Ethical Programs

Brown, James J.

Published by University of Michigan Press

Brown, James J.
Ethical Programs: Hospitality and the Rhetorics of Software.

For additional information about this book
https://muse.jhu.edu/book/42619

For content related to this chapter
https://muse.jhu.edu/related_content?type=book&id=1667089
Introduction: The Swarm

Networked life forces us to interact with others, even when we haven’t extended an invitation and even when we haven’t been invited. Life in a network society—one in which information and bodies constantly move and collide—means never getting to be alone and never getting to be offline. It means never really getting to decide in any thoroughgoing way who or what enters your “home” (your apartment, your laptop, your iPhone, your thermostat). This situation is one defined by hospitality, but not a hospitality that involves a clearly defined host or guest. We certainly extend invitations all the time: an invite to a shared folder in the cloud or a Facebook friend request. But this is but one type of what I call an ethical program, a procedure enacted in the face of networked arrivals. As Wendy Chun demonstrates in *Control and Freedom*, these programs are often put forth on our behalf, without our knowledge: “the moment you ‘jack in’ . . . your Ethernet card participates in an incessant ‘dialogue’ with other networked machines.”

Chun shows us that packet sniffing technology reveals how often “your computer constantly wanders without you.” Our computational machines are constantly engaging in conversations, extending and accepting invitations, deciding who or what gets to enter or not. While these programmed decisions may turn away some packets or people, they never fully shut off the network connection.

We like to think that we have some control over how our digital abodes function, that we know who we are inviting or shunning. But Facebook privacy functions shift and Dropbox usernames are hacked, meaning that the concept of the invitation is mostly a pleasant fiction. The other arrives, over and beyond our choices to filter or turn away. This is not just a problem confined to digital networks. The line between online and offline, never very clear to begin with, is now difficult to trace at all. In the mid-1990s, the screech of a modem served as an aural marker. Dial tone, touch-tone dialing, screech, crackle, connect: “Okay, now I’m online” (at least until my dumb brother picks up the phone and breaks the connection). Today, such a procedure for
“going online,” a phrase that we still hear but that seems quaint and dated, is foreign. Mobile devices and game consoles are online as soon as they’re powered up. In fact, initial plans for the Xbox One game console called for the unit to be always online, and Microsoft initially designed it so that key features would be disabled if an Internet connection was not reestablished every 24 hours. After a great deal of backlash, this design was scrapped, though the Xbox Kinect camera has a default setting of “always on” that must be disabled by users. These blurry lines seemed slightly less blurry in the tone-dial-screech-crackle-connect scenario, but even the bad old days of dial-up Internet access put forth the fiction that there was an offline and an online, a moment when others were arriving and a moment when we were alone. Modems could always be hacked, and one could even use “sneakernet” (the slang term for “physically moving removable media”) to arrive at a computer terminal uninvited.

This book suggests not that networked computing creates the predicament of hospitality but rather that it takes up this very old problem—the problem of others arriving whether we invited them or not—over and over again. Existence, even prior to what we now call the “network society,” always meant taking up the question of the other. In 1967, life was perhaps not quite as explicitly networked as today, but that year brought the publication of Emmanuel Levinas’s essay “Substitution,” which seems to theorize the very set of relations I have begun tracing here. For Levinas, the proximity of one to another suggests a presymbolic relation, one that happens prior to any attempt to thematize the other. When one is approached by another, this comes as a relationship with a singularity, without the mediation of any principle or ideality. In the concrete, it describes my relationship with the neighbor, a relationship whose signifyingness is prior to the celebrated “sense bestowing.”

Prior to any attempt to make sense of our guest, that guest has already arrived, forcing a relation whether or not we have requested it. For Levinas, this means existents are exposed to one another and held hostage to one another, not knowing where a visitor is coming from. Avital Ronell demonstrates how this logic is programmed directly into the beginnings of networked life, by way of the telephone. When I answer my phone, I have already said “yes”:

And yet, you’re saying yes, almost automatically, suddenly, sometimes irreversibly. Your picking it up means the call has come through. It means more: you’re its beneficiary, rising to meet its demand, to pay a debt. You don’t know who is calling or what you are going to be called
upon to do, and still, you are lending your ear, giving something up, receiving an order. It is a question of answerability. Who answers the call of the telephone, the call of duty, and accounts for the taxes it appears to impose?  

Again, the ring of the phone offers a demarcation point that seems to slip away when we begin to consider WiFi connections, but Ronell’s analysis demonstrates that the call precedes any particular technological arrangement. Any encounter comes after an initial “Yes,” which marks an exposedness to an other. Ronell’s analysis of the telephone is the precursor to my own argument. Ethical Programs extends her question about answerability and ethics from circuit switching to packet switching, from Bell to Berners-Lee.

