand law, you need to understand a set of rules. That's a mistake … The law is best understood through stories – stories that teach what is later summarized in a catalog of rules." -- Lawrence Lessig, *Code and Other Laws of Cyberspace*

Introduction: what is code in Wikipedia?

An installation of MediaWiki, the software platform powering Wikipedia, has over 600,000 lines of code in about 900 files, mostly written in PHP and released under an open-source license. It is easy for a Linux systems administrator to configure and install their own instance of MediaWiki, comparable to other platforms like Wordpress, Drupal, or Joomla. In a matter of minutes, a seasoned sysadmin can set-up their own wiki and have a site that will look and feel like Wikipedia, on the surface. There will be wiki pages that users can collaboratively edit. The history of a page's revisions can be accessed, and undesirable changes rolled back. Users can communicate with each other using talk pages. Administrators can protect pages from editing, or block problematic users. It will be a wiki, the kind of website that has come to stand in for an entire revolution in content creation, management, economics, and politics (for a critical analysis of this discourse, see Van Dijck & Nieborg, 2009; Tkacz, 2013).

However, as I and many other founders of their own MediaWiki-based sites quickly learned, many of the features and functionalities that are taken for granted in Wikipedia are nowhere to be found in a 'stock' installation of MediaWiki. While Wikipedia does run on a version of MediaWiki, the version it runs is a highly customized one that relies on far more unofficial or *bespoke code* than stock. By my estimate, the code that runs alongside the official platform rather than being directly integrated into MediaWiki is easily an order of magnitude larger than the ~600,000 lines of code that comprise MediaWiki. This code – some of which fundamentally changes how the wiki operates as a wiki – takes many forms, including PHP extensions, template scripts, user scripts, standalone tools, browser extensions, and fully automated bots. This code is written in a multitude of programming languages, coded in a variety of environments, and is often executed on computers that are relatively independent from those run by the Wikimedia Foundation to host Wikipedia. Without this 'extra' code, editors are not able to even leave the much celebrated '[citation needed]' tag, an artefact and practice that has become a cultural icon of wiki-based collaboration.

My estimate of over six million lines of bespoke code does not even include all the already existing non-wiki software used by members of the Wikimedia community and the Wikimedia Foundation on a daily basis to coordinate administrative and infrastructural tasks. These are the standalone software systems supporting coordination via mailing lists, blogs, and internet relay

chat (IRC) channels, as well as tasks like server administration, vulnerability testing, and donation processing. These platforms are themselves often highly customized versions of already existing open-source software projects like Wordpress, mailman, and nagios. As Kazman and Chen argue in their 'Metropolis model' (Kazman & Chen, 2009) of software-as-a-service development, if software is analogous to a built environment, then platforms like Wikipedia are more analogous to cities than individual buildings in terms of how they are designed and constructed.

While this broad and diverse media ecology in which Wikipedians assemble – as a public, a community, an organization, and an institution – is fascinating and deserves detailed study, this article is slightly narrower in scope. In this article, I focus on a growing phenomenon in contemporary software development I term 'bespoke code'. Bespoke code extends or transforms the operation of software platforms, but runs on top of or alongside existing systems instead of being more directly integrated into and run on software-side codebases. I then discuss a particular kind of bespoke code: bots, or fully automated software agents, which complicate many of the traditional distinctions and dichotomies we use when conceptualizing the role of software in society.

Next, I argue that bots are a particularly vivid way of problematizing how discourses of 'the platform' attribute an always-already stability, unity, and existence to the software code that works to structure and constitute the interactions we have with each other in mediated environments. Like Latour's speed bumps (Latour, 1992), Becker's grand symphonies (Becker, 1982), or Bowker and Star's medical classification systems (Bowker & Star, 2000), I relate a more situated and materialist perspective, which illustrates how Wikipedia as a platform is a diverse and temporarily stabilized assemblage of code. As I illustrate in a series of autobiographical reflections about my experiences as a bot developer strategically weaved throughout this academic argument, the code that governs Wikipedia is less an authoritative, monolithic voice and more of a diverse and ever-shifting cacophony. Understanding how code operates in a social context requires more than parsing through lines of PHP and decoding algorithmic instructions. We must also tell stories about that code, following it wherever and whenever it leads us.

Vignette zero: the bot multiple

Hi, I'm Stuart, and I used to be a bot developer – my bot and I were once part of Wikipedia's platform. We're not anymore, and that's what I'll be telling you about later in these vignettes. In concert with the academic argument about the materiality of code outlined in the previous

section, I will be telling a more personal story about a bot. You will encounter these intermissions throughout this article as this text oscillates between two discursive modes. This is a technique I draw from Annemarie Mol's The Body Multiple (2002)*, which she uses to emphasize the multiple epistemologies and materialities simultaneously at work in her study of atherosclerosis – an argument that resonates with my study of bots.*

This literary technique parallels my argument about how bots and bespoke code operates alongside more established and formal regimes. Bots are situated; they come on the scene alongside more concrete systems, inserting themselves into average, everyday practices and platforms. Bots aren't usually part of some master plan – if they were, they probably wouldn't be bots. They often just seem like a good idea at the time, but you don't really know if they'll work until you try them. And when you first see a bot, you may not be sure how to react, or even what it is you're looking at. They may make you uneasy, and you may not be sure they even belong, especially if they insert themselves into a well-defined format or routine. It is easy to think of them as ancillary, non-essential add-ons, and sometimes they are indeed useless or annoying or just get in the way. But together, bots and software platforms like MediaWiki are more than the sum of their parts. Understanding this multiplicity requires moving between subject positions, discourses, and epistemologies.

Bespoke code and bots

I define the term *bespoke code* as software code that runs alongside a platform or system, in contrast to code that is integrated into server-side codebases and runs on the same servers that host the platform or system. The word 'bespoke' traditionally describes highly customized fashion, such as a bespoke suit; the Oxford English Dictionary defines it as 'goods; ordered to be made, as distinguished from ready-made' (OED, 2013). There is sparse use of 'bespoke code' to refer to custom-made software based on client specification, but I use the term differently: to identify and consolidate a large range of practices I have seen in not just Wikipedia, but a variety of other contemporary software development environments. Like a made-to-order dinner jacket, 'bespoke' indicates that this code is highly customized and specifically written (and rewritten) to fit some already existing entity. In my experience as a developer of Wikipedia and Twitter bots as well as a user of browser-based add-ons, scripts, and extensions that fundamentally change the way I experience the web, this constant customization has emerged as the one most salient commonalities between these disparate software development practices.

