Designing Software Ethics

Christopher N. Chapman, Ph.D.
Microsoft Corporation

Paper presented at:

Society for Philosophy in the Contemporary World
9th Annual Conference

July 2002, Santa Fe, New Mexico

Author address:
Chris Chapman (cchap)
1 Microsoft Way
Redmond, WA 98052

cchap@microsoft.com

Abstract

One may view software systems as creating interpersonal spaces akin to worlds. Because people live as moral agents who take action in the face of constraints and uncertainty, people will exhibit moral behavior in software worlds. Thus, complex software systems necessarily shape or implement the grounds for their users' moral systems (within a given software space). Instead of engaging in prescriptive ethics, I examine the ethical foundations embodied in software. I argue that software designers and technologists should become better aware of the fact that they are designing implicit ethical structures. If they do not, their systems may become increasingly unsatisfactory to users. Philosophers may be of great assistance to engineers in clarifying the tacit assumptions of software systems.

Introduction

In this paper, I argue that software engineers do not merely design computational tools and systems but also design ethical systems. In many cases, creating a software system necessarily involves the creation of an environment in which people engage in moral acts, even if such construction is unknown or unintended by the software designer. I give here an outline of such ethical systems, examine the fundamental bases or concepts of ethical software design, and note implications of this analysis for software design issues.

It is helpful first to clarify my terminology. I regard "software design" as the development of actually implemented software systems that are or will be used by groups of people, such as general consumers or people in a business. My discussion is about user-centered software design, not about conceptual issues of algorithmics. In "software design," I include all aspects of the software "lifecycle," from planning to programming to documentation to support. For convenience, I distinguish behavior mediated by software from behavior taking place through other physical systems (such as face to face interaction), as a distinction between "software worlds" and the "real world."

I distinguish "morals" from "ethics" in the following way. I take "morals" to be those principles that individuals use to make decisions about actions in the face of conflicting demands, desires, or internal or external constraints. I use "ethics" to denote the theoretical analysis of such systems of morals, such as the foundational aspects that are required by or implicit in such systems. Note that I use both words in respect of the systems of such behavior, not of the behaviors themselves. Thus, if a rule is "moral" in my usage, it applies to cases of human decision-making in the face of conflict; this does not imply a value judgment that any particular action or principle is good or bad. (Such a distinction is, of course, inexact, so I beg forgiveness in advance for my inevitable lapses.)

The basic claim of this paper can be stated as a deduction. First, software design sometimes creates spaces for human interaction. Second, if any such space is sufficiently diverse and affords rich interaction, then it opens an avenue for moral behavior. Third, in that case, software design is sometimes tantamount to the design of ethical systems. On the basis of that claim, I note areas in which ethical issues arise in software development. For purposes here, I base my account on an application of ethics that arises from recent reconsideration of early Heideggerian theory (e.g., *Being and Time*). However, that basis is contingent and not necessary; other ethical theories certainly would apply similarly. For that reason, I try to avoid Heideggerian terminology.

There are many areas of “information technology ethics” that I deliberately avoid here in my focus on the ethics implicit in software. Among the areas that I avoid are: (a) the impact of technology on society at large; (b) the impact of computer science and software on conceptions of humanity and human nature; (c) the aims and professional ethics (or “morals”, to be more precise) of engineers and software developers; (d) problematic usage of software (software piracy, hack attacks, and the like); and (e) issues of access and equity (poverty, access for people with disabilities, language issues, etc.). These are important topics, but have received much attention in the existing literature (e.g., Hester and Ford 2001). In contrast, as far as I am aware, my argument here is unique.

Software Worlds

Software design has increasingly attended to the behavior of software users, rather than being exclusively concerned with formal algorithms and data structures. One may distinguish two foci of this attention to people: an “engineering” approach, in which user behavior is studied with behavioral methods akin to those of any other laboratory science, where the objective is to optimize some metric of behavior (e.g., Schneiderman 1998, pp. 61-62); and a “user-centered” approach that studies the interaction of people with one another, and views software as a tool used by people as part of complex ecosystems of working and living, which are best examined by methods involving naturalistic field study (e.g., Beyer and Holtzblatt 1998).

