Digital Geography 3 (2022) 100035

Contents lists available at ScienceDirect

Digital Geography and Society

journal homepage: www.sciencedirect.com/journal/digital-geography-and-society

Everyday code: The project for democracy on our 'desktop'

Mark Purcell

Department of Urban Design & Planning, University of Washington, Box 355740, Gould 410, Seattle, WA 98195, USA

A R T I C L E I N F O

Keywords:
Democracy
Self-management
Software
Linux
Graphical user interfaces

A B S T R A C T

This paper examines what I call the project for democracy, by which I mean a perpetual individual-and-collective project to manage our affairs ourselves, in all areas of our lives. The goal of the paper is to understand better what that project entails and how we can carry it out well. To do so, I examine a very prosaic empirical case, the software code that organizes the everyday environment of our personal computing devices: operating systems, window managers, wireless interfaces, system trays, and so on. What would the project of democracy look like in that context? The bulk of the paper is spent fleshing out an answer to that question. I then suggest that the project for democracy in the digital realm of the desktop is just one instance of the wider project for democracy. The desktop can be a little model that can guide and inspire the project for democracy in other arenas, such as the household, the neighborhood, the city, and beyond.

1. Introduction

This paper's goal is to better understand the project for democracy so that we are able to carry out that project more effectively. I understand democracy differently from its conventional meaning. For me democracy is something much more radical and thoroughgoing, something much closer to the word's original meaning. Democracy as I understand it is a collective political-and-social project in which people strive to increasingly manage their affairs themselves in all areas of their lives (Purcell, 2013). That way of understanding democracy prompts some very pragmatic questions. What would such a project entail, in actual practice? What would it demand of us, if we decide to commit to it and want to carry it out well? This paper goes some way in addressing those questions. It investigates what democracy would mean in practice by examining a very mundane context: the software that runs the digital "desktops" on our personal computing devices. How might we more fully understand, create, and manage that software ourselves? I suggest that we should read the digital desktop as a little model, as an illustrative example of what it would be like to take up the project for democracy in other contexts. Those other contexts are both digital – like big data, data privacy, and digital sovereignty – and more traditionally geographical – like the household, the neighborhood, and the city. My hope is that the paper will help us understand that although the project for democracy is not easy, it is nevertheless entirely possible, and it is, in the main, a remarkably joyous experience. In investigating the digital desktop in this way, as an exploration of our own potential to manage our affairs ourselves, the paper is relatively unusual in the digital geographies literature, where much more attention is paid to digital structures of oppression, domination, and resistance. The paper thus makes a plea for, and tries to model, work that trains its attention instead on our potential to thrive in the digital realm.

2. What is democracy?

Let me say a bit more about how I understand democracy. When we use the word "democracy" today, we usually mean what is in fact *liberal-democratic government*, an arrangement in which people surrender their power to a small subset of elected officials, and those officials make decisions about how to use the limited powers of government.[1] But that is not what I mean by democracy. Democracy as I conceive it is often called "radical" or "direct" democracy, and what that means, at its root, is that in democracy people do *not* surrender their power to an outside entity. Instead, they retain their power and use it to manage their affairs for themselves.

My understanding of democracy is indebted to a small but vibrant body of radical political thought that emerged in the post-war years of the 20th Century. This work includes Castoriadis' (1997 esp. Chapters 8, 10, and 11) idea of "autonomy," Fanon's call for decentralized self-management and reappropriation in the colonial context (Fanon, 1961, see esp. 94, 104, 121-4, 141-3), Lefebvre's (2009) and Vaneigem's

E-mail address: mpurcell@uw.edu.

[1] To give just one example, *The Economist* (2019) recently asked: "Are Western democracies becoming ungovernable?" The article is about liberal-democratic States in Europe, like Spain, France, and Italy.

https://doi.org/10.1016/j.diggeo.2022.100035
Received 21 February 2021; Received in revised form 25 February 2022; Accepted 6 April 2022
Available online 12 April 2022

(1974) concept of "*autogestion*," and the Italian idea of "*autonomia*" (e.g. Agamben, 1993; Negri, 1999; Virno, 2004). More recently, a similar way of understanding democracy has been developed by Rancière (1999) and Abensour (2011).[2] Taken as a whole, this body of thought shares a broad political and ethical commitment to the idea and practice of popular self-management. That is literally what the French word "*autogestion*" means. Originally, the term emerged in heterodox Marxist theory, in the context of the workers' struggle, as a name for workers managing the production process themselves, rather than having it managed for them by capitalists. Subsequently, the work of Castoriadis, Fanon, Lefebvre, Vaneigem, and the Italian autonomists was keen to *extend* this more narrow understanding beyond the workers' movement and the factory to explore how self-management could be practiced in geographical realms like the neighborhood (e.g. Lotta Continua, 1973), the village (e.g. Fanon, 1961), and the city (Lefebvre, 1968).

As *autogestion* was extended to realms beyond the factory, the term *autogestion généralisée* was used to signify the more general struggle for direct control by people themselves. This struggle was against all forms of centralized, bureaucratic, and hierarchical control. Such forms included capitalist corporations, to be sure, but they also included political parties (and State institutions more generally), and workers' unions. When Lefebvre advocated *autogestion* in the context of the city,[3] or the Italian autonomists did so in the context of the neighborhood, they conceived of it as a struggle by people themselves – the users of space – to wrest control of the production of urban space away from State authorities who, as the providers of collective consumption goods, were so prominent in that arena. When Fanon advocated that villagers in Algeria manage rural land themselves, he opposed that self-management to *all* forms of centralized control, whether by a French colonial administration, a traditional village chief, or the nationalist government of the postcolonial State.[4] These are just a few of many possible examples. All forms of bureaucratic, centralized control were seen by these thinkers as usurping the power people should have to directly manage their affairs themselves.

Of course this all can seem quite daunting, to understand democracy like this. It is easy to take the argument I have been making so far and rush to imagine democracy as a form of utopian community in which people fully control every aspect of their lives. But we need to avoid such utopianism. We should not understand "democracy" to mean the perfectly democratic community. Instead, we should understand democracy as a *project*. Democracy is a long-term, patient, and intentional project to retain our power and use it to manage our affairs, to the extent we are able, in whatever arenas we can. We should not expect this project to achieve a perfect, final community called "democracy." Nor should we expect democracy to arrive overnight by means of a revolution. Instead, democracy is a perpetual project that we take up, that we begin today and continue into the foreseeable future. As we engage ourselves in this project, as we practice democracy, as we invest our energy, attention, care, and productive power in the work of managing our affairs for ourselves, we will, little by little, grow stronger. We will become better able to retain and use our power. As we engage in this democratic practice, and as we invent, collectively, our new new democratic lives in common, we will grow more accustomed to self-management, more habituated to the practice, more comfortable with its routines and techniques. We will become, in short, steadily more democratic. That is the project of democracy.