Bespoke code has emerged in response to software-as-a-service and cloud computing, in which software programs are increasingly run on servers instead of local machines. In this model, users

typically access an interface via a web browser, like with Gmail and Google Docs compared to Microsoft Outlook and Word. Open-source advocates including Richard Stallman have critiqued this model of software development as one in which 'the server operator controls your computing' (Stallman, 2013), as it initially appears that the functionality and affordances of such software programs are even more locked down than in traditional closed-source programs, because the code does not even run on the user's computer. However, bespoke code complicates this, as the literature on software mashups (Wong & Hong, 2008) has illustrated: meta-extensions like Greasemonkey enable what has been called 'layman tuning' (Díaz, Arellano, & Iturrioz, 2008) of websites for various purposes. However, mashups and browser-side tuning are only two instances of the broader phenomenon of bespoke code; next, I detail others that operate in Wikipedia.

Many of the commonplace socio-technical practices that Wikipedians have developed to help them collaboratively build an encyclopaedia – even the code that supports leaving the famous [citation needed] tag in articles – are not supported in the stock version of the MediaWiki platform. The [citation needed] tag is supported by a relatively simple template script, of which there are over one hundred thousand in the English-language version alone (Lanzara & Patriotta, 2007). This code is developed and stored on a wiki page in a special section of Wikipedia, which is called every time an editor leaves the text {{citationneeded}} in an article – the double curly brackets are parsed as a function call to the script at Template:Citationneeded, rather than markup.

A second form of bespoke code is bots. Like almost every collaboration or discussion site, Wikipedia relies on anti-spam and counter-vandalism algorithms, but these are also not a part of the core MediaWiki platform. Much of this code takes the form of bots: independently operating, fully automated software programs that review edits on a *post hoc* basis, as human editors do. These bots constantly query Wikipedia's servers for a list of recent changes made to articles, analyse the changes made based on their own heuristics and methods, and sometimes send a request to revert those that passed an algorithmically defined threshold. A third class of bespoke code is the semi-automated tool, which runs as browser extensions or standalone programs and pre-script certain well-defined and routine tasks. The Huggle tool (Geiger & Ribes, 2010) queries the recent changes feed and presents reviewers with before-and-after views of edits in a queue. With one or two clicks, a Huggle user can not only revert an edit, but also send its author a pre-written warning, nominate the page for deletion using pre-written rationales, or request that an administrator block the user.

Bots and bespoke code makes up 'the hidden order of Wikipedia'

Today, Wikipedia's bespoke code plays a role in almost every aspect of the encyclopaedia; bots, specifically help create new articles, edit existing articles, enforce rules and standards, patrol for spam and vandalism, and generally work to support encyclopaedic or administrative work (Geiger, 2009, 2011; Halfaker & Riedl, 2012; Müller-Birn, Dobusch, & Herbsleb, 2013; Niederer & Van Dijck, 2010). Because bots are built to algorithmically enact a particular vision of what encyclopaedia articles or wiki-based collaboration is and ought to be, they have profound influence on what Wikipedia is and how it operates, both an encyclopaedia and a community. For a community of hundreds of thousands of people from across the world collectively authoring millions of articles, bots make it possible to achieve a certain level of uniformity in style and content – that is, when Wikipedians all agree that, for example, reference lists should be in alphabetical order or articles should use American instead of British spelling. Bots also serve key governance roles, independently enforcing discursive and epistemological norms, particularly for newcomers. They have grown so prevalent that as of 2011, less than a third of all new users receive their first interpersonal message from a human using the MediaWiki interface to write a message. Instead, over a third of new editors receive their first message from a human using a semi-automated tool or script, and the remaining third are "welcomed" by fully automated bots – which by policy are not allowed to send blanket welcome messages, only messages that they did something wrong. (Geiger, Halfaker, Pinchuk, & Walling, 2012)

However, bots should not just be seen as force multipliers that merely operate on content, letting a technically skilled Wikipedian merely magnify their intentions – although this certainly does happen. Bots also dramatically extend the functionality of MediaWiki as a platform for interaction by creating new spaces in which others interact, for specific purposes. Wikipedia is often alleged to be an anarchy lacking formal rules and structures – instead governed by some mystical 'wisdom of crowds' (Surowiecki, 2004). However, there are many bureaucratic and formalized procedures in place that make up 'the hidden order of Wikipedia' (Wattenberg, Viegas, & McKeon, 2007). Much of this work is supported by code not built into the MediaWiki platform, notably template scripts, tools, and bots.

This bespoke code supports and structures how Wikipedians review and assess article quality, assemble and coordinate in sub-groups called WikiProjects, choose the day's featured article, curate the Did You Know and In the News sections, promote users to administrators, decide which un-encyclopaedic pages should be deleted, and many other tasks. In some cases, these bots subvert the fundamental idea that wiki pages are documents that users edit, using these flat text files as semi-structured databases. For example, bots implement queuing mechanisms for

processing and distributing administrative requests, such as requests to block a particular user for vandalism. Bots also produce broad views of Wikipedia that emphasize some particular aspect or phenomenon, aggregated out of a multitude of digital traces. Finally, bots work to stitch together otherwise disparate software platforms, sending a message to a specialized IRC channel when a new user edits the Help Desk page or posting to Twitter when a new article is nominated for deletion.

Vignette one: my first bot

AfDStatBot was the first Wikipedia bot I had ever built, which operated in 2008–9 when I was first studying Wikipedia's Articles for Deletion (AfD) processes. Hundreds of Wikipedia articles are nominated for deletion every day, and editors have to discuss whether or not each one should be kept or deleted. As its name implies, AfDStatBot was created for my quantitative research purposes, to archive and capture statistics on these deletion discussions in real time. However, it also served a few secondary purposes, including giving this data back to the community I was also ethnographically studying. I could say that this was part of my participant-observation as a Wikipedian, but it just made sense that if I was capturing and analyzing this information, I ought to use those resources to make the lives of other Wikipedians easier. It's just what you do. I had seen hundreds of these unofficial bots, tools, and scripts that Wikipedians developed for themselves and each other, supporting incredibly specific tasks and practices. I thought I had something to contribute.