Some computer scientists have applied Heideggerian theory to describe the position of people in a social network where software provides a framework similar to that of Heidegger's concept of "world" (Winograd and Flores 1987). In these accounts, software environments do not provide a rigid structure, but are places where people interact, create new modes of behavior, and learn, grow, and change over an extended course of time. Software may create a place where people are able to "take care" and express their "innerworldly" existence as "being-in-the-world" (Heidegger 1996 (1927), p. 141/151). There are other reasons to speak of existence in software-mediated worlds, quite apart from a Heideggerian analysis. For instance, talking of Cyberspace as a distinct "place" has been suggested as one approach to resolving legal issues over jurisdiction for online activities (Johnson and Post 1996 (2001)).

In the neo-Heideggerian view of Winograd and others, complex software may exceed the characteristics of a tool, and starts to take on the attributes of a "totality of useful things" (Heidegger 1996 (1927), p. 64/68). If this view is correct, then people engage with software worlds, in some cases, in very much the same way that they engage with the "real world." They project themselves into relationships with others, and relationships with their own past and future behaviors, and do so in ways that shape themselves and, in turn, form the environment in which they and others live and grow (in respect to the software "world" in question). Chapman writes of software:

> This [Heideggerian] account radically extends our understanding of why people may be passionate about software, why they may derive joy or pain from interacting with it, and why they may form attachments to it. When software is considered as establishing a world in Heidegger's sense – a place

> in which people live and in which their existential natures are exposed, affected, effecting, changing, and being challenged – then we are not surprised to find that the range of human responses to software that is broader than [an empiricist or] cognitivist account would predict. (Chapman 2001, p. 9)

That account does not deny the place for circumscribed or empiricist understanding of software usage, but argues that people's interactions with technology and with one another, mediated by technology, are richer than a simple cognitivist model can explain.

What kinds of software systems qualify as "worlds"? Software could be arranged on a continuum with no distinct boundary between "tool" and "world." However, systems that qualify include some cases of the following: online discussion communities; email discussion groups within organizations and among scholarly or business colleagues; online multi-player game systems; networks in which professionals work on shared projects (such as architectural plans and computer code); and the desktop environments in which people work (e.g., their operating systems and common business applications).

Ethical Existence

Several scholars have recently investigated the degree to which Heidegger's picture of human existence (Dasein) includes ethical concerns. To be sure, Heidegger generally ignored ethical issues in his philosophy (except insofar as ethical concerns may be laid over his discussions of authenticity and technology) and engaged in odious behavior during the Nazi regime. Despite Heidegger's professional disinterest and personal failure, it seems clear that ethical issues necessarily arise for people (Dasein) in Heidegger's thought.

Hatab notes that *Being and Time* points to the essential finitude of human existence, and argues that this finitude necessitates moral engagement with the world. Moral choices are necessary that we may act in a world where choices must be made on the basis of conflicting and scanty information (Hatab 2000, pp. 54-59). People's engagement with the world, he says,

> is "always already" ethical in some way … [T]raditional heritage, upbringing, habits, social practices, linguistic competence, and predisposed interests shape Dasein as an evaluating being, in terms of disclosing better and worse, important and unimportant modes of life. Values, then, go all the way down…. Our everyday social interactions, habits, competencies, and expectations are permeated with and reflect a host of values, norms, and preferences, ranging from common courtesy and etiquette to basic social customs and ethical routines. (Hatab 2000, pp. 59, 64).

People, then, are necessarily engaged in ethical systems. Olafson argues further that Heidegger's account of engagement (Mitsein) constitutes the ground for ethical authority (Olafson 1998, p. 11). However, that claim is stronger than we need here.

If we accept that human existence is necessarily ethical, and if software worlds constitute a meaningful "place" where people exist (whether on a Heideggerian account or not), then there must be ethical issues inherent in those worlds. Such issues are apart from, and in addition to, the traditional “information technology ethics” questions that arise for technology as a part of the “real world.”

