1/13/10

Software Agents, Anticipatory Ethics, and Accountability

Deborah G. Johnson
STS, University of Virginia

Abstract

In search of innovative ways to make law and ethics more dynamic, flexible, and able to keep pace with rapidly developing technologies, this chapter takes up the case of software agents. Software agent technology is a good candidate for anticipatory ethics because it is still in its formative stages, and at least some of the discourse shaping its meaning explicitly involves moral notions. Anticipatory ethics refers to engagement with the ethical implications of a technology while the technology is still in the early stages of development, engagement that is targeted to influence what is developed. The anticipatory ethics endeavor has been facilitated by a major shift in understanding of technological development. Technological development is now understood to be fluid, contingent, and involving intricate negotiations between stakeholders and human and non-human components. Embedded in a critique of technological determinism, scholars of science and technology studies (STS) have provided concepts and theories that explain how technologies are socially negotiated and constituted. After describing how the shift in understanding has made room for anticipatory ethics, this chapter takes up the challenges of anticipatory ethics for software agent technology.

The starting place for the analysis is the concern that software agent technology may be on a collision course with moral notions and practices of accountability; the potential for collision arises, in part at least, from the characterization of software agents as autonomous. The chapter puts forward and examines an argument for a moral ontology for software agents. A moral ontology would conceptualize software agents so as to keep them tethered to those who design and deploy them. The analysis considers several counters to this argument and concludes with some lessons for anticipatory ethics.

Software Agents, Anticipatory Ethics, and Accountability

Deborah G. Johnson, STS, University of Virginia

> *"Classification does indeed have its consequences – perceived as real, it has real effects*." – Bowker and Star, 1999.

1. Introduction

An important current trend in software development is to produce complex systems designed to operate independently from the humans who design and deploy them. Often referred to as *software agents*, at least some of these systems are able to learn and make second order decisions that effectively reprogram how they operate. Software agents may be deployed to perform fairly simple transactions (such as purchasing a product) or to perform extremely complicated operations involving sophisticated decision making (such as the software onboard the Earth Observation satellite that decides which events on earth the satellite should monitor (Chien, Sherwood, Tran, Cichy, Rabideau, and Castano, 2005; Noorman, 2008)). Software agents are not distinctive insofar as they operate separately in space and time from those who deploy them, but rather because of the kinds of decisions they make and their capacity to learn and behave proactively often through processes that are not transparent to designers and users. Generally, this technology is understood to be an extension or species of what was earlier thought of as artificial intelligence.

For some the characterization of software as *agents* seems a fairly straightforward matter of describing software that performs tasks on behalf of humans:

> The Intelligent Software Agents Lab at Carnegie Mellon University's Robotics Institute envisions a world in which autonomous, intelligent software programs, known as software agents, undertake many of the operations performed by human users of the World Wide Web, as well as a multitude of other tasks. (http://www.cs.cmu.edu/~softagents/intro.htm)

For others, the autonomous aspect of software agents is what is significant because it means that they can operate in open-ended situations. The aim of the annual AGENTS: International Conference on Autonomous Agents is explained in the ACM Portal:

> Autonomous agents are software and robotic entities that are capable of independent action in open, unpredictable environments. Agents are currently being applied in domains as diverse as computer games and interactive cinema,

> information retrieval and filtering, user interface design, electronic commerce, autonomous vehicles and spacecraft, and industrial process control. (http://portal.acm.org/browse_dl.cfm?linked=1&part=series&idx=SERIES134&coll=portal&dl=ACM)

For some the nature of the software justifies a category of autonomous cognitive agents in which software and robots are apparently together with humans, though distinguished as artificial:

> Autonomous cognitive agents, whether natural or artificial, are information processing entities that make decisions, recognize patterns, gather information and perform actions. The concept of autonomy refers to the ability to use experience to determine action. This includes being able to adapt behavior in order to pursue goals under changing circumstances. Artificial agents can take a range of forms from software agents to anthropomorphic robots. (Lee and Lacey, 2003)

The metaphor of software as autonomous agents facilitates the use of other concepts, such as negotiation, that are helpful in describing the behavior and operation of software:

> Automated negotiation is a powerful (and sometimes essential) means for allocating resources among self-interested autonomous software agents. A key problem in building negotiating agents is the design of the negotiation strategy, which is used by an agent to decide its negotiation behavior. (Rahwan, Sonenberg, Jennings, and McBurney, 2007)

To think of software as an agent and describe it as autonomous is, as already indicated, to use a metaphor but why this metaphor? What role or function does the metaphor play in the discourse and the design of software agent technology? Metaphors are important in shaping human understanding, though they can be dangerous when they lead to false presumptions or hide key features of the thing being explained. How does the autonomous agent metaphor work in the case of software? Obviously it provides a way of thinking about software, but what is at stake for those who use the metaphor? Does it lead to false presumptions? And what does it hide?