[2] And there are others besides who understand democracy radically, such as Laclau and Mouffe (1985), Badiou (2010), Nancy (2010), Rosanvallon (2006). For fine-grained theoretical reasons that are beyond the scope of the paper, that work is less integral to my way of thinking democracy here.

[3] His well-known "right to the city".

[4] "The important point," says fanon, 1961 134-135, "is not that three hundred people understand and decide, but that all understand and decide."

3. The desktop as a digital geography

For some time now I have been arguing that critical geography more broadly is focusing its attention in the wrong way. It is oriented very much toward structures of power and domination, and it thus is turned away from the activity, strength, and creativity of those who are building alternative lives in common (Purcell, 2016). Sarah Elwood (2021) has recently argued that this broader problem is true of the work in digital geographies as well. She shows (2021, 209–10) how the digital geographies scholarship has "developed rich repertoires for theorizing digital mediations of capture, dispossession, and adverse incorporation," but it has done far less to "apprehend the sites, significance and workings of digital practices of thriving" (see also Elwood & Leszczynski, 2018; Lynch, 2020a). The literature is replete with analyses and critiques of domination. We see numerous critiques, for example, of the digital divide (e.g. Graham, Hale, & Stephens, 2012), or analyses of how algorithms control behavior (e.g. Graham, Zook, & Boulton, 2013; Thrift & French, 2002), or work exposing how smart city discourses normalize domination (Datta & Odendaal, 2019), or studies of the way digital systems reinscribe oppression based on race (e.g. Jefferson, 2017; Noble, 2018), and gender (e.g. Elwood & Leszczynski, 2018; Stephens, 2013), and sexuality (Gieseking, 2017), and class (Thatcher, O'Sullivan, & Mahmoudi, 2016). Even the work on "digital democracy," which is increasingly prominent in the literature (Kinsley, McLean, & Maalsen, 2020, 2), often takes a debunking role, showing how its promise is far greater than its product (e.g. Gastil & Davies, 2020; Haklay, 2013; Hindman, 2009; Powell, 2012).

One might assume that what Elwood and I are hoping for instead is more studies of resistance, more work that shows how people are opposing and disrupting digital domination. This would seem to shift the focus toward the activity of people. Such studies are indeed very much present in the digital geographies literature. Swanlund and Schuurman (2018) document tactics for resisting geosurveillance. McLean (2020) details feminist digital activism that challenges sexism and misogyny. Franklin (2014) examines the "dynamics of power and resistance" around control of the internet. Wolfe (2021) reports on the Russian government's attempts to control resistance in the blogosphere. But such studies, I argue, do not really address the problem. Although they do reorient our attention onto the activity of people rather than the structures that oppress them, such work nevertheless remains trapped in an obsession with structures of domination. It remains fixated on the relation of domination and control, and so the only popular activity it can imagine are attempts to resist, cancel, negate, and destroy those relations. They cannot see, and therefore cannot make us aware of, our own power to create alternative lives in common. They are "unprepared," in Elwood's phrasing, "to apprehend…digital practices of thriving" (2021,210).

To be clear, neither Elwood nor I are saying that this work on domination and resistance is a waste of time. The point is only that we do *too much* of this work. It consumes our attention and effort, and so we have, as a result, a dearth of work on digital practices of thriving. What is needed, therefore, is more work on what alternatives are being created and how. And there is some such work, even if it is relatively more rare in the literature. Elwood points to work on alternative digital infrastructures (Slager, 2018), rogue epistemologies that elude structures of racial violence (McKittrick, 2016), and the way a glitch politics can open possibilities for life for trans people (Cardenas, 2016). In a similar vein, Gerhardt (2020) explores the possibility of creating a post-capitalist digital creative commons. McLean (2020, 99–102) offers a brief account of indigenous Australians using digital storytelling as a way to survive, heal, and even thrive. Young (2019) relates how a rural indigenous community in Canada is using digital technology to build alternative economic practices. And perhaps most closely resonant with the concept of democracy I advocate are Casey Lynch's case studies of self-managed digital initiatives in Spain (e.g. Lynch, 2020b).

What is needed, then, is more work that explores our own capacity to produce ways of life that are conducive to digital thriving. This article speaks to that need. To be sure, it acknowledges the very real limitations imposed by dominant structures over our digital desktops. As I describe below, the forces of capitalist accumulation have indeed worked to alienate us from our desktops in important ways. However, the article's analysis continually returns to our own activity, to our own capacity to learn, decide, and use the everyday code on our desktops for ourselves. This activity is what matters most for democracy: do we decide to take charge of our own affairs, or do we give up access and control, and let our project for democracy wither?

4. On the "desktop": A case study

Before I dive in to those questions, let me specify just what I mean by the digital information on our desktops. This case study focuses on the software code – window managers, system trays, login managers, system monitors, power managers, application launchers, file managers, wifi managers, and so on – that structures the user interface of the operating systems on our personal computing devices. By the latter, I mean the range of devices like desktop computers, laptops, tablets, smartphones, smart watches, and so on. The "personal" modifier is used primarily to exclude servers, mainframes, and supercomputers, which is not to say understanding the software there (especially on servers) is any less important. I use the word desktop to indicate the virtual "desktop" that these user interfaces create for us. They are not really spaces, or places. Those terms could only be applied figuratively. Desktops are perhaps closest to what geographers would call an environment or a milieu: a structuring context that provides specific limitations and opportunities for the user of the device.

I want to acknowledge that this empirical focus is very quotidian. But at the same time, I argue that it matters. For most of us, the desktop is the taken-for-granted medium in which we do our work, access news about the world, connect with our friends, buy goods, navigate our surroundings, and consume our entertainment. It is, increasingly, an essential interface with the material world that makes our daily lives possible. We have increasingly become, as Donna Haraway (1985) put it, a "fabricated hybrid of machine and organism." This cyborg self can be both constraining and liberating, as she says, but in any case there can be little doubt that machines are integral to our bodies and our selves. The software that structures the interfaces of those machines is thus a critical medium of our well-being, even if we rarely think about it (Fraser, 2019).

For most of us, when we are interacting with our desktop, we do so by means of a Graphical User Interface, or GUI. A GUI is a program that allows a user to issue commands to a computer without knowing the actual commands themselves. It opens a window on the screen and presents the user with various graphical *representations* of commands—buttons, icons, drop-down lists, check boxes, and tabs—with which the user can, through a series of gestures and clicks with a mouse or a finger, indicate what changes s/he wants to make. We use GUIs for all sorts of mundane administrative tasks, like connecting to a wifi network, or changing the look of windows, or the behavior of the mouse/touchpad, or the monitor resolution, and so on. Each of these is a trivial thing, but when taken together, they produce our digital environment. We grow very accustomed to how things work in that environment. If those settings were changed, our ability to use the tool we rely on to work and to interact with the world would be greatly disrupted.