AfDStatBot was one of many minor bot-based features. It generated a near real-time noticeboard of active and recently closed deletion discussions, with statistics such as how long the discussion had been open, how many Wikipedians were in favor of keeping vs. deleting the article, when the last comment was made, and so on. Then for the few who opted in, it curated personalized watchlists that helped Wikipedians keep track of all the deletion discussions they had personally participated in. Finally, in what was more of an exercise to see if I could do it, the bot posted to Twitter (@WikiWars) every time it saw a new article nominated for deletion. It was not the most popular or important bot on Wikipedia by far — most of the bot's followers on Twitter were random auto-follow bots — but it did provide a service for a niche audience. A few years later, another Wikipedian would build a much better bot for these tasks, Snotbot, which is still in operation today.

The materiality of software

Software is more than code. This claim about the materiality of software is not a novel one, but it is distinct from the even more accepted notions that it is important to study software development as a socio-cultural practice (Crowston & Howison, 2005; Mockus, Fielding, & Herbsleb, 2002) or to study the economic, political, social, cultural, psychological, and organizational impacts of software on our world. My argument is ontological, one about existence and essence: bots are a vivid reminder that what software is as software cannot be reduced to code and divorced from the conditions under which it is developed and deployed. It is a materialist argument, opposing the 'trope of immateriality' (Blanchette, 2011, p. 3), a discourse that alternatively celebrates or laments the disembodied nature of information technology. As Blanchette reviews, the supposed immateriality of mediated technology is nothing new – it was used to describe the telegraph – but he cites Hayles in arguing that this trope in which digital information is proclaimed to be 'free from the material constraints that govern the material world' (Hayles, 1999, p. 13) is not just a casual metaphor of technologists but a fundamental assumption in contemporary post-human society.

Critiques of code as the essence of software or algorithms are far from new. Design-oriented artificial intelligence (AI) researchers (Elliott & Brzezinski, 1998; Sengers, 1998) have long critiqued their more computationally oriented colleagues in the 'identification of the internal code of the agent as what the agent really is' (Sengers, 2000, p. 14). The ways in which the artificial agent appears to the user are just as important and essential as the code behind it. In a different but related vein, scholars have examined software not only through its code and algorithmic routines, but also through the practices and meetings of network operators (Mathew & Cheshire, 2010), the magnetic storage mechanisms of hard drives (Kirschenbaum, 2008), the copyright licenses of open-source software communities (Kelty, 2008), clerical and support staff in datacenters (Ribes, Jackson, Geiger, Burton, & Finholt, 2013), the role of hackerspaces in China (Lindtner & Li, 2012), and the roles of universities, businesses, and government agencies of Rio de Janeiro (Takhteyev, 2012) or the San Francisco Bay Area (Saxenian, 2006).
Furthermore, ever since Marx's socio-political analysis of factory machinery and other engines and artefacts of capitalism (Marx, 1973; MacKenzie, 1996), scholars have interrogated the material infrastructures and artefacts that are taken for granted in a variety of social institutions: prisons (Foucault, 1977), museums and art galleries (Becker, 1982; Star & Griesemer, 1989), hospitals (Garfinkel, 1967), scientific research (Latour & Woolgar, 1986; Shapin & Schaffer, 1985), public infrastructure (Winner, 1986), economic markets (Mackenzie, 2006), and organizations and firms (Orlikowski & Scott, 2008), to name a few. Abstract, high-level, seemingly immaterial entities like art, culture, discipline, science, truth, value, power, or profit all rely on materially existing infrastructures, artefacts, people, and practices – often operating

behind the scenes – that often fundamentally shape and structure how those seemingly immaterial abstractions operate. Just as the legal system is more than the text of laws and precedent, software systems are more than the text of code.

Vignette two: a legitimate alternative account

If you want to know about the life of AfDStatBot, you might start with the bot's userpage on Wikipedia. If we think of AfDStatBot as a user, this user profile best represents the bot. Profiles take many different forms, but they have long been the spaces in which software platforms represent users. In Wikipedia, these 'user pages' are technically little more than special wiki pages, meaning that anyone can edit anyone else's profile – as I typically did for AfDStatBot. Like most bot userpages, there are several curious elements that indicate this profile is different from that of most human Wikipedia editors. There is a large, red power button that administrators can use to stop the bot from operating if it begins to malfunction. That's not a formal requirement, but it is standard practice in Wikipedia bot development. There are also warning templates at the top that tell the reader that the account is a bot operated by me, and that it is a 'legitimate alternative account' – a notice that is somewhat of a legacy from time when bots were far more contested and controversial in Wikipedia.

'Platforms' obscure bespoke code

Bots are not incorporated into what we commonly refer to as the software platform, as they are instead run by developers on servers that are often independent of those used to run the website. Bots problematize what I call the *discourse of platform sovereignty*, in which platforms are cast as spatially bounded territories, code is cast as having the force of law over such territories, developers are cast as sovereigns who govern territories with code-as-law, and users are cast as subjects who enter territories and are subsequently governed by code. The algorithms of bots can be unpacked from their black boxes and seen as rules that have the force of law, but such a metaphor can de-materialize the infrastructural conditions that make this kind of regulation possible – like talking about the law without talking about the courts or the police. These representations and metaphors are powerful, shaping what we see when we look at Wikipedia as a platform, and have implications for how we understand the nature of authority. The discourses of 'platforms' work to obscure and consolidate a variety of assumptions, practices, and ideologies (Gillespie, 2010), particularly the material conditions necessary for the code to operate as a kind of architecture.

At the heart of this assumption of sovereignty is a fetishization of Lessig's famous 'code is law' (Lessig, 1999) slogan, one that sees code as abstract rules that autonomously govern and regulate according to their own immaterial logic. This abstraction of code into law separates code from the material conditions under which it was not only produced, but also put into force; yet these conditions are what make code the kind of thing that is even capable of governing and regulating. With bots, thinking of platforms as governed by code-as-law can obscure the conditions under which that code-as-law comes to be run at all. Without examining bots, Wikipedia's unexpected success comes to look like the kind of 'self-organizing' system that spontaneously harmonizes according to what Benkler calls an 'aggregate effect of individual action' (Benkler 2006, p. 4). In fact, bot operators in Wikipedia occasionally lament how their bots become invisible and taken for granted, as users see them as always existing inherent features that were written once and forever built into MediaWiki, rather than code that must be constantly run on their own servers.