As Levinas argues, when I am put in relation with another, I am radically open and exposed: “Concretely, this means to be accused of what others do and to be responsible for what others do.” This predicament, which is entirely unavoidable, refigures ethics as something beyond individual choice. Yes, I make decisions about who or what might enter my home, who or what has access to my Twitter feed, but this only happens in the face of an ever-present exposedness to others. Prior to the large-scale availability of networked computational devices, Levinas described the relation that defines networked life. Levinas likens this predicament to that of the hostage, and he suggests that if “we” as humans share anything at all, it is this experience of being held hostage by another that resists representation, that arrives over and beyond our attempts to make sense of that other. Who can deny that this phenomenological description of existence maps directly onto networked life, which is utterly defined by the arrival of others? From social networks to online gaming communities, I am forever exposed to such arrivals. I might put in place filters, blocking the troll or unfriending a former high school classmate, but each of these filters happens only after I have entered a space of exposedness and response-ability, a space of hospitality.

Political theorist Carl Schmitt describes this problem in a very different register and with somewhat different terms. For Schmitt, the question of the other is a political one (one can address the question ethically, but Schmitt would insist that the political is a singular realm in which collectives make decisions), and the arrival of another engages the political decision par excellence: “The specific political distinction to which political actions can be reduced is that between friend and enemy” (870). For Schmitt, any political entity is defined by this decision about friends and enemies. The enemy need not be a villain, but that enemy is different from “us” and is identifiably other. The state, for Schmitt, is defined by a collective’s willingness to die to defend itself against this other:
The political enemy need not be morally evil or aesthetically ugly; he need not appear as an economic competitor, and it may even be advantageous to engage with him in business transactions. But he is, nevertheless, the other, the stranger; and it is sufficient for his nature that he is, in a specially intense way, existentially something different and alien so that in the extreme case conflicts with him are possible. These can neither be decided by a previously determined general norm nor by the judgment of a disinterested and therefore neutral third party.  

Whereas Levinas’s concern is with a primordial ethical relation, one that happens over and beyond any ethical decision, Schmitt’s primary concern is with the distinction between friend and enemy, a distinction that he views as the launching point for the political.  

My own discussion of software and networked life is concerned with both of these sets of questions, both Levinas’s concern for an alterity that arrives without mediation and Schmitt’s concern for deciding who is “friend” and who is “foe.” Ethical Programs takes up the procedures enacted—these may be computational procedures, but they do not have to be—when others arrive. In The Exploit, Alexander Galloway and Eugene Thacker pose this question in the terms laid out by Levinas and Schmitt, asking how ethics might operate in networked spaces that are defined by the constant arrival of a “faceless foe”:

A swarm attacks from all directions, and intermittently but consistently—it has no “front,” no battle line, no central point of vulnerability. It is dispersed, distributed, and yet in constant communication. In short, it is a faceless foe, or a foe stripped of “faciality” as such. So a new problematic emerges. If the Schmittian notion of enmity (friend—foe) presupposes a more fundamental relation of what Levinas refers to as “facing” the other, and if this is, for Levinas, a key element to thinking the ethical relation, what sort of ethics is possible when the other has no “face” and yet is construed as other (as friend or foe)? What is the shape of the ethical encounter when one “faces” the swarm?

Galloway and Thacker push us to ask a number of difficult questions: How does one think an ethics of the network, which thrusts us into a space that welcomes the swarm? What does ethical decision look like in networked life, and what rhetorical actions are possible? Ethical Programs takes up these questions by examining software in networked spaces. How does software navigate between the unconditional welcome granted by a network connection, an invitation extended to a faceless foe, and the measured, conditional gestures
that inevitably emerge in response, the gestures that begin to determine who or what is friend and foe?

The other will have to be dealt with—even if that other is shunned. The question of ethics, of how the other is to be dealt with, sifted, sorted, welcomed, turned away, is not only a question of human decision but also one of machines. In this book, these ethical problems are addressed by way of software and by examining how computational machines move between two poles: an unconditional hospitality that defines ethics and relationality as such and a conditional hospitality that is cultivated in our attempts to deal with “the swarm.” Studying how software moves between these two poles—the conditional and unconditional—is crucial as we design and live within our networked dwellings.

I use the term ethical program to evoke both the computational procedures of software (a computer program) and the procedures we develop in order to deal with ethical predicaments (a program of action). An ethical program, computational or otherwise, is a set of steps taken to address an ethical predicament. These steps are not necessarily arrived at rationally, and they are not always the result of deliberation. In fact, we often enact ethical programs in moments when we do not have the luxury of considering all possible options. Francisco Varela describes how we “immediately cope” in moments of ethical decision, but he argues that this coping is always coupled with ethical deliberation. For Varela, ethical decisions happen in moments of breakdown when we are no longer experts of what he calls our “microworld,” a lived situation in which we develop a microidentity, a “readiness for action.” Varela’s focus on lived situations and situated ethical actions is an attempt to theorize ethics without reducing it to abstractions. He is interested in both “immediate coping” and “deliberation and analysis” as cognitive modes: “It is at the moments of breakdown, that is, when we are not experts of our microworld anymore, that we deliberate and analyze, that we become like beginners seeking to feel at ease with the task at hand.” In the terms laid out in this book, Varela is describing the ethical programs we might enact as we address the predicaments of hospitality.