The surreptitious danger of GUIs, from the perspective of democracy, is that they allow us to use a computer without needing to learn how it works. We don't have to know the commands that need to be issued, or know how to read and write the language of its configuration files, because the GUI gives us a picture-book version. It enables us to use the computer, and, at the same time, it sets up a screen that obscures the code that operates the computer. GUIs make it possible for us to remain passive, to allow others to manage our desktops for us.

Clearly there is a lot to unpack and defend in those claims. Let me start to do so by way of analogy. Across his work, David Foster Wallace, following in the tradition of Ray Bradbury, worried that the entertainment we consume so eagerly is rendering us inactive as an audience. It delivers gratification to us, he said, without demanding any real activity on our part. We don't have to be literate (it's TV), we don't have to get any allusions, we don't have to grapple with a narrative that will trouble us in any way. This inactivity, carried over years of consuming such entertainment, increasingly renders us passive, helpless, unable to think and act for ourselves. Wallace was concerned that we are becoming more and more like an infant, an image he returns to frequently in his writing. We are being returned to the crib, where we are wrapped in a warm blanket, cared for, satisfied, and soothed. This concern reaches its apex in *Infinite Jest* (Wallace, 1996). In that book there is a film called "the entertainment" that effectively makes the viewer feel as though she is back in the crib, being tended to by her mother. It is such a compelling, all-encompassing feeling that viewers literally can't stop watching. They remain fixed in front of the screen in a catatonic state. If they are not cared for by others, they die of dehydration. My argument is that the GUI infantilizes computer users in an analogous way. That is a bold claim, so let me support it some with a little history.

4.1. Hold my hand: A brief history of the GUI[5]

In 1983, when desktop microcomputers were a very new thing, *Byte* magazine published a collection of articles that gave an account of the current state of the relatively new technology. In the 1970s, most computers were *mini*computers, like Digital Equipment Corporation's PDP-10, that cost tens (and sometimes hundreds) of thousands of dollars and were almost exclusively used by highly trained programmers and engineers at commercial research labs, universities, and government agencies (Bell, 2014). By 1983, however, the growing availability of the much smaller desktop *micro*computer was making it feasible for people to buy their own personal machine and use it at home. There was, as a result, a fast-growing number of non-expert computer users in the world. The articles in *Byte* were exploring the question of which operating-system software was most likely to corner this new microcomputer market. CP/M, MS-DOS, and Unix were the main contenders. One article argued forcefully that in order for an operating system to succeed in this market, it needed to understand that there are "two kinds of users: developers and the end users. End users have an entirely different set of needs. They should be isolated from the esoterica of the computer and be given something that is easy to use" (Krieger & Pack, 1983, 209). Because of all the new inexpert users coming into the market, the article goes on, operating systems and software applications now need to do much more "hand-holding" than they did in the past, when users were expected to enter direct commands at a prompt. Instead, the authors argue, the user should be given menus that lay out the available options. And the article also saw, even in 1983, another option on the horizon. Apple's "Lisa" operating system, it said,

> will have a profound effect on the front-ends of all future operating systems. A Lisa user sees a graphics display with pictorial representations instead of words. He operates the system by manipulating a mouse, which moves the cursor among the pictures (or "icons"), each of which signifies a command or the data to be handled. In addition, extensive use is made of windowing, which enables the user to see

[5] This history is drawn from a variety of primary and secondary archival sources, which are cited in the text as they are used.

several displays at once. Mice and windows are proliferating[6] (Krieger & Pack, 1983, 210).

It is clear that for *Byte*'s readers in 1983, the idea of a GUI-driven computer environment, with "pictorial representations instead of words," was entirely new and strange. At the time, it was normal for a computer to present its user with only a command prompt. The user was expected to issue direct commands to make the computer do what s/he wanted it to do, an interface called the "command line" (Stephenson, 1999). Radio Shack's TRS-80 Model 4, for example, which debuted in that same year, presented the user with a blank screen that just said:

```
READY
  >_
```

It offered only a keyboard for the user to operate it. This command-line interface was normal then. It would utterly confound the average user today. S/he would assume something had gone terribly wrong with the system.

When the *Byte* article came out, graphical interfaces between computer and user had been developing behind the scenes for decades. Probably the most important catalyst to the rise of the GUI on personal computers was the work carried out at Xerox's Palo Alto Research Center (PARC) in the 1970s, where technologies like the mouse, hypertext, what-you-see-is-what-you-get word processing, object-oriented programming, and Ethernet were developed (Yesterday's Computer of Tomorrow: The XEROX Alto, 2017). PARC produced the Xerox Alto, which was the first microcomputer to offer a primarily graphical interface: windows, icons, point-and-click selection of objects, and cut-and-paste text editing, all situated in a metaphorical "desktop" environment.

Xerox never packaged these technologies into a microcomputer that sold successfully on the mass market. But in 1979 Xerox invested a small amount of capital in Apple, and part of the terms of that deal gave Apple some access to the research at PARC. Apple CEO Steve Jobs took advantage of that access eagerly, and Apple subsequently used PARC's ideas liberally in the rapid development of their GUIs (Kay, 2017; The Apple Connection, 2017). While the Apple II family of computers of the late 1970s and early 1980s had only minimal graphical interfaces, the interfaces on the Apple Lisa (1983) and Macintosh (1984) were almost entirely graphical. Jobs understood very well that GUIs opened up an enormous new market for microcomputers because they made computers usable for the mass of inexperts. While the Lisa cost about $10,000 and did not sell well, it quickly gave way to the Macintosh, which cost about $2500 and became the first GUI-driven microcomputer that sold on a large scale (Williams, 1984). Apple's business model was precisely to hold the hands of the "end users" we saw in the *Byte* article, so that Apple could sell those users computers that were "easy to use." The Macintosh presented a graphical interface that enabled computer illiterates to use it. Apple never hid this fact. Just the opposite. In the marketing for the Macintosh, it celebrated itself as a small hive of geniuses who had graciously allowed the bereft public to access the utterly inscrutable world of computers. The copy from one of their advertisements of that era is worth citing at length:

> In the olden days, before 1984, not very many people used computers. For a very good reason. Not very many people knew how. And not very many people wanted to learn. After all, in those [dark] days it meant…falling asleep over computer manuals. And staying awake nights to memorize commands so complicated you'd have to be a computer to understand them. Then, on a particularly bright day in Cupertino, California, some particularly bright engineers had a particularly bright idea: since computers are so smart, wouldn't it make more sense to teach computers about people, instead of teaching people about computers?…And when the engineers were finally finished, they introduced us to a personal computer…so easy to use, most people already know how. They didn't call in the QZ190, or the Zipchip 5000. They called it Macintosh™ (see Fig. 1).