Bots and other bespoke code problematizes the implicit assumptions present in the metaphor of the platform, which often serves to obscure the specific work that is done to make the platform appear to be a stable, coherent infrastructure – and the more successful infrastructures are, the more invisible they become (Star, 1999). In studies of Facebook, YouTube, and Twitter, more and more researchers refuse to take for granted the pre-existing stability of a term like 'platform' (Hands, 2013). Theorists have critiqued not only the logical code that structures interaction on social media sites, but also the socio-technical logics: 'the norms, strategies, mechanisms, and economies' (Dijck & Poell, 2013, p. 1). This is related to earlier work emphasizing the ideological (Chun, 2004) or performative (Mackenzie, 2005) nature of software, interrogating the work that must take place in order for something like 'platforms' or even 'software' to exist as a monolithic entity.

As Gillespie argues in his critique of the term 'platform' as deployed by YouTube's stakeholders, metaphors like the platform 'represent an attempt to establish the very criteria by which these technologies will be judged, built directly into the terms by which we know them'. According to Gillespie, 'platform' is increasingly deployed in ways that silently consolidate a number of otherwise disparate technical and political concepts, implicitly erasing the tensions that may emerge if these concepts are more explicitly discussed. He notes that there are 15 uses of the term in the Oxford English Dictionary, and that the term 'platform' had political, architectural, and figural implications long before it was a computational term. Calling something like YouTube a platform carries significant implications, not only in what each of the individual connotations imply, but also the assumed stability and coherence implied by such a synthesis. The bespoke code of bots governs Wikipedia (and is governed by Wikipedians) just as stock

code does, but this mode of power and governance does not easily lend itself to a discourse of platform sovereignty.

Vignette three: the bot approval group

In a single line in an infobox on the righthand side of my bot's user page, there is a link to a formal approval from the Bot Approval Group, which I had to petition before legitimately running my bot. If we think of bots as ontologically similar to that of laws, in a kind of Lessig-esque fashion, then this approval perhaps best captures the essence of the bot. In order to be accepted as a bot operator and for my bot to be accepted as a legitimate bot, I had to translate the bot into text that others could come to understand and evaluate. If I ran a bot without approval, my bot may have been discovered by humans (or anti-bot bots) and both I and the bot could end up banned from Wikipedia. So this code-as-law had to be approved by the Bot Approval Group, but this didn't take long – although when my proposed rate of editing for updating the statistics pages was deemed too high, I lowered it. If my bot's task was more controversial, like a bot that would remove all images without proper copyright documentation from every article, then there may have been more discussion, but there was no controversy here. And all of this was recorded, captured, and preserved for a number of socio-technical ends, serving to represent the collective intent of Wikipedia's administrative apparatus, such that it could be easily linked to from my bot's userpage.

'Platforms' obscure space and place

My critique of discourses of platform sovereignty is related to the critiques of 'digital dualism' (Jurgenson, 2012), which interrogate the discourses that draw sharp boundaries between the 'online' and the 'offline' world – particularly those that deploy a moralistic argument claiming what occurs 'in real life' is inherently more social, substantive, significant, and healthy than what occurs in 'the virtual world'. The architectural metaphor of the platform becomes problematic when it implies that online interaction occurs exclusively within 'virtual' spaces that are well-defined and delimited by the boundaries of the platform. Research on social networking critiques this tendency to see interaction as taking place in some distinct space from 'real life', emphasizing the integration of practices and activities across mediated and face-to-face contexts (Baym, 2010; Ellison, Steinfield, & Lampe, 2007; Lundby, 2011; Wilson, 2005). Research on the occupy movements, for instance, has shown how the use of multiple media is not just convergence (Jenkins, 2006), but a complex integration of simultaneously overlapping mediated and co-located environments (Bennett & Segerberg, 2012; Thorson et al., 2013).

Researchers have noted that the idea of a virtual or online community – or, in organizational research, the virtual team (Fiol & O'Connor, 2005; O'Leary & Cummings, 2007) – is outdated and needs reformulation. Some are self-reflexively using metaphors of networks to counter these metaphors of bounded spatiality, such as Burrell's reconceptualization of the 'fieldsite as network' (Burrell, 2009) in her ethnography of youth and Internet cafes in Ghana. Other ethnographers have advocated shifting analytical frames from 'co-location to co-presence' (Beaulieu, 2010) to better capture the range of multi-faceted interactions taking place in a given organization or community. These approaches have opened up new avenues for research into online interaction, and I argue that much value can be gained from similarly thinking of a platform as a networked assemblage, performed and made present in various settings and contexts.

As I and other ethnographers have found, the idea that Wikipedia only takes place on wikipedia.org – or even entirely on the Internet – is a huge misunderstanding (Konieczny, 2009; Reagle, 2010). Wikipedia is not a virtual world, especially one located entirely on the wiki. A substantial amount of activity does occur 'on-wiki' or in a diverse array of other computer-mediated channels, but Wikipedians also constantly meet up in face-to-face contexts as well. Local groups hold regular meetups in cities across the globe, which range from small groups 'talking shop' over beer to massive outreach-oriented 'edit-a-thons' (Donald, 2012; Phillips, 2013). The annual Wikimania convention/conference has brought Wikipedians (and contributors to associated wiki projects) from around the world together since 2005. The role of the Wikimedia Foundation, with a staff of over 100 in its offices in downtown San Francisco, is often dramatically overlooked in discussions about why and how Wikipedia works.

Software platforms are also not best understood by making sharp distinctions between the online and offline or the virtual and 'real life'. In Wikipedia, what comes to look like a unified platform that structures and governs interaction is actually the product of a wide variety of code running on a multitude of different servers. This has specific consequences for those who are concerned with how these systems operate, particularly because as part of this assemblage, bespoke code can break or disappear in ways that platform code typically cannot. As scholars of infrastructure argue, (Bowker & Star, 2000; Star & Ruhleder, 1996) cases of breakdown reveal aspects that are otherwise invisible when such systems are functioning properly. A particularly vivid case of breakdown can be seen when the volunteer-run servers hosting the counter-vandalism bot ClueBot NG – which in January 2011 was singlehandedly responsible for making 13.7% of all reverts (or rejections of a change to an article) – inexplicably went down for days at a time in Spring 2011. This had significant consequences for the editing community, taking Wikipedians almost twice as long to remove undesirable content without this gatekeeping agent (Geiger & Halfaker, 2013). Such a moment of breakdown reveals that algorithmic power in Wikipedia is

highly distributed and decentralized, in stark contrast to a more integrated and institutionalized gatekeeping system like YouTube's ContentID.