Importantly, use of the metaphor connects software agents to a number of fictional entities described in popular literature and media. Whether we take the monster in Mary Shelly's *Frankenstein*, the computer, Hal, in the film *2001* or the robots in the more recent film, *I Robot*, connections to the discourse of software agents are obvious. Indeed, the popular literature seems to express a mixture of fascination with and fear of these technologically-created, human-like

beings as does the discussion of software agents of today. The possibility of humans losing control of human-made entities was the major thrust of Bill Joy's "Why the Future Doesn't Need Us" (2000). When it first appeared, Joy's piece unsettled many scientists and intellectuals because an insider – a leader in the science and engineering community – was expressing concern about, and even reluctance about going forward with combining genetics, nanotechnology, and robotics. The combination, Joy warned, might be so powerful as to produce entities that would take over the world and render humans irrelevant.

As will be discussed further in the next section, the ideas that circulate during the early stages of development of a new technology influence the construction of meaning as well as the material design of the technology. Ideas contribute to the delineation of a technology as something in particular. New technologies can challenge deeply rooted beliefs about what it means to be human; they can challenge the distinction between what is human and what is artificial; and raise daunting normative questions about how the human-technology relationship should be constituted. The discourses around new technologies both *express* these deep human concerns and *shape* the developing technology.

This chapter begins with the premise that the characterization of software as *autonomous agents* is influencing the meaning and material design of the technology; that is, use of these terms and the metaphor is having an effect on what will eventually be developed and how it will be understood. The chapter takes as its starting place the observation that using the metaphor of autonomous agents may be setting the scene for a collision with moral and legal notions and practices of accountability. If software agents behave autonomously, in ways not fully understood by their human designers or users, who will be accountable when something goes wrong? How will accountability be handled when the behavior of software agents leads to harmful consequences?

To comprehend the potential for collision, consider a worst case scenario. Suppose: 1) very powerful software agents are put to use in activities that have powerful consequences for human well-being; 2) no human beings are able to understand what these powerful agents have learned and how they make many of the decisions they make since they act autonomously; 3) a software

agent's behavior is the major factor leading to a catastrophic event (e.g., an industrial accident, launch of a nuclear weapon, a major electricity shut down); 4) victims of the event and the public call for an explanation, that is, they demand that someone be held accountable for the event and the harm done; and 5) victims and the public are told that it was the behavior of the software agent that led to the event, that no human can understand why the agent did what it did and, therefore, no human beings are responsible for the behavior of the software agent.

Whether this scenario ever occurs, software agents are being designed to perform tasks, to learn as they operate, and to change their decision-making strategies for achieving designated tasks as they learn. When the behavior of a software agent results in harm to humans, issues of accountability are likely to arise. Yet the conceptualization of software agents seems to be setting the scene for a deflection of responsibility, at least *human* responsibility for the behavior of software.

2. Making Room for Anticipatory Ethics

Given the potential for collision between the development of software agents and prevailing moral and legal notions of accountability, software agent technology seems an ideal case for anticipatory ethics. *Anticipatory ethics* refers here to: (1) engagement with the ethical implications of a technology while the technology is still in the earliest stages of development; and (2) engagement that is targeted to influence the development of the technology. Software agent technology is in the early stages of development; thus, it offers the possibility of anticipating accountability issues now, rather than waiting until the technology is well developed and an untoward event occurs. The challenge is to see whether issues of accountability can somehow be taken into account and incorporated in the design and early thinking about this technology, with an eye to avoiding collision.

Since anticipatory ethics is a new approach to addressing ethical issues related to technology, some background will be helpful in understanding the endeavor. Currently, the most visible and well-funded attempt at anticipatory ethics in the U.S. is focused on nanotechnology. A major impetus for this work has been a government mandate (and the availability of resources) to examine the social and ethical implications of nanotechnology. The 21st Century

Nanotechnology Research and Development Act (Public Law 108-193 passed in 2003) specified that the National Nanotechnology Infrastructure (NNI) include activities that ensure that "ethical, legal, environmental, and other appropriate societal concerns… are considered during the development of nanotechnology." A number of scholars have risen to the task and a growing literature on the social implications of nanotechnology and nanoethics has developed. (See, for example, the journal *Nanoethics* and most recently the *Yearbook of Nanotechnology in Society* (Fisher, Selin, and Wetmore, 2008)). Since nanotechnology is still very much a technology (technologies) in-the-making, work in this area falls into the category of anticipatory ethics.

Why nanotechnology? Why now? Why address the social and ethical implications of a technology before it is ready for use? At least part of the explanation has to do with a shift in understanding of the processes of technological development. Until recently (i.e., the last several decades) and with a few exceptions, ethics scholars paid little or no attention to technology. Technology was considered irrelevant to ethics both because ethics was understood to be about human behavior and because technology was thought to be neutral – values lay in how humans used technology. Interest in technology began to develop in the last half of the twentieth century. Some would say it was the powerful social effects of an array of modern technologies including the atomic bomb, industrial chemicals, computers, and genetic engineering. Whatever the underlying causes, concerns about technology coincided with the movement in practical ethics. The attention of ethicists slowly turned to technology, especially computers and information technology.