Ethical programs are enacted constantly, by both humans and computational machines, and software studies presents a set of terms and concepts for making sense of those programs. Like Lev Manovich, my aim is to apply those terms and concepts to the software that is often considered more tool than expressive artifact. In Software Takes Command, Manovich carries out analyses of media editing software in order to understand how that software has actively shaped design concepts. Manovich aims to unearth the seemingly more mundane computational machines of our digital media ecologies, extending
the work of those focusing on electronic literature, critical code studies, and platform studies. Along these same lines, Ethical Programs focuses on how tools such as MediaWiki and Twitter enact ethical programs and express arguments about how best to contend with hospitality. Such work is crucial if we are to understand how computation is shaping networked life. For instance, let’s consider what happens when I try to log in to my university’s NetID system, and I’ve forgotten my password. Three failed log-in attempts results in the software locking me out of the system. This is the system coping, enacting an ethical program in the face of a breakdown in its microworld. If the software does not receive the correct credential information, it determines that someone is trying to hack the account by guessing passwords, and it locks everything down. This is a coping mechanism, one that is enacted in the interest of the safety and security of both the user and the system. This particular program may require me to make a phone call, talking to a network administrator to get the account unlocked. That person will have to enact her own ethical programs to determine identity, asking for a pin number or my mother’s maiden name. Whether a gatekeeping mechanism is a computational machine or a human, and whether that mechanism is enacted as a way of immediately coping (locking me out of the system) or analyzing (verifying identity by asking security questions), an ethical program has been triggered.

In networked life, ethical programs enact rules, procedures, and heuristics about how (or whether) interactions should happen. Blogs and other websites employ systems to filter comments and to allow users to promote or demote contributions from other users. These sites often lay out detailed policies about what can or cannot be posted and how inappropriate material, from spam to trolling, will be dealt with in digital spaces. Some sites allow users to flag material, and others have human conversation moderators. These are all ways of dealing with the predicaments of a hospitable network that welcomes writing and writers from all angles. The ethical programs I focus on in this book come in the form of software platforms that shape, enable, and constrain networked life. Software on the network cannot avoid questions of ethics and hospitality, and this is because the network is based upon the assumption that others will arrive. Even firewall software that takes the shutting out of the other as its basic strategy fits this definition of an ethical program since it must address the question of the other over and over again. Readers, writers, and programmers in the network are continually confronted with the swarm, which incessantly invites faceless others. While they may never arrive at the answer to this ethical predicament, ethical programs continually address such questions. We might say that they iterate through solutions, testing out possibilities.

As I explain in more detail in chapter 1, the network serves as a constant
reminder of what Jacques Derrida calls the Law of hospitality. This Law defines life in the network in that others are welcomed, regardless of identity or credentials. Networked technology would not exist without the Law of hospitality, since connectivity is necessary for such technology to function. On the other hand, this Law of hospitality is perverted and undermined at every turn as we filter, sift, and sort arrivals. These filters are the laws of hospitality, which we enact in response to the Law. I use the term ethical programs to describe our efforts to write the laws of hospitality, which are always in tension with the Law. Neither the laws nor the Law exhaust one another—they require one another. The Law provokes laws, and laws can never remain absolutely faithful to the Law. While we could examine many types of ethical programs enacted to navigate networked life—programs that cut across all kinds of human-machine assemblages—I have chosen to examine software since it stands as a particularly useful example of our attempts to author contingent responses to the universal and unending difficulties of hospitality. Software is an interesting place to trace out ethical programs since it enacts rules and procedures, shaping and constraining what can or can’t happen in a given space. Just as commenting procedures lay out the rules of engagement in an online community like Reddit, this same community employs computational procedures that limit how often new users can post comments. In both cases, an ethical program enforces rules, creating (or preventing) certain kinds of relations between users and systems.

The ethical decisions coded into software must continually address the problem of hospitality. This means that ethical programs are iterative, contingent responses to the Law of hospitality. As a scholar of rhetoric and writing, I am particularly interested in how the ethical programs I examine in this book confront specific situations and exigencies. Software enacts ethical programs, but such programs are only ever temporary solutions, and software can be rewritten to enact different kinds of ethical programs in different situations. Software does not simply describe procedures, it enacts them, meaning that it shapes networked life in fundamental ways. Understanding how software’s ethical programs are written and rewritten and how they engage the Law of hospitality is central to understanding, in Galloway and Thacker’s words, “the shape of the ethical encounter when one ‘faces’ the swarm.” But each of these ethical programs is rhetorical. It makes an argument, marshals persuasive resources, and addresses the particulars of a situation. The university server in the lost password example above examines the situation and determines that the best course of action is to lock me out. This action implies an argument about the best way to keep the network (and my account) secure, and that argument enacts an ethical program that responds to a contingency. It makes assumptions about the shape of the ethical encounter, and it helps
to determine how relations can or cannot happen in a given space. While the
danger of any ethical program is that it becomes codified and calcified, that it
becomes too focused on immediate ethical concerns at the expense of broader
questions, networked life means that the Law of hospitality will continue to
present itself. Life in the network means never being able to turn off the other
(a problem that, as I've already argued, is not even actually new), and ethical
programs have to be continually reinterpreted, rewritten, and reimagined to
deal with this unending call.