Apple is happy to tell us what is going on here.[7] In fact they gladly proclaim it in short, easy to understand sentences: computers are boring and unimaginably complex, and most people are not "bright" enough to understand them. So there is absolutely no point in "teaching people about computers." People are too dumb. What people need is some "very bright" engineers who will do the thinking for them, who will make computers less inscrutable and easier to use.[8] The ad implies, unsubtly, that the Macintosh has pulled us out of the Dark Ages and into the Enlightenment. However, even if this shift made it possible for there to be more computer users, those users were almost exactly as computer illiterate as they were before. The new Macintosh users did not understand how a computer works; they depended wholly on the help of "bright engineers." The real change that this purported Enlightenment brought was to transform computer-illiterate non-customers of Apple into computer-illiterate customers of Apple. That was the whole point.

The Macintosh and its GUIs sold successfully. At the time, Microsoft controlled the market for command-line operating systems on minicomputers with its MS-DOS software. But it was clear that Apple's new system was a success, and so Microsoft reacted by developing its GUI-filled "Windows" software, which ran on top of MS-DOS. By the end of the 1980s, GUIs had become the primary user interface on microcomputers. Of course today they are entirely normal. We now expect an interface that has "pictorial representations instead of words." We are so steeped in GUIs that, for us, the distinction between icon and command has collapsed. Note how the *Byte* article explains that the "icons" are *representations* of commands.[9] We have mostly lost any awareness of that distinction. We think the pictures *actually are* the commands and that using a mouse to click on an icon at the bottom of the screen is how you launch a program. Most of us probably don't even know it is possible to issue direct commands to the computer.

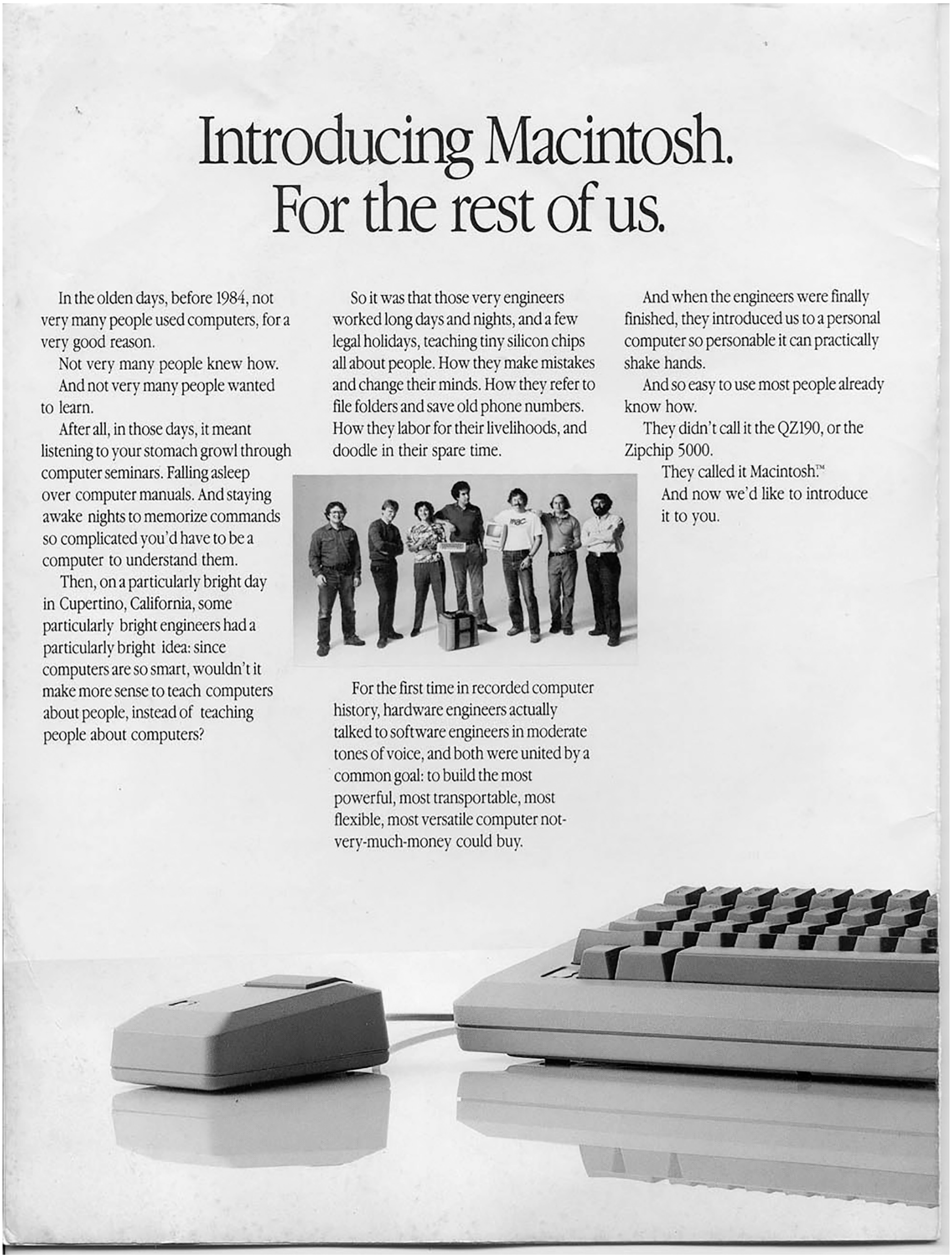
Given that the proliferation of the GUI was driven by Apple's corporate need to expand their market, we should be clear-eyed in understanding that there is indeed a force here that is beyond the control of users, and that force, to an extent, could be said to have *imposed* GUIs on users. But that is not really the whole story. Users also *participated* in the process. They participated eagerly. They pointed and clicked and abandoned (or skipped over) the command line with great enthusiasm. Users were told to their face that they were too dumb to use computers, that they needed bright engineers to draw pictures for them. And they agreed. They bought, at $2500 a unit, what Apple was selling. They let the geniuses do the thinking for them. Even as Steve Jobs was doing more than anyone in history to ensure widespread and deep-seated computer illiteracy, our culture energetically celebrated him as a hero of our time. It seems clear that we users are not just helpless victims of the GUIfication of the desktop. We are active participants.

[6] Confronted with this line, I can't help mentioning that in the GNU/Linux ecosystem there is a window manager called "Ratpoison" that allows the user to do everything with the keyboard and never touch a mouse at all (http://www.nongnu.org/ratpoison).

[7] Well, except for the fact that most of the "bright engineers" who had these "bright ideas" worked at PARC, not Apple.