Initially ethicists adopted a framework in which technology was understood to be, primarily, the outcome of the work of scientists and engineers (Johnson and Wetmore, 2008). In this framework, whether we distinguish scientists and engineers and whether they work in the ivory tower, in corporations, or in government, the presumption is that technologies develop somewhat separately from society, and, when completed, are delivered to society. Society can then chose whether to adopt a delivered technology, and, if adopted, a technology may then have *social impacts*.

The task of ethicists is, in this framework, to examine the social impacts and to note and address how the introduction and adoption of a technology creates ethical issues and affects important social values.[1] The social impact framework presumes that scientists and engineers work in relative isolation, doing what nature dictates, and that the resulting technologies are neutral. Values come into play when humans decide whether or not and how to take up what has been delivered. Working within this framework, ethicists do not ask about, let alone examine, the social forces, the institutional actors, the interests, or the values that have directed attention and resources to a particular technological endeavor; nor do they examine the factors that determined the design features of a new technology. Engineers are understood to be *applying* science, and since nature dictates science, engineers are constrained by nature.

In this framework it may look like nature necessitates the kind of technology that is produced, i.e., the kind of airplanes, medical devices, and power plants that are "delivered to society." And, if nature dictates the character of technologies, then there is little room for ethics or values to come into play. The only role for ethicists (or consumers and users for that matter) is to decide whether or not, and how, to use the technology that scientists and engineers deliver. The primary role for ethics, in this framework, is reactive. Ethicists can critique what is delivered; for example, they can show how surveillance technologies violate privacy. They can call for modifications in design; for example they can call for wider sidewalks and ramps next to stairways to ensure access by those confined to wheelchairs. Or ethicists can analyze social practices involving technology; for example, ethicists have analyzed the fairness of various procedures for distributing scarce medical resources. In this mode of operation, it is not surprising that ethicists may be accused of being *anti-technology* for in their reactive role, they are more likely to notice technologies that disrupt or threaten moral practices or values than to notice those that fit neatly in or enhance prevailing moral practices and values.

To be sure, ethicists working in the social impact framework have made important contributions; the fields of biomedical ethics, computer ethics, and environmental ethics have flourished with this model of technological development. The problem is not that the framework prevents the

[1] This is the framework I presumed in most of my early work on computer and engineering ethics; see, for example, *Computer Ethics* 1st edition, Prentice Hall, 1985.

lens of ethics from being brought to bear. Rather, the problem is that the framework pushes the processes by which technology is developed out of sight; it turns attention away from technology while it is still in-the-making – while its meaning and material design are in the process of being set. In short, the framework turns the lens of ethics away from the earliest and, arguably, most powerful stages of technological development.

Scholars in the field of science and technology studies (STS) have provided a critique of the social impact framework and introduced alternative models of the processes by which technologies are designed, adopted, modified, and used. The literature emphasizes that technology develops in a social context, by means of social processes, and that technology is not just material objects, but rather sociotechnical ensembles – combinations of artifacts, social practices, institutional arrangements, systems of knowledge, and nature. To say that technologies are sociotechnical is to say that technological endeavors are achieved by combinations of artifacts and social practices. For example, software does nothing on its own; software functions in combination with hardware (e.g. computers, electrical systems, routers) and humans organized in various ways (in organizations, agencies, families, etc.) and behaving in particular ways (e.g. using keyboards, responding to signs on screens, interpreting output).

Since the processes by which technologies are designed and developed and come to have meaning are *social* processes, ethicists and ethical notions can be and often are part of the processes. That is, since technological development is not entirely dictated by nature and since what is developed is a function of social factors and arrangements as well as nature, ethical notions and ethicists can influence what is developed.

This has both normative and descriptive implications for anticipatory ethics. Since technological development is a social endeavor, the endeavor can be intentionally structured so that ethical concerns are taken into account early on. This is precisely the normative mission of the 21st Century Act; it structures the research environment for nanotechnology to ensure that ethical issues are addressed early on. But there are also implications for understanding how technological development occurs and always has happened. That is, social forces, stakeholder interests, politics, and history have always influenced the development of technologies and so

have moral notions and practices. Think here of the debate about stem cell research; it is a debate about whether or not and how a moral belief should shape the processes of scientific and technological advancement. And think of the regulations with regard to the use of human and animal subjects in research and how these moral concerns have affect the nature of research.

So, ethical notions and practices and ethicists *do* in fact influence technological developments and *could and should* have a more intentional role in shaping technologies in the future. We can examine the influence of moral notions and practices on past and current technologies and we can consider how best to structure (or restructure) design and development processes so as to give ethicists and moral notions a role in development processes.

With the new understanding of technological development, anticipatory ethics is not just plausible but is an activity that has been ongoing, albeit often below the surface of recognition and somewhat out of the grasp of intentional efforts. Acknowledging that ethicists and ethical notions and practices can and do have a role in shaping technological development is, in some sense, the easy part. The hard part is to figure out how to bring ethical notions and practices and ethicists explicitly, intentionally, and effectively into the fray. The difficulty of the task will now be illustrated by taking up the case of software agents.