In the interest of understanding how ethical programs continually engage
the difficulties of hospitality in networked life, this book is made up of a num-
ber of stories about how software helps determine the shapes of our ethical
encounters. My focus on particular stories means that this book takes on a
“case study” tone. I address the value and limits of this approach in more de-
tail in the conclusion, but for now I'll say that my theoretical frame for the
study of software in networked environments is rhetorical theory, which is
concerned with tracing the particulars of a given attempt to persuade. In each
chapter, this focus on particulars is coupled with broader arguments about
how that software takes part in complex activities, encouraging certain kinds
of action and constraining others, welcoming some writers and bits of infor-
mation while shunning or filtering others. I examine the software itself, how
it is used (or abused), and how it is part of controversies and complications.
This focus on particular moments allows me to look carefully at a series of
rhetorical situations, teasing out how the complexities of hospitality are be-
ing negotiated and determining what this situation can tell us about software,
rhetoric, and ethics in networked life.

As a rhetorician, I am drawn to this type of analysis, situating rhetorical
action in terms of strategies, exigencies, purposes, and audiences. However,
my hope is that these case studies are also able to move beyond the particular,
enabling us to zoom out and discuss some more far-reaching problems. This
balancing act between the particular and the general mirrors Derrida’s con-
cerns with the Law of hospitality and the laws of hospitality. In fact, it mirrors
the tensions inherent in any ethical program: How does one move between
specific ethical predicaments that call for immediate action and the universal
principles that are inevitably undercut by that action? Every universal prin-
ciple, taken to its logical end point, results in unethical activities. This is most
clearly demonstrated by the biblical story of Lot, a troubling tale about how
hospitality by no means implies kindness. When Lot (one of Derrida’s more
instructive examples of hospitality) turns over his daughters to be raped by a
mob rather than surrendering his houseguests, he is abiding by the Law of
hospitality, the insistence that the master of the house protect his visitors. The
mob demands that Lot hand over his guests, but the host’s ethical program
forbids it, and he offers his daughters instead. Lot enacts his procedure based on the Law of hospitality, a rigid and unwavering precept stating that a host is responsible for his guests. Immanuel Kant’s famous insistence, in the *Groundwork for the Metaphysic of Morals*, that lying or deception is wrong, regardless of circumstance, operates by way of a similar logic. On its own, the Law of hospitality is often far from what we might consider ethical—this is why we enact laws.

To use an example closer to the subject matter of this book, consider RFC 761, the document that establishes the Transmission Control Protocol (TCP) and determines how packets of information move through the Internet. RFC 761 lays out the philosophy of information transfer in terms of the Robustness Principle: “TCP implementations should follow a general principle of robustness: be conservative in what you do, be liberal in what you accept from others.” This principle (also known as Postel’s Law since it is attributed to Jon Postel, one of the authors of the TCP documentation), insists that when transmitting information, a server should conform to the rules and protocols for transmission as closely as possible, but that this same server should be flexible when determining whether incoming packets conform to that same protocol. This design principle is meant to make it easier for systems to speak with one another. Eric Allman puts it this way:

> If every implementation of some service that generates some piece of protocol did so using the most conservative interpretation of the specification and every implementation that accepted that piece of protocol interpreted it using the most generous interpretation, then the chance that the two services would be able to talk with each other would be maximized.

While the Robustness Principle means greater interoperability, taken to its logical end we would have an Internet that is entirely inoperable. If all systems were to be “liberal in what [they] expect from others,” then the protocols that determine how packets of information move would quickly become irrelevant. Put differently, at some point we must both follow the rules and ask others to do the same.

Allman makes such an argument in his call for a “middle way” when applying the Robustness Principle. He argues that Postel’s Law has come under fire and has often been ignored. Allman’s explanation for why this happens hints at the predicament of hospitality in networked life: “This isn’t because implementers have gotten more stupid, but rather because the world has become more hostile.” That hostility can be linked to nefarious activity on the part of hackers, but it can also be explained in terms of the increasing complexity of network activity: “The Robustness Principle was formulated in an
Internet of cooperators. The world has changed a lot since then. Everything, even services that you may think you control, is suspect.” That is, the hostility described here isn’t necessarily about bad people but is instead about a distributed group of technologies and people that may or may not be cooperating toward some shared goal. Allman suggests that such a complex network requires a rethinking of the Robustness Principle:

The atmosphere of the Internet has changed so much that the Robustness Principle has to be severely reinterpreted. Being liberal in what you accept can contribute to security problems. Sometimes interoperability and security are at odds with each other. In today’s climate they are both essential. Some balance must be drawn.