[8] Apple continues to push this fundamental relation with respect to information today. Apple stores have a "genius bar" where non-genius users are expected to take their machine when they have a problem. Users are actively discouraged, in a variety of ways, from fixing it themselves.

[9] This is all so very, very similar to the original "icons": images painted in Christian churches to convey religious meaning to the mass of illiterate believers.



Introducing Macintosh.
For the rest of us.

In the olden days, before 1984, not very many people used computers, for a very good reason.

Not very many people knew how.

And not very many people wanted to learn.

After all, in those days, it meant listening to your stomach growl through computer seminars. Falling asleep over computer manuals. And staying awake nights to memorize commands so complicated you'd have to be a computer to understand them.

Then, on a particularly bright day in Cupertino, California, some particularly bright engineers had a particularly bright idea: since computers are so smart, wouldn't it make more sense to teach computers about people, instead of teaching people about computers?

So it was that those very engineers worked long days and nights, and a few legal holidays, teaching tiny silicon chips all about people. How they make mistakes and change their minds. How they refer to file folders and save old phone numbers. How they labor for their livelihoods, and doodle in their spare time.

For the first time in recorded computer history, hardware engineers actually talked to software engineers in moderate tones of voice, and both were united by a common goal: to build the most powerful, most transportable, most flexible, most versatile computer not-very-much-money could buy.

And when the engineers were finally finished, they introduced us to a personal computer so personable it can practically shake hands.

And so easy to use most people already know how.

They didn't call it the QZ190, or the Zipchip 5000.

They called it Macintosh.™

And now we'd like to introduce it to you.

Fig. 1. Apple Advertisement.

4.2. GUIs in GNU/Linux

I want to explore further this issue of active participation by users in GUIfication by turning to a different software ecosystem where it is more readily apparent: the multitude of GNU/Linux desktop operating systems. Whereas Apple and Microsoft operating systems are largely closed-source, proprietary software sold for a profit by very large corporations, GNU/Linux systems are free and open-source software distributed by a whole host of different entities. GNU/Linux systems combine 1) a software kernel called the Linux kernel, and 2) a suite of software tools produced by the GNU project, the leading force in the free and open-source software movement (see gnu.org). Both Linux and GNU are licensed under the GNU General Public License, the strongest possible guarantee that the code will remain free and open. Neither Linux nor GNU is a for-profit corporation, and so in this ecosystem there is much less motivation to increase revenue by expanding a customer base.[10] Here the motive of corporate profit is not the main force driving the production, distribution, and use of the GUI. And yet, the GUI is very much there.

GNU began developing its software tools, all of which are intended primarily to be used on the command-line, in the mid-1980s (gnu.org/gnu/gnu.html). The Linux kernel was first released in 1991. By 1994 the first full-fledged GNU/Linux operating systems, which are called "distributions," or usually just "distros," were being released. Distros like Debian (debian.org), Slackware (slackware.com), and Red Hat were the pioneers. Although GNU/Linux has come to dominate the server, mainframe, supercomputer, smartphone and tablet markets,[11] it has not grown to rival Windows, or even MacOS, in the market for desktop operating systems. The primary obstacle for GNU/Linux in the latter market is not price, since virtually all GNU/Linux distros are available at no charge. The main difficulty is that Windows and MacOS users do not choose their operating system. It just comes with the computer they bought, already installed. In order to use a GNU/Linux distro, most users have to actively choose to *switch* to GNU/Linux from the system that came with their machine.

GNU/Linux distros very much want users to switch to their distro, and so most are concerned to be welcoming and easy to use. They want their distro to be easy to install, to "work out of the box," to look polished and attractive, and to present a graphical environment that is similar enough to Windows or MacOS that the new user will not feel disoriented. Most distros agree that all this requires GUIs, and lots of them, to hold users' hands. And so, beginning in the 1990s there were several major GNU/Linux efforts to develop sophisticated graphical desktop environments that could rival MacOS and Windows (de Icaza, 1997). The KDE project and the Gnome project were both released in the late 1990s. Since then, there has been a proliferation of graphical desktop environments (about 10 major ones) and window managers (probably over 50) in GNU/Linux, most of which are designed to ensure that GNU/Linux looks good and is easy to use. For example, consider how one distro describes the Gnome desktop environment:

> GNOME is a user-friendly desktop environment that enables users to easily use and configure their computers. GNOME includes a panel (for starting applications and displaying status), a desktop (where data and applications can be placed), a set of standard desktop tools and applications, and a set of conventions that make it easy for applications to cooperate and be consistent with each other. Users of other operating systems or environments should feel right at home using the powerful graphics-driven environment that GNOME provides.[12]

The proliferation of GUIs in GNU/Linux is near-universal. Most distros assume that software used to install the system must present a GUI to guide the user through installation. Further, they assume that once the distro is installed and the machine has booted, a GUI must take the user through login. And then a whole suite of GUIs (a power manager, a wifi manager, a sound manager, a look-and-feel manager, an application launcher, a system menu, a status bar, etc.), must present the user with a seamless graphical environment that is visually appealing. S/he must feel oriented, at home, and even *pleased* in the new environment. S/he must be able to do what s/he needs to do with minimal inconvenience. The overarching assumption, widely shared as common sense among most GNU/Linux distros, is that if this graphical experience does not hold users' hands effectively enough, they will abandon GNU/Linux and return to Windows or MacOS.

However, even if that assumption is widely shared, it is not universal. Some distros, like Slackware, Gentoo (gentoo.org), and Arch Linux (archlinux.org), assume, instead, that users are capable, and that they want to manage their own system. Slackware and Gentoo, for example, expect users to download the raw code for their software and compile that software themselves, on their desktop. Arch eschews a graphical installer and instead expects users to install their system using command-line tools, which, while harder to use, give users much more control over the installation. The lack of a GUI installer can be pretty nerve-wracking, because the installation process requires users to make really important decisions, like how the hard drive will be partitioned, what boot loader is installed and where, and what file system will be used. If these questions are not handled properly, the whole system will be inoperable. In the same vein, some distros don't provide a login manager. They expect users to login and choose their graphical desktop environment on the command line. Some distros don't even come with a graphical desktop environment at all. They let users use the command line to install and configure one, if they choose to. These distros assume users are willing – and even eager – to work, study, learn, and develop their computer literacy. They assume that users don't need or want their hands to be held, that they want to decide for themselves how their operating system will work (e.g. www.gentoo.org/get-started/philosophy). Although none of these distros is the most widely used distro, each has a significant and enthusiastic user base. None is in danger of fading into irrelevance any time soon.

4.3. The infant inside me[13]

So far I have been discussing the relationship between users and the GUI in the GNU/Linux ecosystem at the most general level. But the relationship operates at all levels, and it runs all the way down to the level of the individual or body. So this section explores the relation at that more fine-grained level by offering a brief autoethnographic example of my relation with GUIs.