<u>Ethical Discourse in Software</u>

If software worlds are ethical, then software designers are performing “ethical design,” even though they are likely unaware of that fact. In other words, they are designing systems that embody fundamental assumptions about the kinds of interactions that users can have, and which therefore determine the kinds of ethical issues and customs that are possible in those systems. I propose that there are 3 fundamental stances that software takes with respect to interpersonal interaction, and that these are, in fact, ethical assumptions. These fundamental stances concern personal identity, the structure of social discourse, and the structural conditions for social customs and rules (moral behavior). I would like to note differences between software worlds and the real world with respect to each of these.

First, the structural conditions for social customs and rules in the real world are established entirely by the operation of human institutions and individual human actions. Deep and well-established parts of social existence, such as the fact that there are social rules that one should or is compelled to follow, have developed over thousands of years of human existence. More malleable rules, such as a country’s specific laws, change through political processes; those processes are in turn based on more fundamental kinds of rules or assumptions (including coercion, of course).

In any case, these moral assumptions are formed by social processes that occur in the same world in which the assumptions shape behavior and lives.

Software worlds differ from that picture in two fundamental ways. First, the assumptions embedded in them are likely not to be formed by the people interacting within the world, and are probably not very malleable. Instead, the moral assumptions, or at least the scope and boundaries of the assumptions, are created by the system developers. Second, changes to these assumptions occur on a rapid timescale and are likely to occur discontinuously; newly implemented features immediately change the possibilities of interaction in the system.

Let me introduce a psychological hypothesis: people behaviorally expect stability and consistency in ethical systems across the real or virtual communities in which they participate. If so, when users experience a software system whose ethical assumptions are different than they are accustomed to in the real world, they will respond to that discrepancy in some way. For instance, they may feel uneasy or may complain about the constraints placed upon them by the software (or by the real world). It is not difficult to predict a response for users faced with inflexible ethical assumptions that are subject to discontinuous change: they might feel as if some personal need, which they can’t identify in terms of specific features, has been ignored, and they may be unlikely to embrace opportunities for continued change (such as software upgrades).

Those user responses occur, of course, and I think software developers can do a much better job of building systems that allow users and system administrators to have flexibility in how they build and maintain social rules systems. Also, software should pay more attention to questions of continuity in the rules that apply to users; it is perfectly reasonable for people to expect that a system upgrade will not cause large-scale discontinuity in features, in personal performance, or in the types and grounds of social interaction that the system permits.

With respect to the structure of social discourse, assumptions in real world ethical systems include expectations that other people are present to one another physically or can be present in principle; that they share a language, or are able to adapt interactively to arrive at a mutually shared language; and that interactions are guided mutually and can be engaged on the basis of physical presence (e.g., we must agree to be present in order to communicate; or, if someone leaves suddenly, I can follow him and attempt to resume a conversation after an apology).

I think these problems of social discourse can be handled adequately in software systems. In the case of language, software worlds function like any other linguistic medium. Although Dreyfus argues that the lack of bodily presence fundamentally and severely restricts software worlds such as online communities (Dreyfus 2001), I see that issue not as a problem of bodily presence as such, but as a question about what sense can be made of any stable presence in whatever terms are applicable to the given software world. Thus, being an identifiable addressee, who participates in a timely fashion, and so forth, is an adequate “presence” for me in an e-mail discussion system.

However, that issue ties into a final problem that is pervasive and fundamental in the design of software ethics, yet arises in the real world only in exceptional cases: the problem of personal identity. Without identity, there is no definite moral agent who makes decisions and no actor who is identifiable to other actors in a moral system (e.g., for praise or punishment).

Traditional ethical systems have been able to assume that a moral agent is identifiable both to herself and to others, and have further been able to assume that a person is uniquely identifiable with a single agent (and vice versa), that all actors are human, that identities are public (in the sense that the actor may be apprehended, i.e., observed in person), and that the identity is stable over time. Each of these assumptions is problematic or even false with respect to actors in software worlds. Thus, to talk about software ethics, we must rethink the notion of identity with respect to software.