3. Anticipating Software Agents: An Argument for Moral Ontology

Where might we look to observe moral notions and practices being negotiated in software agent technology? Where might we look for opportunities to normatively influence what is being developed? In the case of software agents, opportunities are not hard to find because at least some of the discourse around this technology is explicitly directed at its moral features and moral implications. This literature clusters around the question whether, or in what sense, software agents (and embodied machines containing software, e.g., robots) can be said to be moral agents (Floridi and Saunders, 2004; Allen, et. al., 2005; Allen, et. al., 2000; Moor, 2006). The issue arises in part at least because autonomy is a central component in traditional notions of moral agency. Philosophically, morality only makes sense for autonomous beings; moral agency is only possible in entities with moral autonomy. Thus, if software agents or robots have autonomy, they are candidates for *moral* autonomy and moral agency. This has led a number of

philosophers to entertain the possibility of *artificial moral agents* (or AMAs) (Floridi and Saunders, 2004; Moor, 2006; Wallach and Allen, 2009).

The theoretical possibility of artificial moral agents is embedded in a discourse of artificial intelligence, computational modeling, and cognitive science. If computation can "unlock the mysteries of the universe" by modeling reality, then it ought to be able to model morality. In other words, if morality is comprehended by humans through their intelligence and cognition, then *artificial* intelligence and computational cognition should be able to model morality. The model can, then, be embedded in software and hardware to produce entities that behave in ways that are comparable to human moral behavior. These entities, according to the argument, will be artificial moral agents.

The interest of computational modelers in developing artificial moral agents has converged with a more pragmatic, computational endeavor to program machines to behave morally. The pragmatic endeavor is to ensure that when software makes decisions, the decisions it makes (and the consequent behavior) accord with morality. *Machine Ethics* is the term being used for this activity. Anderson and Anderson (2006) describe the goal as follows:

> A goal of machine ethics is to create a machine that's guided by an acceptable ethical principle or set of principles in the decisions it makes about possible courses of action it could take. The behavior of more fully autonomous machines, guided by such an ethical dimension, is likely to be more acceptable in real-world environments than that of machines without such a dimension. (Anderson and Anderson, 2006, page 10)

So two different interests – one in modeling morality as an exercise in artificial intelligence and computational cognition and the other in building decision making devices that incorporate moral principles – converge on a set of questions about the status, meaning, and significance of software agents. The convergence has generated a rich discourse, though it is wide-ranging both because it draws from a variety of disciplines and because it spans a spectrum from highly speculative futuristic visions to concrete programming strategies.

The discourse has not eschewed discussion of responsibility and accountability because in moral theory, notions of agency and autonomy are intertwined with moral responsibility. Individuals have moral responsibility for their behavior in virtue of their having autonomy. If individual behavior were entirely controlled by factors outside the individual or outside the control of the individual, then the individual could not be held morally responsible for their behavior. In this context the autonomy of software agents is crucial to claims about their moral agency. As well, in this context, the move to locate moral responsibility in the software agent seems plausible if not necessary.

The convergence of interests and the resulting discourse around software agents takes us back to the starting place of this chapter, the potential for a collision between the development of software agents and notions and practices of accountability. The discourse around software agents is a discourse about the meaning, significance, and design of a technology. It is about what to 'make' of software agent technology – how it should be understood, what features it should be understood to have, and what role it should have in human lives. The collision course concern is that the construction of software agents as autonomous (moral) agents 'makes' them something that humans will not be able to understand and control. That they are autonomous may mean that they will be – to some extent at least – beyond human comprehension and control. It is this construction of software agents that seems likely to collide with notions and practices of accountability.

The question is, then, whether anticipatory ethics can or should intervene to avoid a collision or to fit moral notions and practices to what is being developed. As will be illustrated in a moment, it is in the nature of anticipatory ethics that we can't be sure that the collision will take place without intervention; hence, the question whether anticipatory ethics should intervene is by no means simple.

In the remainder of this chapter I will put forward and defend an argument of the kind that seems to be called for by anticipatory ethics. It is an argument for rejecting the characterization of software agents as autonomous. The argument was initially introduced in Johnson and Miller (2008). Here it is extended and elaborated as a way of exploring the promises and pitfalls of

anticipatory ethics. To be sure, it is an odd argument since individuals can use words as they wish and they would not be using "autonomous" if its use did not achieve some useful purpose. Although odd, the argument is, nevertheless, important because it does precisely what is sought in anticipatory ethics. It aims to influence the technological endeavor early on by influencing the construction of the meaning of software agents. One of the most powerful ways to change a process and its outcomes is to change the understanding of the endeavor. A change in the understanding of what is being sought ultimately changes what is produced. Thus, a change in the conceptualization, understanding, and discourse of software agent technology would be a significant outcome of anticipatory ethics.