In this little example, I am running a GNU/Linux operating system called MX Linux (mxlinux.org), I am on a laptop computer, and I am using a desktop environment produced by a software project called Xfce (xfce.org). In this system, the monitor resolution is changed by issuing the following command to the computer:

[10] To be sure, that motive is present here as well: Red Hat (recently bought by IBM: www.redhat.com/en/about/company) and Canonical (canonical.com/) are two corporations that distribute GNU/Linux distros. They do not sell the software the way Microsoft does, but their corporate revenues are still very much linked to the size of their user base.

[11] Google's Android operating system, which dominates both the smartphone and tablet markets, runs a Linux kernel.

[12] This quotation is from the DragonflyBSD handbook, Chapter 11 (dragonflybsd.org/docs/handbook). Dragonfly is not in fact a GNU/Linux distro but a Unix distro descended from the Berkeley Software Distribution, which is also free and open-source.

[13] In this section my methods turn from archival to autoethnographic (Butz & Besio, 2009; Fraser, 2019). The data is drawn from ten years of experience using Linux as my desktop operating system.

```
xrandr –output LVDS1 –mode 1366 × 768
```

'xrandr' is the program that issues the command, the "–output" flag tells the computer which monitor to adjust, and the "–mode" flag tells the computer which resolution to set that monitor to (in this case: 1366 pixels horizontally by 768 pixels vertically). I can make these changes on the command line, by just typing the command above into a terminal and pressing enter. But if I don't know that command, I can't issue it directly to the computer. I need to use a GUI. The GUI on a system running Xfce is a program called "xfce4-display-settings." To use that program, I can open a terminal and type "xfce4-display-settings," but if I don't know how to do that, I need to use a GUI to open the GUI. I use a mouse to move the arrow cursor to the "Menu" button in the top-left corner of the desktop. I click on that button, which shows me a base menu of options (e.g. "Internet," "Office," "Settings," "System," etc.). Clicking "Settings" on that menu opens a sub-menu, on which I can click "Display." This click, at last, issues the command "xfce4-display-settings" to the computer, and the GUI for monitors is launched. Xfce4-display-settings draws a new window on the screen. But I don't know the names of the monitors that are available to adjust, and I don't know how to ask what those names are. So the GUI makes a query to discover that information for me. The real name of my laptop's monitor is LVDS1, but the GUI doesn't report that. It just presents me with an icon that looks like a monitor, and it labels this icon "Laptop". I click on that icon. But I don't know what resolutions that monitor is able to display, so the GUI makes another query to discover that information for me, and then it presents me a drop-down box that lists the available resolutions. I use my mouse to click on what seems to be the biggest resolution, 1366 × 768, and then I click the "Apply" button. It is at this point, at last, that the GUI issues the "xrandr" command we started with above. After a brief flicker of the screen, the resolution changes. At this point, GUI checks back in with me, using a dialog box, to ask if I am happy with the new monitor resolution. I answer the GUI by using the mouse to click on the "yes" button or the "no" button.

Nearly all of us use a GUI to change our monitor resolution. We rely on it, because we don't know very much about how our monitor works, or what commands will make the changes we want to make. We don't even know how to launch the GUI program itself, and need a graphical menu to help us do that. The GUI enables our ignorance. It uses pictures and simple words to ask us questions, and then it carries out all the queries and commands in the background, screened from our view. We are probably not even aware queries and commands are being issued. The options just appear; the monitor just changes. The GUI takes care of it.

Broken down into such detail, I am sure this example seems painfully trivial. It is not as though democracy will die in darkness if we don't know how to change our monitor settings. What I am trying to show is the fine-grained contours of our alienation from the digital information on our desktop. It is hidden from us. We do not know how to access and use it. GUIs make it unnecessary. Although in isolation each instance like this is trivial, when they are taken together – the display resolution, the keyboard map, the mouse's behavior, and so on – they matter. They provide the interface that allows our device to work in the way we are used to.

What I think makes my case worth investigating in detail is that my alienation is entirely voluntary. There is no dark force is preventing me from knowing how. The "xfce4-display-settings" GUI is not produced by a large corporation trying to increase its market share. Xfce gives its software away for free. They provide the GUI because they assume most people who run Xfce don't know xrandr, and so if a GUI is not provided, users will not want to use Xfce. Xfce doesn't *prevent* me from learning xrandr, it just benignly assumes that I won't, and it offers me a GUI as an easy way out. And I take it. I let the GUI take care of me.

And I do this even though the command-line tool, xrandr, is readily available to me. It is open-source, free of charge, and comes pre-installed on most GNU/Linux distros. Its manual is included. The manual is only 2100 words long, and it is comprehensive. I can mostly learn xrandr in about an hour. It is a far more direct, efficient, and powerful tool. It is entirely possible for me to use it, if I decide to. I just need to start: read the xrandr manual, issue commands, and see what happens. When those commands work, I will be encouraged, and I will push on to try out other features of the command. When they fail, when I break something (and I will), I will need to figure out how to fix it. Or, more likely, I can turn to other users who have broken the same thing, and they can share their experience. Once I learn xrandr, I will know everything about monitors that I didn't know in the paragraphs above, and I will be able to make the queries and issue the commands I could not before. The Apple advertisement we saw above is wrong. Xrandr isn't "so complex only a computer could understand it." I can understand it. Once I do, I won't need the GUI anymore. What is more, and I think this is critical, as I build my strength in this way, by studying, learning, and practicing, I will very likely find I have a taste for it. I will come to *enjoy* the feeling of learning a command, issuing it directly to the computer, and seeing the changes happen. I will come to *prefer* that way of interacting with machines to the alienation of the GUI. This feeling—call it pleasure, or joy, or delight—is vital. It will have to be there if we are going to succeed. It isn't a cheap pleasure, but a deeper one, slower burning but longer lasting. It is a pleasure we can settle into, that we can make a habit out of. It is the pleasure of the project of democracy (Purcell, 2013).

4.4. An individual-and-collective project

We have been trained, over the last 40 years, to think of the desktop as an individualized milieu, each of us operating on our own *personal* computer.[14] This training can tempt us to think of the project of democracy on the desktop as an individual project. I alone must make a commitment to eschew GUIs and manage the information on my desktop myself. I am the one who will have to make countless decisions to remain active and in control. No one will be standing over me, checking if I am doing my work. I will have to rely on my own will, my own strength. Moreover, the benefits of this project mostly accrue to me. When I learn to read and write the code that runs my desktop, it is a project to improve myself, to increase my own strength, my own ability to act. Much of the joy will be felt by me, as I manage my machine myself.