Vignette four: moving out and away

AfDStatBot has since left Wikipedia (and Twitter), although its traces in those spaces do remain. As another mode of visual representation shows – screenshots of its Wikipedia contribution history and Twitter feed – my bot abruptly ceased all activity on June 1st, 2009. The last recorded action the bot made was at 5:30 pm EST, when it saw a new AfD open for the article 'K38IM', a low-power repeater for an Illinois-based Christian television network. It saved it in its database, posted this development to Twitter, and, had it stayed online for just a few minutes more, it would have included this new discussion in its regular half-hour update to the AfDStatBot information pages on Wikipedia. But it didn't. Instead, for what has now been just over four years, the bot went offline and never logged in again. As with so many other Wikipedia editors who suddenly stop contributing to the project, the bot's userpage remained in the same state it was in for years, representing a user who was no longer present. But these traces relate what the bot did every single day, serving as a diary of sorts. By 1 June 2009, the bot had seen 23,240 articles nominated for deletion, each of which it tracked to the best of its ability.

1 June 2009 was an emotional day for me, one I'll remember for some time – and no, not because it was the day my bot died. Back then, I was living in Washington, DC. My bot had been running on my own personal desktop computer in a corner of a tiny apartment in Dupont Circle, where I had been living for almost two years with my partner at the time. We had recently broken up, and our lease ended on 1 June. We had both found new places to live and had been moving our stuff out of the apartment bit by bit, but June 1st was the date we had to get everything out. At the time, unplugging my desktop computer and taking the bot down was the least of my concerns. It seemed that everything in my life was in a state of transition, and I was already shirking obligations that were far more pressing than my bot. Besides, my Master's thesis on Wikipedia – the original reason I developed the bot – was done, defended. I didn't need to collect any more data, and as everyone feels after a big project, I was ready to take a break from Wikipedia. To be honest, I don't know if I even remembered that the bot was still running on that computer when I shut it down and pulled the plug. There were a lot of things I lost that day, from the many physical and digital objects I threw away or destroyed, to parts of myself I wouldn't know were gone until weeks, months, even years later. That evening, as I entered the sterile apartment in Southeast, DC, which would never really be home, I was struggling to work out a lot of changes in who I was, but the fact that I wasn't a bot operator anymore didn't even register to me.

Platform sovereignty and the materiality of code

The *discourse of platform sovereignty* arises at the intersection of two pervasive metaphors: sovereignty derives from the familiar 'code as law' metaphor from Lessig, while the concept of platforms is based on spatial metaphors of 'cyberspace', in which interaction is seen as taking place in a territory. These metaphors mutually reinforce each other: if we think of YouTube or Wikipedia as a platform that we stand on and inhabit, as a territory in cyberspace, then it makes sense to use sovereignty to conceptualize of the modes of power and control that operate within. Early writings about governance and the Internet explicitly made such comparisons, most notably Barlow's 'Declaration of the Independence of Cyberspace' (Barlow, 1996). Graham (2012) has written on the history of spatial metaphors as a way of making sense of the Internet; he argues that metaphors casting the Internet as an unregulated territorial space emerged alongside discourse that framed the Internet as a problem of governance for states.

Lessig's famous 'code as law' (Lessig, 1999) slogan often undergirds contemporary discourses of platform sovereignty, although Lessig has a much more materialist foundation to his analysis of code – which is often quite lacking by those who reference his slogan. Lessig's writings in *Code* implicitly cast cyberspace as governed territory, but *Code 2.0* (Lessig, 2006) not only defends but also embraces these territorial metaphors as productive ways of talking about the implications of where code is run from. As an example, Lessig references the way many discussion-oriented websites have implemented 'democracy like' (285) software systems, like Slashdot's distributed moderation and meta-moderation features (Lampe & Resnick, 2004; Poor, 2005). However, Lessig claims that there is a difference between how these systems architecturally structure the activities of their users and the conditions under which they are fundamentally implemented. In this view, 'Democracy is the practice of the people choosing the rules that will govern a particular place' (Lessig, 2006, p. 285) and the people of Slashdot do not have the freedom to change this particular democratic system. That is up to the people who own and administer the site's servers, as he claims that 'on the Internet … the "owner" of the space is the sovereign' (285). In this sense, Lessig's argument and his use of discourse of platform sovereignty is actually a materialist one at its most fundamental level, as it is based on the implications that arise from the historically contingent fact that websites and other Internet-mediated platforms are regulated by code that runs on servers owned and operated by specific individuals or firms. However, he laments that this model – in which those who own the servers get to dictate what code runs as law – is the dominant mode in which individual websites are governed.

For Lessig, code has the power to regulate because the people who own the servers have sovereign control over what code is run, and this leads to 'merchant-sovereigns' dominating

platforms almost everywhere on the Internet – that is, except for Wikipedia, which he rightfully but briefly notes and then passes on. Lessig does not mention that the servers powering Wikipedia are owned and operated by the Wikimedia Foundation, which is seen as ultimately responsible to the community. However, this is only part of the explanation, as until relatively recently, the Wikimedia Foundation was small and primarily occupied with keeping the servers running, making few controversial decisions about how MediaWiki operated. The code-as-law that regulates Wikipedia is instead as much the bespoke code of bots and scripts as it is the platform-based code that powers MediaWiki or Slashdot's 'democracy like' discussion forums.
That said, there are times when the Foundation staff had to make a controversial decision about whether to modify the basic functionality of MediaWiki. However, decisions that affect the entire community (or even just entire language versions of Wikipedia) can take years to sort out – as did, for example, the controversies over the rollout of the Flagged Revisions extension (Müller-Birn et al., 2013; Schindler & Vrandecic, 2009). For most of Wikipedia's history, controversial platform-level changes to MediaWiki's code were so rare that there is still no agreed-upon model for how those kinds of decisions are even to be made – as we would presume would be in place in Lessig's ideal model of platform governance. Instead, much of the code that needed to be written to govern the work of writing and maintaining Wikipedia as an encyclopaedia and a community was bespoke code, written by developers and executed from computers all over the world.