The Argument

In essence, the argument is an argument for a moral ontology.[2] Since the core idea of anticipatory ethics is that moral concerns be taken into account early on in a technology's development, the argument is that accountability issues should be used in conceptualizing what the technology is and identifying its distinctive significance.[3] Software agents ought to be understood – ontologically – as human-made components of sociotechnical systems. They ought to be understood as components of systems constituted and deployed by humans, for human purposes. Software agents function as a result of combinations of humans and artifacts working together. Constructing software agents as autonomous gives the technology an ontological status that disconnects its behavior from those who design and put it into use. Conceptually tethering software to the humans who design and use it constrain the temptation to move the locus of accountability for the harmful effects of software agent behavior away from humans to software agents (Johnson and Miller, 2008).[4]

Anticipating Accountability

The argument gains support from an analysis of notions and practices of accountability. At the most basic level, systems and practices of accountability involve the idea that individuals and

[2] I am grateful to Martin Anderson for first characterizing the argument in this way (as moral ontology) while it was still somewhat inchoate in my thinking.

[3] The argument is inspired by Bowker and Starr (1999) and other work that points to the powerful effects of systems of classification.

[4] Of course, software behavior causally contributes to events with untoward consequences. See Johnson and Powers (2005). The locus of accountability is connected to but different from causality.

collectivities (organizations, companies, agencies, countries) are expected to behave in particular ways – according to norms, standards, expectations, or principles. When an individual or a collectivity fails to adhere to a norm, the individual or organization is expected to explain, that is, to give an account, and, depending on the account given, the individual or organization may be liable to certain consequences – shame, mistrust, punishment, compensation, further scrutiny, etc.

Accountability works both retrospectively and prospectively. In retrospective accountability one is held to account for one's actions (or inactions) after they have occurred. We most often see retrospective accountability operating when something untoward has happened, something, that is, that was not supposed to happen. Thus, in the aftermath of Hurricane Katrina, FEMA was held accountable for not responding quickly to the event. When Bernie Madoff's ponzi scheme was uncovered, Madoff was held retrospectively accountable for years of deceitful, exploitative, criminal behavior. In both cases consequences followed, though the nature of the consequences varied. FEMA's Director was fired; public trust in the organization was diminished. Madoff's retrospective accountability involved arrest, trial, testimony, and jail.

Prospective accountability is future looking; it involves practices that inform and remind individual or institutional actors that they are expected to behave in certain ways. In certain domains of life, prospective accountability has an added dimension; individuals and organizations are formally required to demonstrate, by giving an account of some kind, that they are adhering to rules or fulfilling their responsibilities or doing what should be done to prevent untoward events from happening in the future. The most salient examples of this aspect of prospective accountability are in institutional accountability. Employees are asked to fill out conflict of interest statements to demonstrate that they are not in relationships that might bias their decisions. Public companies are required to provide reports to demonstrate that they are fulfilling responsibilities to stockholders.

These two forms of accountability work together. Prospective accountability is aimed at preventing the occurrence of incidents or events that would call for retrospective accountability. Likewise, retrospective accountability supports prospective accountability in the sense that when

individuals and organizations are retrospectively held to account, it demonstrates to others that they are accountable for their behavior; that is, it reminds others that if they don't behave according to norms or expectations, they may have to account, retrospectively, for their behavior.

The argument for a moral ontology for software agents is an argument for conceptualizing the technology in a way that will facilitate the operation of prospective and retrospective accountability. The argument is especially compelling because of the current state of legal accountability (liability) for software. In the U.S. at least, legal liability for harmful effects resulting from the use of software is highly uncertain. At best it is a patchwork of generic laws and extremely varied case law drawing on contracts, strict liability, negligence, misrepresentation, limited warranties, and unconscionable disclaimers (Zollers, et. al., 2005; Ballman, 1996; Terry, 2002; Childers, 2008) Generic laws applying to products and services apply to software but software defies attempts to fit it to one of these categories and prevailing law or legal precedents depend on this distinction. There is no major legislation in this domain and no major legal decisions have broadly addressed software liability. In 2009 the American Law Institute (ALI) approved the final draft of Principles of the Law of Software Contracts but the document testifies to the complexities of liability within software contracts. Thus, currently software developers have little to go on to anticipate their liability in the event that their software causes untoward consequences. Other than the broadest principles of legal liability, there is nothing certain about retrospective or prospective accountability. Yet computer scientists are well aware of the risks in software and they are pervasive and powerful.[5]

4. Anticipating Software Agents: The Counterarguments

The Concern is Premature

An important counter to the argument for a moral ontology is the claim that it is too early to interfere with the development of software agent technology. It is premature – some might say – to stop the use of what is a very useful metaphor, and, anyway, sooner or later – so the argument goes – issues of accountability will be addressed.