What Allman is advocating for here is a revisiting of the laws of hospitality in the face of the Law of hospitality. The Law will always invite packets of information, in whatever form, whether they follow procedure or not. The laws of hospitality must draw lines and sort through what should or should not be allowed to pass. Computational machines, as ethical programs, must continually navigate this tension. While broad ethical principles are imperfect, without them we would have no way of comparing specific actions in particular rhetorical situations to those enacted in other situations. This, of course, is not a problem unique to new media or to software. However, my discussion of how software shapes, enables, and constrains rhetorical action continually oscillates between these two poles, between the Law and the laws.

This discussion of the Law and the laws is traced out in more detail in chapter 1, as I explain the terms and concepts that guide my analysis. I provide a detailed explanation of Derrida’s theory of hospitality, showing that he saw the problem of hospitality as one that was exposed, in a particularly radical way, by networked technologies. From wiretapping to e-mail, Derrida’s work attempts to understand what happens when we confront the fiction that “the home” is somehow sealed off from the outside. Connected to networks, the home (which can stand in for our house but can just as easily be used to describe the smartphone) is defined by an exposedness to others. Networked life reminds us, over and over again, that there is no home without connections to the outside, no house without windows and doors. This is the Law of hospitality that defines networked life, and it demands that we author ethical programs that take up the questions of the other’s arrival. Software is one way to author such ethical programs, and it allows us to enact rules that help shape our rhetorical dwellings.

Engaging the hospitable network means developing contingent ethical programs that somehow leave open the possibility that we have gone astray,
allowing for the possibility that we might decide to revise our program. As I have already suggested, ethical programs will always have to negotiate between the immediate difficulties of a particular situation (the arrival of some specific other) and the ethical demand of the Law of hospitality. If this Law is forgotten or if an ethical program attempts to solve the problem of the other in any final way, we are confronted with both ethical problems and questions of technological feasibility. The extermination of the other is not only ethically wrong; it is also a technological impossibility. Unplugging from the network— which would seem to be the only way to answer the Law of hospitality in any complete or final way—not only makes networked technology useless but also fools us into thinking that networked life is confined to our digital interactions. From financial transactions to surveillance technology to the magnetic strip on a subway ticket, we have daily reminders that it is impossible to fully disconnect from the network. This means we will always require ways of engaging the difficulties and predicaments of the hospitable network, which welcomes everyone, from the troll to the Good Samaritan. Ethical programs are arguments, rhetorical engagements with networked life that determine how to be connected. They establish, break, and manage relations.

After establishing the relationship between ethical programs, hospitality, and rhetoric, the following four chapters move through rhetorical analyses that examine the complications of hospitality and networked life. These four chapters are divided into two sections, with each pair taking up a key question of new media studies. Chapters 2 and 3 focus on the power dynamics of networks. The case studies in these chapters examine how exploits, software that exposes security vulnerabilities, and procedural rhetoric, the use of processes to mount arguments, can open up space for rhetorical engagement in networks. Galloway defines networks in terms of protocols, “the set of technical procedures for defining, managing, modulating, and distributing information throughout a flexible yet robust delivery infrastructure.” For Galloway, understanding the movement and distribution of power in networks requires both that we understand the material technologies that determine how those networks work and the way power is distributed through them. Most important, such analyses must resist the temptation to theorize networks as open, free, rhizomatic, or flat. Networks are not free of hierarchies, and they feature top-down assertions of power by way of protocols. Chapters 2 and 3 attempt to map out the dynamics of two particular arrangements of networked power and to describe what rhetorical action in those networks looks like.

Through an analysis of the 2008 Obama presidential campaign’s use of social networking software, chapter 2 presents one example of how protocological power operates. The Obama campaign’s mybarackobama.com (which
many referenced by way of an unfortunate acronym: MyBO) operated by way of protocol’s bidirectional power structure, exerting influence with hierarchical structures while also allowing volunteers to operate in peer-to-peer, horizontal networks. The campaign was not the first to use social networking to expand get out the vote (GOTV) efforts, but it did pioneer certain strategies and software packages to distribute campaigning activities. Volunteers used MyBO to call potential voters, gather data about those voters, organize campaign activities, and earn points that reflected their level of commitment to the campaign. While campaign leaders guided volunteers in these activities, it also allowed those volunteers a certain amount of leeway as they, for instance, determined how best to conduct phone conversations with potential voters. Thus, the campaign made use of the hospitality extended by networked life, extending invitations to volunteers, but it did so while carefully crafting its own laws of hospitality that controlled how these “guests” of the campaign operated. In my own analysis of these activities, I describe the campaign’s protocological infrastructure and how it used software to shape volunteer activities. However, the chapter also explains how Obama campaign volunteers used what Ian Bogost calls “procedural rhetoric” to navigate this complex protocological network. Procedural rhetoric is the use of processes, computational or otherwise, to persuade. Bogost uses videogames to demonstrate procedural rhetoric in action, showing how computational procedures can be used to model worlds and make arguments. But Bogost also argues that procedural rhetoric is not just useful for understanding videogames and that it can also help us understand how all procedures express arguments. This chapter puts that argument to the test, examining the procedural rhetoric of the Obama campaign and also the procedural arguments crafted by its volunteers. What we find in that analysis is that procedural rhetoric offers one possibility for taking on the complex and conflicting power relations of protocol, which extend an invitation while also providing hierarchical structures that shape and constrain activity.