People using online software systems often are able to have multiple identities and to change identities at any time. Turkle examined the implications of this fact from the perspective of personal identity, and argued that software worlds allow people to experience and experiment with various personality facets that are unavailable in their real world roles (Turkle 1995). This fact has ethical implications, of course, in terms of what it means to assign responsibility. Any software system takes a stance on personal identity in the software world: users will have some identity, or multiple identities, or a physical network address, or no identity at all (e.g., when the system rejects the concept of an identity as in the case of purely anonymous network access). No matter which design they choose, software engineers implement an identity system that shapes the ethical system for their users.

Software identities are not necessarily human, or distinguishable from human, and may not persist in any predictable way over time. In addition, identities in software are not necessarily or clearly connected to any real individual. There are many technologies that offer differing solutions to these problems. For instance, some technologies attempt to sever all links between real and software identity by guaranteeing anonymity (e.g., by masking network addresses); others such as biometric devices attempt to bridge real and software worlds by linking real physical attributes to software identities protected via encryption.

The key point to understand is that no such technology can offer a solution to all problems, because each implements a set of ethical assumptions that address concerns for some users and interested parties but not others. Thus, people who prefer an ethos of freedom may endorse a solution that emphasizes anonymity while maintaining online reputation (May 1994 (2001)); those who worry about the ability of governments to enforce their responsibilities argue that governments should maintain a system to link digital and real identities (Denning, 1996 (2001) #27, although she moderated her view in Denning 2001).

For software designers, none of these issues is one of pure technology, nor of simply acceding in an unthinking manner to the demands of the ethical system in which we operate as developers. Rather, as software designers, we should begin to think and talk more about the kinds of ethical systems that we are creating for our users. Why should we do this? Because if we don’t, it seems increasingly likely that we will be unable to fulfill the human demands placed upon our complex software: our users may be uncomfortable and discontent, our systems may lurch from one tacit moral system to another, and our software may be unable to adapt fluidly to the moral world surrounding it. Ultimately, then, if our software worlds ignore ethics, they may fail to find inhabitants.

I hope that this work will serve to awaken engineers to the fact that they are designing ethical systems, whether they like that or not, and will encourage developers and philosophers to work together to clarify the ethical assumptions that are increasingly important in software.

References

Beyer, H., and K. Holtzblatt. *Contextual Design: Defining Customer-Centered Systems*. San Francisco: Morgan Kaufman, 1998.

Chapman, Christopher N. "Thinking with Heidegger about Software Usability." A paper delivered at the Society for Philosophy in the Contemporary World, Annual Meeting, Santa Fe, NM, 2001.

Denning, Dorothy E. "Afterword to "The Future of Cryptography"." In *Crypto Anarchy, Cyberstates, and Pirate Utopias*, ed. Peter Ludlow, 103. Cambridge, MA: MIT Press, 2001.

Dreyfus, Hubert L. *On the Internet*. London: Routledge, 2001.

Hatab, Lawrence J. *Ethics and Finitude: Heideggerian Contributions to Moral Philosophy*. Lanham, MD: Rowman & Littlefield, 2000.

Heidegger, M. *Being and Time*. Translated by J. Stambaugh. Albany, NY: SUNY Press, 1996 (1927).

Hester, D. Micah, and Paul J. Ford, eds. *Computers and Ethics in the Cyberage*. Upper Saddle River, NJ: Prentice-Hall, 2001.

Johnson, David R., and David G. Post. "Law and Borders: The Rise of Law in Cyberspace." In *Crypto Anarchy, Cyberstates, and Pirate Utopias*, ed. Peter Ludlow, 103. Cambridge, MA: MIT Press, 1996 (2001).

May, Timothy C. "Crypto Anarchy and Virtual Communities." In *Crypto Anarchy, Cyberstates, and Pirate Utopias*, ed. Peter Ludlow, 65-79. Cambridge, MA: MIT Press, 1994 (2001).

Olafson, Frederick A. *Heidegger and the Ground of Ethics*. Cambridge: Cambridge University Press, 1998.

Schneiderman, B. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. 3rd ed. Reading, MA: Addison-Wesley, 1998.

Turkle, Sherry. *Life on the Screen: Identity in the Age of the Internet*. New York: Simon & Schuster, 1995.

Winograd, Terry, and Fernando Flores. *Understanding Computers and Cognition: A New Foundation for Design*. Boston: Addison-Wesley, 1987.