[5] Discussion of the risks from defects and failure of software can be found in The Risks Digest, a Forum On Risks To The Public In Computers And Related Systems; the Forum is an activity of the ACM Committee on Computers and Public Policy, moderated by Peter G. Neumann and found at: http://catless.ncl.ac.uk/Risks/

This is an important counter because it is, at least in certain respects, consistent with the descriptive implications of the model of technological development described above. The development of software agent technology, like all technological development, is not just socially embedded, the trajectory of development is fluid and contingent. This means that any number of factors may come into play at any time and it means that it is possible and perhaps likely that ethical concerns and issues of accountability will eventually arise and be addressed. At some point or another, the public, the law, or politics will respond to what is being developed. If software agents are deployed in situations in which they put individuals at undue risk or have harmful effects, there will be a response and practices of accountability will be worked out.

Although Wallach and Allen (2009) do not explicitly make this argument in *Moral Machines*, they implicitly adopt the strategy; they predict that in the short term "product safety laws will continue to be stretched to deal with artificial agents" and that dangerous practices will be dealt with first by the courts and later by legislation. They predict that "companies producing and utilizing intelligent machines will stress the difficulties in determining liability and encourage no-fault insurance policies" (p. 198). So Wallach and Allen seem to think that issues of accountability will arise and be addressed sooner or later. Technological development is a social endeavor embedded in society and so social and ethical norms are likely already in play or will come into play at some point or other.

However, the normative thrust of anticipatory ethics is that it is better to take ethical concerns into account *sooner rather than later*. The argument from moral ontology does precisely that; it proposes that while software agent technology is still "in the making," the endeavor to create it should not be understood as an endeavor to create discrete and autonomous entities, but rather to create sociotechnical systems, i.e., systems that work through the combination of human and non-human components. Such a re-conceptualization does not prevent explanations of software and hardware behavior, it prevents the complete deflection of human responsibility for harmful effects resulting from software behavior.

So, is it better to intervene in the construction of software agents early on or better to let the development process unfold without constraining the way the technology is conceptualized and understood? This is not a simple question. Addressing ethical issues early on has advantages; it also has disadvantages. Bringing in moral concerns early on may blunt efforts that would have been fruitful and might have turned out to be morally unproblematic. On the other hand, not bringing in these concerns early on might lead to technologies that run into trouble with the public, the law, or other stakeholders later on. The development process is *not* – as it is understood in technological determinism – a process in which a particular or pre-determined entity is waiting to "emerge." In the technologically deterministic view, all we need to do is make sure that the environment for development is unencumbered so that the predetermined entity can emerge quickly. On the contrary, bringing ethical issues into technological development has promise precisely because the development process is fluid and contingent.

Research and development take place in particular places, by particular individuals and groups and the when, where, how, and who is involved make a difference, just as the amount of funding and a myriad of other factors make a difference. Anticipatory ethics and the argument for a moral ontology should not, then, be seen as an encumbrance to development. Taking issues of accountability into account early on will make a difference and there may even be trade-offs in doing so, but so it is with all the factors that influence technological development.

The parallel between software agent technology and ethical issues in other domains of research is helpful here. Consider the moral constraint that has been institutionalized for medical research involving human subjects. When the requirement that scientists doing medical research obtain the informed consent of human subjects was first instituted, it was felt, by some scientists at least, to be a constraint on their research. Some thought that the science would proceed more quickly and more effectively if scientists didn't have to obtain the informed consent of human subjects. The requirement is now well accepted in medical research and it seems fair to say that what is discovered and developed in medical research involving human subjects is different than what might have been learned otherwise. Some would say the science is better for it; others might not agree. And, of course, it depends on what you mean by 'good' science.

The lesson, it would seem, for anticipatory ethics and the development of software agent technology is that conceptualizing the technology in ways that keep it tethered to the humans who design and use it will make a difference.

Software Agents *Are* Autonomous

But what about the counterclaim that software agents *are* autonomous? Isn't it a misconception to think that software agents are not autonomous? They operate independently – in space and time at least – from the humans who design and deploy them! Oddly, no one seems to claim exactly this; that is, no one seems to claim that the terms autonomy and autonomous agent have a singular, objectively definable meaning. These terms are being used in a wide variety of different ways. Some may deny my characterization of their use as metaphorical; others may embrace it. Computer scientists and software developers seem to use the terms as a metaphor that helps them describe how software agents operate in a simple way that does not require technical expertise. In some sense, computer scientists and software developers may not have a serious stake in the use of these terms since they understand what they are doing in a technical disciplinary discourse, be in programming, software development, or electrical engineering.

Whatever their interests, software developers have quite different stakes in the metaphor than do philosophers and cognitive scientists. In fact it would seem that philosophers and cognitive scientists use autonomy and autonomous in technical ways, technical in the sense that they are embedded in philosophical theory. One of the most influential pieces on the topic of artificial agents argues for autonomy as an operationally defined term tied to a particular level of abstraction (Floridi and Sanders, 2004). Others seem to make equivalency claims, that is, they claim that machines will be moral agents in the sense that they will have features that are equivalent to those of human moral agents. Of course, it is far from clear what constitutes equivalency in this context, and it is not at all clear what the significance is of an "entity" that has autonomy at a specific level of abstraction.