Chapter 3 also takes up the question of how rhetors might resist or act within networks, but it does so by focusing on a rhetorical tactic put forth by Galloway and Thacker: the exploit. While their book, The Exploit, never uses the term “rhetoric,” their understanding of how software exploits can reshape protocological structures is profoundly rhetorical. A software exploit exposes a gap in the security or functionality of a computational environment. Exploits do not necessarily step through rational arguments about how a given digital space is designed or about the ethics of that space. Instead, exploits perform their argument by hacking a networked space and, in some cases, transforming it. In the interest of understanding the exploit as a rhetorical maneuver, I examine two exploits of the leaky boundaries of the Twitter microblogging
platform. The first is a cross-site scripting (XSS) attack that resulted in Twitter users unknowingly retweeting links to, among other things, pornography. The second primary example in this chapter involves a hole in a security protocol, OAuth, implemented for the Twitter Application Programming Interface (API). While the first exploit played out in plain view, affecting web users in real time, the hack of OAuth never saw the light of day. Addressed by corporations and programmers behind closed doors at a professional conference and in private meetings, the OAuth exploit was fixed before it could cause too many problems for implementations of the protocol (which extended to other companies as well, including Google and Facebook). Telling the stories of these two hacks is important as we try to understand the rhetorical possibilities of the exploit and how it presses against the vulnerabilities of hospitality. Beyond showing how the exploit is rhetorical and how it exposes some of the available means of persuasion, the two exploits examined in this chapter also stand as case studies of how particular digital spaces contend with the predicament of hospitality differently. A software exploit exposes the hospitality of networked environments, demonstrating how those spaces work (or how they don’t). As Galloway explains, “protocol outlines the playing field for what can happen, and where.” Exploits trace the contours of that playing field, showing us what is and is not possible, and this is the exploit’s connection to rhetoric. By demonstrating the possible, it exposes the available means of persuasion and foists a new rhetorical arrangement upon users and software designers. The implication of such computational arguments is that the space can (or should) work differently.

Chapters 4 and 5 move from the ethical difficulties of protocological spaces to questions of database and narrative, which Lev Manovich first theorized in *The Language of New Media*. Manovich describes database and narrative as two worldviews that compete in the age of new media. While narrative presents a sequence of events told in a particular order, database presents a number of possibilities at once, making fewer determinations and allowing competing narratives to coexist. It might be tempting to argue that the database is more hospitable, but such a claim would ignore the determinations that database designers must make as they decide what data and categories are reflected in a given database. Rather than determining which of these worldviews is more or less hospitable or more or less ethical, chapters 4 and 5 examine the predicaments of a world in which databases are increasingly hospitable to every keystroke. If, as Manovich suggests, narratives present particular ways of traversing databases and making sense of information, then the growth of contemporary databases has welcomed a staggering number of competing and conflicting narratives. The hospitable network extends an invitation to data, tracking every click, calling for writers and programmers
to determine which ethical programs are best suited for parsing a world in which the relationship between narrative and database is shifting.

Chapter 4 addresses the question of hospitable databases by examining one of the largest writing spaces in the world, Wikipedia, and its software platform. That software platform, MediaWiki, engages the difficulties of hospitality in particularly interesting ways. Welcoming writers from all angles, Wikipedia has had its share of controversies and has forced us to confront the problem of how to present a coherent narrative (in this case, an encyclopedia article) in the face of a massive amount of data. But Wikipedia’s controversies have not been confined to its content, and this chapter focuses on one of the more famous dustups regarding Wikipedians themselves. Essjay, a prominent Wikipedian who presented himself as a professor of theology, is the primary focus of this chapter, since his story demonstrates how ethos is one of the primary rhetorical resources in a massive database such as Wikipedia. Essjay claimed to be something he was not, but his attempts to use this constructed ethos to steer conversations about certain Wikipedia articles were often resisted by other Wikipedians. Most important, for our purposes, the chapter addresses how MediaWiki software creates a digital space in which a user such as Essjay can construct that ethos and in which other users can critique it by referencing a user’s deep archive of activity. MediaWiki tracks nearly all user activity. It is a hospitable archive that files away keystrokes, building a database that is deep and wide. In such a space, ethos becomes the Wikipedian’s primary strategy for influencing conversations about articles. In addition to examining the Essjay controversy, this chapter also analyzes a project called Citizendium, which attempts to build a wiki-based encyclopedia that avoids the trappings of anonymity. Started by one of Wikipedia’s cofounders, Larry Sanger, Citizendium also uses MediaWiki software, but Sanger’s encyclopedia institutes a set of procedures and protocols for determining the identity of writers. Citizendium’s use of such procedures demonstrates that the use of MediaWiki does not perfectly determine how identity and ethos operate in a textual space. Still, while Citizendium shows how MediaWiki does not offer a single possibility space for how writers interact, this chapter does demonstrate that MediaWiki’s proclivities for a deep textual archive are, in many ways, difficult to avoid in any implementation of the software. As we see in this chapter, software plays a crucial part in our networked rhetorical situations, and users can move through the hospitable database by leveraging (or suffering at the hands of) its deep archive.