Vignette five: code/coda

It was much easier for me to find the bot's source code and even the database it used, as I had many complete backups I made when I began data analysis for my thesis. However, I didn't want to start this story by talking about code and data. But if you want to know what AfDStatBot was, looking at the code and data will tell you a number of things you may not be able to otherwise know. What it meant to make this bot socially literate can be seen in how it used traces that were left in the creation of internal MediaWiki categories to detect new deletion discussions. Wikipedia-wide infrastructural policies about how much load bots could place on the server can be seen in rate-limiting functions that are bundled with each request. The bot interacted via Wikipedia's Application Programming Interface (API), which had been specifically designed to enable bot development – a situation that is not always the case in most other platforms. Someone who can read PHP may be able to walk through my code and be understand what that bot was doing. You can find all kinds of socio-technical artifacts in that code – in two senses of the term, as they are both technological artifacts as well as artifacts of a certain social, historical, and political situation.

AfDStatBot could have been revived. In fact, it still could be – I have the source code. I took that desktop computer apart, sold some components, and upgraded it to a gaming machine. Once I had everything settled and got Internet access in my new apartment, I could have started the bot back up again. Even during the move, I could have "just" taken these few hundred kilobytes of code and run them on a different server. But I didn't. If I had, the bot may still be running today. Scotty may never have made Snotbot to do the work it wasn't doing anymore. But if I had not chosen to run that bot from that tiny apartment I shared in Dupont Circle, would it have been the same bot? What if, from the beginning, I had decided to run my bot on the toolserver, a shared server funded and maintained by a group of German Wikipedians for all kinds of purposes, including bots? If so, the bot may have run the same code in the same way, producing the same effects in Wikipedia, but it would have been a different thing entirely. If you want to know what AfDStatBot was, you have to understand how it was the kind of bot that, when things got rough, when life got in the way, it was something I literally pulled the plug on without so much as a second thought.

Conclusions

Bespoke code is everywhere. In one sense, this means that the bespoke code is increasingly prevalent, as more and more code is being deployed alongside existing software platforms rather than being directly installed into server-side codebases. This complicates the metaphor of sovereignty in which the owners of a server have exclusive control over the code that structures interactions. But in another sense, the bespoke code is everywhere because it is not to be found on these sovereigns' server farms, rather running from a multitude of geographical locations. And this material fact matters. Opening up the black box of Wikipedia's platform, for example, should not be thought of as just an exercise in parsing through code powering MediaWiki – or even all the bespoke code that runs alongside it. The reading of code can be quite revealing, but it requires a relational, networked, infrastructural understanding of code and how it is situated in many different kinds of spaces. The server farm has traditionally been a privileged site of inquiry for platform studies – or non-inquiry, in the case of those who try and fail to study search engine ranking algorithms, for example (Gillespie, 2014) – and this article advocates expanding our frame to other spaces where bespoke code is written and run.

Bespoke code is important, but not only because the code itself often plays important roles in mediated interaction and hence must not be ignored for the sake of completeness. Bespoke code is also important because it challenges many of the default assumptions and discourses of platform sovereignty that undergird many conversations about the sociality of software. For example, at a first glance, Wikipedia is easy to celebrate as an instance of a minimally regulated,

laissez faire mode of production and organization, as its 'anyone can edit' model has been generally accepted as the great determinant of its success. However, this is only the case if we limit our understanding of code as law to the code built into the server-side platform. Instead, similar to how MacKenzie (2006) has shown with the material and semiotic infrastructure that had to be built to structure and sustain financial markets, Wikipedia's 'minimally regulated' peer production environment rests on top of a wide variety of bespoke code that has been specifically developed and deployed to produce, sustain, and enforce particular understandings of what 'the wiki way' is and ought to be. In broader conversations about how platforms govern and are governed, we must seek out and critically investigate all the bespoke code that is being developed in places far away from the traditional sites of platform-based power.

Opening up the black box of Wikipedia's platform involves understanding how all of this stock and bespoke code collectively constitutes the kind of coherent, well-functioning platform that looks like it is governed only by some mystical 'wisdom of crowds' (Surowiecki, 2004). Because bespoke code does not operate in a strictly sovereign model, it often does not appear to be governance or regulation as such. Taking the platform for granted submerses all the work performed by these bespoke codes into the infrastructure, making Wikipedia seem like the kind of community in which everything spontaneously self-organizes out with little to no rules or regulatory structures. And if we see Wikipedia as the kind of platform that is governed by just enough architectural code to let rationally acting individuals spontaneously self-organize, with order arising out of the uncoordinated coordination of the multitude, then why not remake all social and political institutions in such a manner? Against these ideologically charged discourses, we must emphasize the concrete, material, local, and specific conditions that make projects like Wikipedia operate in the manner that they do. And to do that, we need to tell stories.

References

Barlow, J. (1996, February). A declaration of the independence of cyberspace. Retrieved from http://homes.eff.org/~barlow/Declaration-Final.html

Baym, N. (2010). *Personal connections in the digital age*. Cambridge: Polity.

Beaulieu, A. (2010). From co-location to co-presence: Shifts in the use of ethnography for the study of knowledge. *Social Studies of Science*, 40(3), 453–470. doi: 10.1177/0306312709359219

Becker, H. (1982). *Art worlds*. Berkeley: University of California Press.

Benkler, Y. (2006). *The Wealth of Networks*. New Haven, CT: Yale University Press.

Bennett, W. L., & Segerberg, A. (2012). The logic of connective action. *Information, Communication & Society*, 15(5), 739–768. doi: 10.1080/1369118X.2012.670661

Bespoke, adj. (2013). In Oxford English dictionary online. Retrieved from http://www.oed.com/view/Entry/18156

Blanchette, J.-F. (2011). A material history of bits. *Journal of the American Society for Information Science and Technology*, 62(6), 1042–1057. doi: 10.1002/asi.21542

Bowker, G. C., & Star, S. L. (2000). *Sorting things out: Classification and its consequences*. Cambridge, MA: The MIT Press.