These different uses of "autonomy" and "autonomous" contribute to the creation of a rich discourse, a discourse with the potential for creative thinking and fruitful cross fertilization among disciplines and theoretical frameworks. On the one hand, the discourse often seems

confused and misleading as terms are moved from one context to the next, are used in widely different ways, and interlocutors often seem to miss one another. Do the issues need to be so complex? The discourse is complex in part because it involves an ontological struggle. The discourse is about what we are to 'make' of what is being developed.

Suppose we try to make it simpler. Why not understand "autonomous agents" simply to refer to things that operate on their own. It is easy enough to think about refrigerators, automatic pilot systems, and search engines (all of which today are in part constituted with software) as operating on their own. When pressed, however, it is not so easy. That is, when we try to justify this easy account, it becomes clear that we are engaged in ontology. To consider refrigerators, automatic pilots, search engines or software as entities – not even agents, just entities – is to engage in the mental exercise of separating them out from the complex sociotechnical systems of which they are a part.

Yes, my refrigerator maintains its internal temperature "on its own"; the thermostat signals other components to behave in certain ways that raise and lower the temperature. The problem with saying this is that my refrigerator only works insofar as it is plugged into an enormously complex power grid, a power grid that depends on many human and non-human components. In fact, the institutional arrangements constituting the power grid are an enormous feat of human social cooperation and interdependence. And, of course, my refrigerator only works as it is supposed to if I buy food that needs to be refrigerated, open and close the door to put the food in and take it out, pay my utility bill, and so on. Where does the entity that I call "my refrigerator" begin and end? It seems that we have decided (perhaps arbitrarily, perhaps not) to draw lines. We have conceptually (abstractly) decided we will count the rectangular chunk of plastic and metal that sits in my kitchen as "a refrigerator." We have decided to leave on the other side of the line (outside of the concept) the electrical grid to which it must be connected, all the people who maintain the electrical grid, and my behavior in opening and closing the door to put in and take out food. Yet my "refrigerator" does not work as a refrigerator unless all these other human and non-human components do their part. My refrigerator is a sociotechnical system, and the hunk of metal and plastic that I brought home from an appliance store is merely one component of that system.

It's the same for the automatic pilot. The automatic pilot works only insofar as it is delicately connected to other parts of the airplane. Whether the automatic pilot goes on only when human pilots flip a switch or goes on under specific conditions without human action is a design feature chosen by humans. What the automatic pilot does is the result of interactions among human and non-human parts. Where, again, does the automatic pilot begin and end? Is it and entity in itself or a component of an airplane? The airplane itself is a component in an enormously complex air transportation system. We draw lines; the lines specify what an automatic pilot "is"; what an airplane is, and so on. We choose, that is, what we will conceptualize as the part and what we will conceptualize as the whole.

It is the same for software. We say that a set of lines of code is software. Some call the software an agent. Of course, the software does nothing unless it is put into machines. Some call the software and hardware together an agent. Of course, humans had to create the software and humans had to build the machines and embed the software in the hardware. Humans turn on the machines, test and monitor their behavior. Where does the software agent (or the robot) begin and end? Some may argue that there is something different and distinctive about computers and software – they are not just chunks of metal and plastic; they are computational. This, they will say, makes them closer to or the same as humans. But this is another line drawing matter.

Lines do, of course, have to be drawn and they are drawn for various purposes. The thrust of the argument here is not to deny this, but to argue for bringing moral considerations into our line drawing. The argument for a moral ontology is an argument to draw the lines of "software agents" with an eye to keeping the locus on accountability with humans.

In this context two different sorts of dangers seem at issue. One has already been identified, that software entities might be conceptualized so as to suggest that no humans are accountable for the behavior of the software agents. The other is that theory- or context-dependent notions will move from one context to another in ways that cause confusion and are misleading. Grodzinsky, Miller, and Wolf (2008) illustrate this when they use the Floridi and Sanders (2004) notion of autonomy at different levels of abstraction. Focusing on tables used in programming, they show

how software can be autonomous to users (who cannot modify the table) while at the same time not autonomous to the designers (who can modify the table).

Thus, the argument for moral autonomy cannot be countered by the claim that software agents just are autonomous. Software agents are sociotechnical systems. They depend for their operation on being embedded in larger, more complicated systems that function by combinations of human and non-human or artifactual activity. Conceptualizing software agents as sociotechnical systems might well make a moral difference.