However, the minute I begin the project of learning how my system works, I see immediately how inadequate this individualized understanding is. I quickly realize I am just one element in a very large and vibrant collective mass, a community of people who have been producing and managing software together for a long time. To return again to my xrandr case, when I open the manual, I immediately connect to the program's authors, who also wrote the manual, and they help me understand the tool more fully. If I have difficulty following the explanations in the manual (which is pretty common), I can connect with other xrandr users and ask *them* questions. There is an active Internet Relay Chat channel for most applications in Linux, and there are countless online forums where other users are going through a very similar learning experience. All this information is archived and freely shared with others. As I begin to ask questions, others teach me how to ask them well, and I also get better at interpreting the responses. In time, I come to connect with a rich store of common wisdom, even on a topic as

[14] As the history above suggests, this personalized quality was not inevitable. It had to be actively produced by firms wanting to market the *personal* computer. The architecture of Unix, which was developed in the 1970s, was designed for one large central computer that many users shared, using "dumb terminals." It was only starting in 1980 or so, with the market successes of MS-DOS running on IBM's *Personal* Computer, and Apple's desktop computers, that we came to see computers as personal possessions. This shift reached its nadir in Apple's "i" line of products (iMacs, iOS, iPhone, etc.) that aim to make one's computer into a container for, and a statement about, one's individual personality.

insignificant as xrandr. My specific questions will almost always be answered, my mastery will grow, and my connection to this common knowledge will be deepened.

So we must understand that the project of producing and managing desktop information ourselves is a project that is both individual and collective. It does indeed involve something that resembles the "autonomous individual" of the liberal Enlightenment imagination who *dis*connects from others and diminishes her dependence on them in order to be autonomous, in order to manage her affairs her*self*. This individual grows stronger and more independent—*grows up*, we might say—when she does things for herself and on her own, rather than when she relies on others. But that individual would be dashed against the rocks of frustration and fatigue in a matter of days if she tried, alone, to take command of even the smallest corner of her desktop. If we are to really thrive in this project, we must undertake it in association with others. It must be a collective project. In the case of the desktop, this means that for a user is to be successful in managing information herself, she must connect with other users. She must develop her own technical knowledge by tapping into the collective knowledge of her peers. And those peers must freely share what they know. She can only grow stronger and more self-reliant if she makes these connections to others.

At this point one might raise the objection that the project I am describing is really just trading our dependence on GUI for a dependence on a community of fellow users and their common knowledge. My response is that I think in the former condition—most people's current condition—is truly a relation of dependence, because the user is passive, disconnected, illiterate, and wholly reliant on the help of the GUI. The project of democracy, by contrast, calls for users to become active, and part of that activation involves engaging with a community of similarly active users. We must still decide where to look and who to trust. We must still evaluate the advice we receive, and decide what to use and what to discard in the project of assembling a growing expertise in relation to a store of common knowledge. I would contend that the former condition deserves the name "dependence," but the latter condition is more properly called something like "collective engagement," or maybe just "democracy."

5. Conclusion

I have tried to give some account of what this individual-and-collective project of democracy, this project to invest our energy in the practice of managing our affairs ourselves, would mean in the very specific and mundane context of the digital "desktop." But of course democracy in the digital realm is not limited to the question of the software on our personal computing devices. There are countless other digital contexts in which democracy can be pursued. I offer my exploration of the desktop in the hope it can be a kind of model, or analogy, for pursuing democracy in other digital contexts. More specifically, I hope my case provides the digital geographies literature with a rich example of what it would mean to train our attention on our own capacity to produce ways of life that are conducive to our digital thriving. I hope I have inspired others to undertake similar explorations that can, taken together, augment that relatively under-examined aspect of digital geographies.

I want to end the paper by acknowledging that non-digital geographical realms are also critical to the project of democracy, and to suggest how we might extend the project of democracy in those areas. For example, geographers often think in terms of the household, the neighborhood, and the city. As I mentioned above, the last two were a central concern of the radical-democratic literature on autonomy and *autogestion* that so inspires my thinking on democracy. Our project to manage information on the desktop ourselves is analogous to our project to manage the neighborhood and the city ourselves, almost to the point of the two projects being isomorphic. In managing the information on our desktops, we users decide to take up the project of producing and managing this vital realm ourselves. The whole point of Lefebvre's "right to the city," for example, is the same: we who inhabit the city – the "users" of space, he called us (Lefebvre, 1974) – can take up the project of actively producing and managing urban space ourselves. Whether our particular urban focus is affordable housing, or access to transportation, or segregation, or education, or policing reform, or environmental hazards, the project of democracy in the city involves urban inhabitants searching for new ways to become active and take up the project of managing and producing urban space themselves, in whatever way they are able in the context they inhabit.

And of course the project to manage our desktops and the project to manage urban space are only two of the many projects that matter. When Lefebvre turned his attention toward the city and the urban inhabitant, and then to space more generally, he was trying to generalize the concept of *autogestion*, beyond the factory and beyond the working class, to the city and to the urban inhabitant. There is no reason to stop there. Whether it be the rural villages that Fanon focused on, or the household, the neighborhood, the school district, the police, the housing market, the banking system, the farm, the climate, the desktop: all are arenas in which we can take up the project of democracy. I think we should think of these all as essentially equivalent political projects. There is no reason to nest or hierarchize them. A project for democracy on the desktop is no more or less important than a project for democracy in the village, or the household, or the city, or the school. Each further develops our ability to manage our affairs ourselves. Each teaches us the habits, skills, and attitudes we'll need to maintain our project. Each trains us to know what it is like to appropriate a sphere of experience, to take up the challenge of becoming active. Each time we do, we become more aware of our own power to create, to manage, and to decide. Each little project helps us know what it feels like: the pleasure, or joy, or delight, of democracy. Each is a little attempt—always both individual and collective—to save our lives. What we need to do, I would argue, is not to rank them or prioritize them. We need to notice them, amass them, connect them together into a spreading project for generalized *autogestion*, into a spreading project for democracy.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Abensour, M. (2011). *Democracy against the state: Marx and the machiavellian moment. Translated by Max Blechman and Martin Breaugh*. Malden, MA: Polity.
Agamben, G. (1993). *The coming community. Translated by M. Hardt*. Minneapolis: University of Minnesota Press.
Badiou, A. (2010). *The communist hypothesis*. New York: Verso.
Bell, G. (2014). STARS: Rise and fall of minicomputers. *Proceedings of the IEEE, 102*(4), 629–638.
Butz, D., & Besio, K. (2009). Autoethnography. *Geography Compass, 3*(5), 1660–1674.
Cardenas, M. (2016). Trans of color poetics. *The Scholar and Feminist Online, 13*(3–14), 1.
Castoriadis, C. (1997). *The Castoriadis reader. Edited by David Curtis*. Oxford; Cambridge, Mass: Wiley-Blackwell.
Datta, A., & Odendaal, N. (2019). Smart cities and the banality of power. *Society and Space, 37*(3), 387–392.
Elwood, S. (2021). Digital geographies, feminist relationality, black and queer code studies: Thriving otherwise. *Progress in Human Geography, 45*(2), 209–228.
Elwood, S., & Leszczynski, A. (2018). Feminist digital geographies. *Gender, Place and Culture, 25*(5), 629–644.
Fanon, F. (1961). 2004. The wretched of the earth. In *Translated by Richard Philcox*. New York: Grove Press.
Franklin, M. I. (2014). *Digital dilemmas: Power, resistance, and the internet*. Oxford: Oxford University Press.
Fraser, A. (2019). Curating digital geographies in an era of data colonialism. *Geoforum, 104*, 193–200.
Gastil, J., & Davies, T. (2020). Digital democracy: Episode IV—A new hope. *Digital Government: Research and Practice, 1*(1).
Gerhardt, H. (2020). Engaging the non-flat world. *Antipode, 52*(3), 681–701.
Gieseking, J. J. (2017). Messing with the attractiveness algorithm: A response to queering code/space. *Gender, Place and Culture, 24*(11), 1659–1665. https://doi.org/10.1080/0966369X.2017.1379955