Chapter 5 deals with this same problem of the hospitable database by examining how we might move between the worldviews of narrative and database. The chapter takes up this question by focusing on the world of profes-
Introduction

I examine robots developed by a company called Narrative Science, algorithms that are authored by teams of what the company calls “meta-writers” (comprised of both journalists and computer scientists). These meta-writers author software that generates stories, from game recaps to quarterly earnings reports. The company sees these algorithms as a way to transform the data of spreadsheets into narrative forms that are more useful to humans. Kristian Hammond, chief scientist at Narrative Science, argues that these computational systems take a “cognitive burden” off the reader of spreadsheets and databases, transforming data into narrative. In the interest of removing this cognitive burden, robot writers make ethical decisions about what should or should not be included in a narrative. These computational machines are ethical programs that determine how to move between data and story, determining what a reader does or does not see. While many take the existence of such bots as a threat to the supposedly human realm of writing, the very fact that these machines are part of the ethical terrain of networked life suggests that any engagement with database and narrative is machinic. Thus, robot and human writers are more similar than we might like to admit. In this chapter, I suggest that rhetoric’s 2,500-year history has presented us with a number of “machines” for understanding the shifting relationship between database and narrative. Rhetoric, as a set of machines for generating and interpreting arguments, provides ethical programs for moving back and forth between the worldviews of narrative and database, navigating the challenges of hospitable databases. As ethical programs, Narrative Science’s software sits uneasily between narrative and database, making judgments, and its procedures share a great deal with the tools of the rhetorician, which involve attempts to see the world from the perspective of both database and narrative. Machinic understandings of narratives and arguments allow us to gain insight into the robot writers that have joined our networked conversations and also present us with strategies for mediating the worldviews of narrative and database.

In the concluding chapter, I provide a framework for understanding the rhetorics (plural) of software that emerge in the hospitable network. I find persuasive Ian Bogost’s arguments in *Persuasive Games* that digital rhetoricians must attend to computation and that the field of digital rhetoric has too often focused on “the text and image content a machine might host and the communities of practice in which that content is created.” Recently, much current work in digital rhetoric is working to remedy this problem, and *Ethical Programs* counts itself among this group. For instance, Annettee Vee’s detailed rhetorical analysis of how computer code has been defined at various times as text, speech, and machine is evidence that rhetoricians are no longer neglect-
ing the realm of computation. Vee tracks arguments in law and among programmers about the legal status of code, demonstrating how those who write code attempt to influence such discussions (and how they sometimes aim to find workarounds for outdated laws by crafting their own laws that are better suited to the complexities of computation). Vee is not content only to study arguments about code—she links these arguments to specificities of code and software. Kevin Brock’s dissertation “Engaging the Action-Oriented Nature of Computation: Towards a Rhetorical Code Studies” takes a similar approach, offering extensive reviews of the computational gap in rhetorical scholarship and the rhetorical gaps in software studies scholarship. However, Brock also offers a way forward for rhetoricians seeking out ways to link rhetorical studies to software studies and critical code studies. He notes that work in rhetoric and writing has only begun to attend to this level of rhetorical activity, often focusing on interface at the expense of a detailed account of code and computation. He suggests a multifaceted rhetoric of code studies, one that would account both for how code itself expresses meaning and how programmers’ comments are used to argue and persuade. His study of the FizzBuzz test, a common test offered to those applying for computer programming jobs, is particularly enlightening, in that it explores how rhetorical style (and arguments about that style) circulate in programs and programming communities. Also important is David Rieder’s work on the Oulipo conceptual writing collective and how its procedural focus is of use to those attempting to link literacy with numeracy, as is his forthcoming book on physical computing, *S cues te Iterations: Rhetoric, Writing, and Physical Computing.*

Bogost’s concept of “procedural rhetoric”—the use of processes, computational or otherwise, to persuade—is the clearest contribution toward this more focused and rigorous effort to account for the rhetoric of software. However, an expanded understanding of the rhetorics of software—including, but not limited to, procedural rhetoric—can benefit both rhetoricians and scholars in software studies. The concluding chapter of *Ethical Programs* follows the scholars cited above as it lays out multiple levels of rhetorical activity that emerge in computational, networked environments. While the four case studies presented in the book take an inductive approach, tracing how software navigates the difficulties of hospitality in specific instances, this concluding chapter zooms out, drawing broader conclusions about how to build theoretical tools for the rhetorical analysis of software. This chapter serves to tie together the book’s case studies, and it makes explicit the different levels of rhetorical activity present in each case. These levels intersect and blend with one another, as each of the case studies demonstrates. From discussions about computer programs to the actual bits of code that shape our networked environments, we find multiple rhetorics of software.
Understanding how software cuts across argument, persuasion, and communication in the hospitable network requires that we account for a range of rhetorical activity. To this end, I describe three different rhetorics of software: arguing about software, arguing with software, and arguing in software. Each of these realms of rhetorical activity is shaped by the ethical predicament of hospitality that defines networked life. Networked technologies invite people from diverse backgrounds to discussions about technology. This means that arguments about software are now conducted among a broad swath of people, from experts to novices. Rhetoricians have long analyzed how we attempt to argue and persuade, so arguing about software is most in line with the long history of work in rhetorical studies. How are discussions about software conducted? What strategies are used? What counts as evidence? What power relations are at work, and who is seen as credible in such discussions? These are the kinds of questions one would ask when addressing this first level of rhetorical activity.