4. Conclusion

There are several lessons to be derived from the preceding analysis. The first is that anticipatory ethics is tied to a view of technological development as a fluid and contingent social endeavor and, hence, one that can be influenced by ethics and ethicists. This view has both descriptive and normative implications; it reveals both that moral notions and practices may have been at work influencing technological development in the past and influencing the development of existing technologies *and* that they can be more intentionally and effectively brought into play in the development of new technologies. In the case of software agent technology, the analysis focused on an argument for a moral ontology for software agents. The argument claims that we ought to conceptualize and understand software agents in ways they keep them tethered to the humans who design and deploy them, so as to avoid a deflection of human responsibility for the behavior of software agents. The claim is that an ontology of this kind will allow prospective and retrospective accountability to work. Although the argument illustrates the promise of anticipatory ethics, in the end it seems that anticipatory activity must be viewed cautiously. There is no certainty, for example, that eliminating the characterization of software agents as autonomous is the only or best way to address issues of accountability. Since technological development is fluid and contingent there are any number of ways that moral norms and practices can come into play. What is clear, nevertheless, is that technologies including software agents are sociotechnical systems and while we can conceptualize them in other ways, doing so can be misleading and may get in the way of human accountability.

References

Allen, C., Smit, I., and Wallach, W. (2005). Artificial morality: Top-down, bottom-up, and hybrid approaches. *Ethics and Information Technology* 7: 149-155.

Allen, C., Varaner, G. and Zinser, J. (2000). Prolegomena to any future artificial moral agent. *Journal of Experimental and Theoretical Artificial Intelligence* 12 (3): 251-261.

The American Law Institute (2009). Principles of the Law of Software Contracts, Proposed Final Draft March 16, 2009.

Anderson, M. and Anderson, S.L. (2006). Machine ethics? *IEEE Intelligent Systems* 21 (4):10-11.

Anderson, M. and Anderson, S.L. (2007). The status of machine ethics: a report from the AAAI symposium. *Minds and Machines* 17: 1-10.

Ballman, D.R. (1996). Commentary: Software tort: evaluating software harm by duty of function and form. *Connecticut Insurance Law Journal* 3.

Bowker, G. C. and Starr, S. L. (1999). *Sorting Things Out Classification and Its Consequences*. Cambridge, MA: The MIT Press.

Chien, S., Sherwood, R., Tran, D., Cichy, B., Rabideau, G., Castano, R. (2005). Lessons Learned from autonomous sciencecraft experiment. Paper presented at the Autonomous Agents and Multi-Agent Systems Conference, Utrecht, Netherlands.

Childers, S. J. (2008) Don't stop the music: no strict products liability for embedded software. U. Fla. J.L. & Pub. Pol'y 19

Fisher, E., Selin, C., and J. M. Wetmore. (2008). *The Yearbook of Nanotechnology in Society Vol. 1: Presenting Futures*. Springer.

Floridi, L. and Sanders, J.W. (2004). On the Morality of Artificial Agents. *Minds and Machines* 14 (3): 349-379.

Grodzinsky, F.S., Miller, K. W., and Wolf, M. J. (2008). "The ethics of designing artificial agents" *Ethics and Information Technology* 10: 115-121.

Hecht, M. (2005). Products Liability Issues for Embedded Software in Consumer Applications. *2005 IEEE Symposium on Product Safety Engineering* (Schaumburg, IL, Oct. 3–4, 2005).

Johnson, D. and Wetmore, J. M. (2008). STS and Ethics: Implications for Engineering Ethics. In E. Hackett, O. Amsterdamska, M. Lynch, and J. Wajcman (Eds.) *New Handbook of Science, and Technology Studies*. The MIT Press.

Johnson, D. and Miller, K. (2008). Un-making artificial moral agents. *Ethics and Information Technology* 10 2-3, 123-133.

Johnson D. and Powers, T.M. (2005b). Computer systems and responsibility: a normative look at technological complexity. *Ethics and Information Technology* 7 2, 99-107.

Johnson, D. (1985). *Computer Ethics* 1st edition, Prentice Hall, 1985.

Joy, B. (2000). Why the future doesn't need us. *Wired* 8 4, 238-262.

Lakoff, G., & Johnson, M. (1980). *Metaphors we live by.* Chicago: University of Chicago Press.

Lee, M. H. and Lacey, N.J. (2003). *Minds and Machines* 13: 367-395.

Moor, J. H. (2006). The nature, importance, and difficulty of machine ethics. *IEEE Intelligent Systems* 21 (4): 18-21.

Noorman, M. (2008). *Mind the Gap A Critique of Human/Technology Analogies in Artificial Agent Discourse*. Universitaire Pers Maastrict,

Rahwan, I., Sonenberg, L., Jennings, N., and McBurney, P. (2007) Stratum: A methodology for designing heuristic agent negotiation strategies. *Applied Artificial Intelligence* 21 6, 489-527

Scott, M.D. (2008). Tort liability for vendors of insecure software: has the time finally come? *Maryland Law Review* 67.

Terry, N.P. (2002). When the "machine that goes 'ping" causes harm: default torts rules and technologically-mediated health care injuries. *Saint Louis University Law Journal* 46.

Wallach, W. and Allen, C. (2009). *Moral machines Teaching robots right from wrong*. Oxford: Oxford University Press.

Zollers, F.E., McMullin, A. Hurd, S.N., and Shears, P. (2005). No more soft landings for software: liability for defects in an industry that has come of age. Santa Clara Computer & High Tech. L.J.