Graham, M., Hale, S., & Stephens, M. (2012). Featured graphic: Digital divide: The geography of internet access. *Environment and Planning A, 44*(5), 1009–1010.

Graham, M., Zook, M., & Boulton, A. (2013). Augmented reality in urban places. *Transactions of the Institute of British Geographers, 38*, 464–479.

Haklay, M. (2013). Neogeography and the delusion of democratisation. *Envirnment and Planning A, 45*, 55–69.

Haraway, D. (1985). A cyborg manifesto: Science, technology, and socialist feminism in the 1980s. *Socialist Review, 80*, 65–108.

Hindman, M. (2009). *The myth of digital democracy*. Princeton University Press.

de Icaza, M. (1997). *The GNOME Desktop Project*. Gtk-List. https://mail.gnome.org/archives/gtk-list/1997-August/msg00123.html.

Jefferson, B. J. (2017). Digitize and punish. *Society and Space, 35*(5), 775–796.

Kay, A. (2017). *What Was It like to Be at Xerox PARC When Steve Jobs Visited?* Quora, 2017, June 13 edition https://www.quora.com/What-was-it-like-to-be-at-Xerox-PARC-when-Steve-Jobs-visited.

Kinsley, S., McLean, J., & Maalsen, S. (2020). Editorial. *Digital Geography and Society, 1*(1), 1–3.

Krieger, M., & Pack, F. (1983). Unix as an application environment. *Byte, 8*(10), 209–218.

Laclau, E., & Mouffe, C. (1985). *Hegemony and socialist strategy: Towards a radical democratic politics*. London: Verso.

Lefebvre, H. (1968). *Le Droit à La Ville*. Paris: Anthropos.

Lefebvre, H. (1974). 1991. The production of space. In *Translated by Donald Nicholson-Smith*. Cambridge, MA: Blackwell.

Lefebvre, H. (2009). State, space, world: Selected essays. In N. Brenner, & S. Elden (Eds.), *Translated by Gerald Moore, Neil Brenner, and Stuart Elden*. Minneapolis: University of Minnesota Press.

Lotta Continua. (1973). Take over the City: Community Struggle in Italy. *Radical America, 7*(2).

Lynch, C. (2020a). Contesting digital futures. *Antipode, 52*(3), 660–680.

Lynch, C. (2020b). Unruly digital subjects. *Digital Geography and Society, 1*(1), 1–8.

McKittrick, K. (2016). Diachronic loops/deadweight tonnage/ bad made measure. *Cultural Geographies, 23*(1), 3–18.

McLean, J. (2020). *Changing Digital Geographies*. Springer Nature.

Nancy, J.-L. (2010). *The truth of democracy*. Translated by Pascale-Anne Brault and Michael Naas: Fordam University Press.

Negri, A. (1999). *Insurgencies: Constituent power and the modern state. Translated by M. Boscagli*. Minneapolis: Univeristy of Minnesota Press.

Noble, S. (2018). *Algorithms of oppression*. New York: NYU Press.

Powell, A. (2012). Democratizing production through open source knowledge. *Media, Culture and Society, 34*(6), 691–708.

Purcell, M. (2013). *The Down-deep delight of democracy*. Chichester: Wiley-Blackwell.

Purcell, M. (2016). Our New Arms. In Simon Springer, Kean Birch, & Julie MacLeavy (Eds.), *The Handbook of Neoliberalism* (pp. 613–622). New York: Routledge.

Rancière, J. (1999). *Disagreement: Politics and philosophy*. Translated by J. Rose. Minneapolis: University of Minnesota Press.

Rosanvallon, P. (2006). *Democracy past and future*. New York: Columbia University Press.

Slager, E. (2018). *Infrastructures of survival*. Dissertation, Seattle: University of Washington.

Stephens, M. (2013). Gender and the GeoWeb. *GeoJournal, 78*(6), 981–996.

Stephenson, N. (1999). *In the beginning there was the command line*. New York: Harper Collins.

Swanlund, D., & Schuurman, N. (2018). Resisting Geosurveillance. *Progress in Human Geography, 43*(4), 596–610.

Thatcher, J., O'Sullivan, D., & Mahmoudi, D. (2016). Data colonialism through accumulation by dispossession. *Envirnment and Planning D: Society and Space, 34*(6), 990–1006.

The Apple Connection. (2017). *In: Revolution, the first 2000 years of computing*. Mountain View, CA: Computer History Museum.

The Economist. (2019). *Are western democracies becoming ungovernable?*, 2019, August 1 edition.

Thrift, N., & French, S. (2002). The automatic production of space. *Transactions of the Institute of British Geographers, 27*, 309–335.

Vaneigem, R. (1974). *De La Grève Sauvage à l'autogestion Généralisée*. Paris: Éditions 10/18.

Virno, P. (2004). *A grammar of the multitude*. Los Angeles: Semiotext(e).

Wallace, D. (1996). *Infinite Jest*. New York: Little, Brown, and Co.

Williams, G. (1984). The apple Macintosh computer. *Byte, 9*(2), 30–57.

Wolfe, S. (2021). Blogging the virtual: New geographies of domination and resistance in and beyond Russia. *Antipode, 53*(4), 1251–1269.

Yesterday''s Computer of Tomorrow: The XEROX Alto. (2017). *In: Revolution, the first 2000 years of computing*. Mountain View, CA: Computer History Museum. https://www.computerhistory.org/collections/catalog/102738587.

Young, J. (2019). Rural digital geographies and new landscapes of social resilience. *Journal of Rural Studies, 70*, 66–74.