Burrell, J. (2009). The fieldsite as a network: A strategy for locating ethnographic research. *Field Methods*, 21(2), 181–199. doi: 10.1177/1525822X08329699

Chun, W. H. K. (2004). On software, or the persistence of visual knowledge. *Grey Room*, 18, 26–51. Retrieved from http://www.brown.edu/Departments/MCM/people/chun/papers/software.pdf doi: 10.1162/1526381043320741

Crowston, K., & Howison, J. (2005). The social structure of free and open source software development. *First Monday*, 10(2), 1–21. doi: 10.5210/fm.v10i2.1207

Díaz, O., Arellano, C., & Iturrioz, J. (2008). Layman tuning of websites. In *Proceedings of WWW 2008* (p. 1127). New York: ACM Press.

Dijck, José Van, & Poell, T. (2013). Understanding social media logic. *Media and Communication*, 1(1), 2–14.

Donald, A. (2012). Throw off the cloak of invisibility. *Nature*, 490(7421), 447. doi: 10.1038/490447a

Elliott, C., & Brzezinski, J. (1998). Autonomous agents as synthetic characters. *AI Magazine*, 19(2), 13–18.

Ellison, N. B., Steinfield, C., & Lampe, C. (2007). The benefits of facebook "friends": Social capital and college students' use of online social network sites. *Journal of Computer-Mediated Communication*, 12(4), 1143–1168. doi: 10.1111/j.1083-6101.2007.00367.x

Fiol, C. M., & O'Connor, E. J. (2005). Identification in face-to-face, hybrid, and pure virtual teams: Untangling the contradictions. *Organization Science*, 16(1), 19–32. doi: 10.1287/orsc.1040.0101

Foucault, M. (1977). *Discipline and punish: The birth of the prison*. London: Penguin.

Garfinkel, H. (1967). "Good" organizational reasons for "bad" clinic records. In H. Garfinkel (Ed.), *Studies in ethnomethodology* (pp. 186–207). Englewood Cliffs, NJ: Prentice Hall.

Geiger, R. S. (2009). The social roles of bots and assisted editing programs. In *Proceedings of WikiSym 2009*. New York: ACM Press.

Geiger, R. S. (2011). The lives of bots. In G. Lovink & N. Tkacz (Eds.), *Wikipedia: A critical point of view* (pp. 78–93). Amsterdam: Institute of Network Cultures. Retrieved from http://www.stuartgeiger.com/lives-of-bots-wikipedia-cpov.pdf

Geiger, R. S., & Halfaker, A. (2013). When the levee breaks: Without bots, what happens to Wikipedia's quality control processes? In *Proceedings of WikiSym 2013*. New York: ACM Press.

Geiger, R. S., Halfaker, A., Pinchuk, M., & Walling, S. (2012). Defense mechanism or socialization tactic? Improving Wikipedia's notifications to rejected contributors. In *Proceedings of ICWSM 2012*. New York: AAAI.

Geiger, R. S., & Ribes, D. (2010). The work of sustaining order in Wikipedia: The banning of a vandal. In *Proceedings of CSCW 2010*. New York: ACM Press.

Gillespie, T. (2010). The politics of "platforms". *New Media & Society*, 12(3), 347–364. doi: 10.1177/1461444809342738

Gillespie, T. (2014). The relevance of algorithms. In Tarleton Gillespie, Pablo Boczkowski, & Kirsten Foot (Eds.), *Media technologies* Cambridge, MA: The MIT Press. Retrieved from http://mitpress.mit.edu/books/media-technologies

Graham, M. (2012). Geography/Internet: Ethereal alternate dimensions of cyberspace or grounded augmented realities? *The Geographical Journal*, 179(2), 177–182. doi: 10.1111/geoj.12009

Halfaker, A., & Riedl, J. (2012). Bots and cyborgs: Wikipedia's immune system. *Computer*, 45(3), 79–82. doi: 10.1109/MC.2012.82

Hands, J. (2013). Introduction: Politics, power, and "platformativity". *Culture Machine*, 14, 1–9.

Hayles, K. (1999). *How we became posthuman: Virtual bodies in cybernetics, literature, and informatics*. Chicago, IL: University of Chicago Press.

Jenkins, H. (2006). *Convergence culture: Where old and new media collide*. New York, NY: NYU Press.

Jurgenson, N. (2012). When atoms meet bits: Social media, the mobile web and augmented revolution. *Future Internet*, 4, 83–91. doi: 10.3390/fi4010083

Kazman, R., & Chen, H. -M. (2009). The metropolis model: A new logic for development of crowdsourced systems. *Communications of the ACM*, 52(7), 76–84. doi: 10.1145/1538788.1538808

Kelty, C. (2008). *Two bits: The cultural significance of free software*. Durham, NC: Duke University Press.

Kirschenbaum, M. G. (2008). *Mechanisms: New media and the forensic imagination*. Cambridge, MA: The MIT Press.

Konieczny, P. (2009). Wikipedia: Community or social movement? *Interface*, 1(2), 212–232. Retrieved from http://interfacejournal.nuim.ie/wordpress/wp-content/uploads/2010/11/Interface-1-2-pp212-232-Konieczny.pdf

Lampe, C., & Resnick, P. (2004). Slash(dot) and burn: Distributed moderation in a large online conversation space. In *Proceedings of CHI 2004* (pp. 543–550). New York: ACM Press.

Lanzara, G. F., & Patriotta, G. (2007). The institutionalization of knowledge in an automotive factory: Templates, inscriptions, and the problem of durability. *Organization Studies*, 28(5), 635–660. doi: 10.1177/0170840607068309

Latour, B. (1992). Where are the missing masses? The sociology of a few mundane artifacts. In J. Law & W. Bijker (Eds.), *Shaping technology/building society: Studies in sociotechnical change* (pp. 225–258). Cambridge, MA: The MIT Press.

Latour, B., & Woolgar, S. (1986). *Laboratory life: The social construction of scientific facts*. Princeton, NJ: Princeton University Press.

Lessig, L. (1999). *Code and other laws of cyberspace*. New York, NY: Basic Books.

Lessig, L. (2006). *Code 2.0*. New York, NY: Basic Books.