Arguing with software, on the other hand, insists on understanding how computation itself can be used to persuade. When arguing with software, one uses software as a tool, much like the orator uses language. Computational procedures become the claims and evidence, tropes and figures, gestures and intonations. In short, they become a persuasive medium. My use of the word “with” here introduces some productive ambiguity, since “arguing with” can evoke not only tool use but also an argument between two people or positions. The concept of arguing with software accounts for both of these meanings, since any attempt to use software as a tool for persuasion will also mean confronting the constraints of software, the ways that it shapes and constrains expression. While I use software to make arguments through the deployment of procedural rhetoric (discussed in chapter 2) as well as the exploits (covered in chapter 3), these attempts will always mean that I enter into rhetorical negotiations with the software itself. Software is both tool and interlocutor. These complexities of arguing with software are evident as programmers use computation to expand the available means of persuasion and to demonstrate what is possible in a given environment. The hospitable network invites such hacking (to varying degrees) and determines how far these explorations can go.

While the idea of software as interlocutor falls within the realm of “arguing with,” it also bleeds into the next category, arguing in software, which is a layer of rhetorical activity that accounts for how software shapes our networked rhetorical situations. Software helps to establish and institute the spaces in which we communicate and argue, and it often welcomes a deep archive of information. In this sense, we are always “in” software; we are in spaces that are shaped by computational artifacts and platforms that welcome
and track data. Understanding the rhetorical strategies that emerge in these spaces is yet another of the rhetorics of software. This discussion of software as environment might lead us to think of software as “backdrop” for communication and persuasion, but the very idea that I have to lock horns with a piece of software as I attempt to communicate demonstrates that arguing in software means more than understanding software as a kind of container for arguments and persuasion. When I am arguing in software, I am negotiating a complex rhetorical ecology of audiences, from parsers and APIs (which I take up in more detail in chapter 3) to the companies tracking my keystrokes.

While rhetoricians are accustomed to theorizing what happens at the level of “arguing about,” the other two levels of activity described here are just as relevant to the study of rhetoric and to determining how ethical programs shape our lives. In this concluding chapter, I explain these different rhetorics of software and make the case for allowing them to bleed into one another as we conduct rhetorical analyses of networked software environments. I argue that zeroing in on the computational artifacts at play in a given rhetorical ecology can help build a robust framework for understanding how rhetorical action is shaped, enabled, and constrained by ethical programs.

In the pages that follow, I examine controversies and flash points, moments when software exposes (and participates in) difficult ethical questions, without necessarily answering them in any final way. But these situations lay out questions and problems to which we are called to respond, and those responses can be understood as rhetorical engagements with complex exigencies. The ethical programs I analyze in this book take up the unending call of hospitality. They are arguments about how networked life can or should happen. Whereas an ethical program might conjure images of a set of rules by which we determine what behavior is more or less ethical, the programs I present here do not always lay out a plan or a series of criteria for judging what is or is not ethical. Instead, these ethical programs are incomplete, temporary attempts to address the difficult questions of hospitality. Those predicaments do in fact call for judgments and answers, for determinations about how we should act, but the answers are infinitely complex; even after we construct such answers, the questions remain.

Each of my rhetorical analyses follows roughly the same pattern by laying out one of the predicaments of hospitality exposed by software and then offering some of the rhetorical tactics available to those responding to that predicament. Rhetors (here understood as writers, speakers, programmers, and sometimes software itself) can use procedural rhetoric and exploits to navigate the complexities of protocological power. They can use ethos to navigate databases that track every keystroke and build a deep archive of information. They can use rhetorical theory’s long history of machinic thinking to navigate
the competing worldviews of database and narrative. Each of these rhetorical strategies is necessary because of the hospitable network, which sets the stage for protocological power and for the shifting relationship between databases and narratives. In *Lingua Fracta: Toward a Rhetoric of New Media*, Collin Brooke argues that a rhetoric of new media should be “actionary rather than re-actionary,” that it should not be content with describing how new media spaces operate but should also develop ways of understanding how to create, code, and write (in) digital spaces: “as actionary, a rhetoric of new media should prepare us for sorting through the strategies, practices, and tactics available to us and even for inventing new ones.” This book operates with this same understanding of rhetoric and digital media, seeking out not only how software shapes interactions but also how rhetors continue to develop new practices for answering the Law of hospitality. Networked life calls for new ways of engaging ever-shifting ethical questions, and *Ethical Programs* traces out how digital rhetors have already begun the search for ways to argue and communicate in complex, networked spaces.