Lindtner, S., & Li, D. (2012). Created in China: The makings of China's hackerspace community. *Interactions*, 19(6), 18–22. doi: 10.1145/2377783.2377789

Lundby, K. (2011). Patterns of belonging in online/offline interfaces of religion. *Information, Communication & Society*, 14(8), 1219–1235. doi: 10.1080/1369118X.2011.594077

Mackenzie, A. (2005). The performativity of code: Software and cultures of circulation. *Theory, Culture & Society*, 22(1), 71–92. doi: 10.1177/0263276405048436

MacKenzie, D. (1996). Marx and the machine. In *Knowing machines: Essays on technical change* (pp. 23–48). Cambridge, MA: The MIT Press.

Mackenzie, D. (2006). *An engine, not a camera: How financial models shape markets*. Cambridge, MA: The MIT Press.

Marx, K. (1973). The fragment on machines. In *The grundrisse* (pp. 690–712). London: Penguin.

Mathew, A., & Cheshire, C. (2010). The new cartographers: Trust and social order within the Internet infrastructure. In *Proceedings of the 38th research conference on Communication, Information and Internet Policy*. Retrieved from http://papers.ssrn.com/abstract=1988216

Mockus, A., Fielding, R. T., & Herbsleb, J. D. (2002). Two case studies of open source software development: Apache and Mozilla. *ACM Transactions on Software Engineering and Methodology*, 11(3), 309–346. doi: 10.1145/567793.567795

Mol, A. (2002). *The body multiple: Ontology in medical practice*. Durham, NC: Duke University Press.

Müller-Birn, C., Dobusch, L., & Herbsleb, J. D. (2013). Work-to-rule: The emergence of algorithmic governance in Wikipedia. In *Proceedings of Communities and Technologies 2013* (pp. 80–89). New York: ACM Press.

Niederer, S., & Van Dijck, J. (2010). Wisdom of the crowd or technicity of content? Wikipedia as a sociotechnical system. *New Media & Society*, 12(8), 1368–1387. doi: 10.1177/1461444810365297

O'Leary, M. B., & Cummings, J. N. (2007). The spatial, temporal, and configurational characteristics of geographic dispersion in teams. *MIS Quarterly*, 31(3), 433–452.

Orlikowski, W. J., & Scott, S. V. (2008). Sociomateriality: Challenging the separation of technology, work and organization. *The Academy of Management Annals*, 2(1), 433–474. doi: 10.1080/19416520802211644 ,

Phillips, L. B. (2013). The temple and the bazaar: Wikipedia as a platform for open authority in museums. *Curator: The Museum Journal*, 56(2), 219–235. doi: 10.1111/cura.12021

Poor, N. (2005). Mechanisms of an online public sphere: The website slashdot. *Journal of Computer-Mediated Communication*, 10(2).

Reagle, J. (2010). *Good faith collaboration: The culture of Wikipedia*. Cambridge, MA: The MIT Press.

Ribes, D., Jackson, S., Geiger, R. S., Burton, M., & Finholt, T. (2013). Artifacts that organize: Delegation in the distributed organization. *Information and Organization*, 23(1), 1–14. doi: 10.1016/j.infoandorg.2012.08.001

Saxenian, A. (2006). *The new argonauts: Regional advantage in a global economy* (p. 424). Cambridge, MA: Harvard University Press.

Schindler, M., & Vrandecic, D. (2009). Introducing new features to Wikipedia – Case studies for web science. In *Proceedings of web science 2009*. Retrieved from http://journal.webscience.org/213/

Sengers, P. (1998). Do the thing right. In *Proceedings of the second international conference on Autonomous agents – AGENTS '98* (pp. 24–31). New York: ACM Press.

Sengers, P. (2000). *Narrative intelligence. Human cognition and social agent technology*. Philadelphia, PA: John Benjamins Publishing.

Shapin, S., & Schaffer, S. (1985). *Leviathan and the air-pump*. Princeton, NJ: Princeton University Press.

Stallman, R. (2013, September). Why free software is more important now than ever before. *Wired*. Retrieved from http://www.wired.com/opinion/2013/09/why-free-software-is-more-important-now-than-ever-before/

Star, S. L. (1999). The ethnography of infrastructure. *American Behavioral Scientist*, 43(3), 377–391. doi: 10.1177/00027649921955326

Star, S. L., & Griesemer, J. R. (1989). Institutional ecology, "translations" and boundary objects: Amateurs and professionals in Berkeley's museum of vertebrate zoology. *Social Studies of Science*, 19(3), 387–420. doi: 10.1177/030631289019003001 , ,

Star, S. L., & Ruhleder, K. (1996). Steps toward an ecology of infrastructure: Design and access for large information spaces. *Information Systems Research*, 7(1), 111–134. doi: 10.1287/isre.7.1.111 ,

Surowiecki, J. (2004). *The wisdom of crowds: Why the many are smarter than the few and how collective wisdom shapes business, economies, societies and nations*. New York, NY: Doubleday.

Takhteyev, Y. (2012). *Coding places: Software practice in a South American city*. Cambridge, MA: The MIT Press.

Thorson, K., Driscoll, K., Ekdale, B., Edgerly, S., Thompson, L. G., Schrock, A., Wells, C. (2013). YouTube, Twitter, and the occupy movement. *Information, Communication & Society*, 16(3), 421–451. doi: 10.1080/1369118X.2012.756051

Tkacz. (2013). Open sesame. *Aeon*. Retrieved from http://www.aeonmagazine.com/world-views/nathaniel-tkacz-open-source-government

Van Dijck, J., & Nieborg, D. (2009). Wikinomics and its discontents: A critical analysis of Web 2.0 business manifestos. *New Media & Society*, 11(5), 855–874. doi: 10.1177/1461444809105356

Wattenberg, M., Viegas, F., & McKeon, M. (2007). The hidden order of Wikipedia. In *Proceedings of OCSC 2007* (pp. 445–454).

Wilson, B. (2005). Rave and straightedge, the virtual and the real: Exploring online and offline experiences in Canadian youth subcultures. *Youth & Society*, 36(3), 276–311. doi: 10.1177/0044118X03260498

Winner, L. (1986). Do artifacts have politics? In *The whale and the reactor: A search for limits in an age of high technology*. (pp. 19–39). Chicago, IL: University of Chicago Press.

Wong, J., & Hong, J. (2008). What do we "mashup" when we make mashups? In *Proceedings of the 4th international workshop on end-user software engineering – WEUSE '08* (pp. 35–39). New York: ACM Press.