SOFTWARE ENGINEERING ETHICS IN A DIGITAL WORLD

Awais Rashid, *Lancaster University, UK*
John Weckert, *Charles Sturt University, Australia*
Richard Lucas, *Australian National University*

In ethics-aware software engineering, ethical considerations are explicitly taken into account across the software development life cycle and are an integral part of risk assessment and acceptance criteria.

The Internet has become pervasive in our day-to-day lives through the wide availability of broadband as well as its ubiquity due to the proliferation of mobile devices. This ubiquitous access to the Internet coupled with innovations such as social networking and Web 2.0 has given rise to the digital world phenomenon. Social networking sites such as Facebook, MySpace, and Twitter; virtual worlds such as SecondLife; and massively multiplayer online games such as the *World of Warcraft* have blurred the boundary between the online world and the physical world.

We live in an always-connected world, with an increasingly larger number of services accessible online. We conduct business, watch our favorite television programs, engage in interaction with friends and family, and make new social contacts online.

On the one hand, the digital world phenomenon improves accessibility of services and communal interaction, but on the other, it also raises several ethical issues relating to privacy, monitoring, data protection, and so on. Significantly, the features that make services or interactions in a digital world more attractive also have the potential to cause harm. The benefits of a digital world are obvious, but it also facilitates organized criminal activities such as pedophilia and terrorism, for example.

THE DUAL-USE DILEMMA

This problem of the same technology (or science) having the capacity for both good and ill has been called the dual-use dilemma.

Many, perhaps all, technologies can be used for both benefit and harm. The issue is not, in most cases, a call for a moratorium on the development of the technology but rather an attempt to find ways of maintaining the benefits while avoiding or minimizing the harm. It is worth noting that, at least in the US, the term "dual use" is used in two senses. In one use, it is not a dilemma. Any products, software, or technology that can be used for both civil and military purposes are considered to be dual-use items. In

 Published by the IEEE Computer Society

this sense, the term is used merely to show that one technology can have two distinct uses.

In the second sense, however, dual use does represent a dilemma. According to Seumas Miller and Michael Selgelid, the dual-use dilemma arises in the context of research in the biological and other sciences as a consequence of the fact that the same piece of scientific research sometimes has the potential to be used for harm as well as for good.[1]

Three cases help to illustrate the dual-use dilemma in the context of the digital world phenomenon.

Google Maps

Google has developed the technology (or used existing technology) to put 3D images of city streets on the Internet. On the benefits side, "Google promotes Street View as a useful tool for house hunting, planning holidays or working out where to meet friends. … combined with the Google Maps service, it was a 'reasonable proxy' for going there yourself" (http://technology.timesonline.co.uk/tol/news/tech_and_web/article6058197.ece). However, two types of concerns have been expressed. First, some claim that Street View is an invasion of privacy. Second, others contend that it aids terrorists, thieves, and robbers in finding their targets.

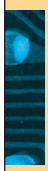
Chat rooms and peer-to-peer networks

Chat rooms, such as IRC and MSN, are prevalent on the Internet, as are peer-to-peer (P2P) networks such as Gnutella, BitTorrent, and EDonkey. Chat rooms allow geographically distant friends to interact with each other. However, pedophiles and terrorists also use them to plan their activities. In fact, chat rooms enable transformation of previously disorganized criminal activity into organized activity by providing a medium for criminals to interact around the world. Even more critically, most chat rooms allow users to assume any identity, hence facilitating criminals in their illegal activities. In many cases, pedophile rings use multiple online personas and share them so that various culprits can groom potential victims at different times.

Similarly, although P2P networks facilitate media sharing—for instance, the Norwegian broadcaster NRK offers popular shows via BitTorrent—they are also notorious for illegal sharing of copyrighted material or illegal pornographic material. A 2006 study discovered that 1.6 percent of searches and 2.4 percent of responses on the Gnutella P2P network relate to illegal sexual content.[2] Given the scale of the Gnutella network and its user base, and the fact that the study only covered a portion of the network, this implies that on the Gnutella network alone hundreds or thousands of searches for illegal sexual content occur every second. The study also revealed a trend toward core distributors of illegal sexual media: Of those users sharing illegal sexual content, 57 percent were solely devoted to such distribution, while half of the material shared by another 17 percent involved such content.

Facebook

The third, and perhaps most surprising, example is Facebook. This social networking website is undoubtedly a boon to many, assisting them in making new friends and keeping in touch with old ones, among other things. According to a recent report, however, "68 per cent of students who used Facebook had a significantly lower grade-point average than those who did not use the site" (www.theaustralian.news.com.au/story/0,24897,25325762-12332,00.html). Perhaps those who are more likely to receive low grades are also more likely to spend time on Facebook, but it is also possible that Facebook provides a distraction that affects students' grades. While the developers of Google Maps could have reasonably expected privacy issues to be raised, it is unlikely that those who developed Facebook could have anticipated the negative effect it has on some students.



The dual-use dilemma arises as a consequence of the fact that the same piece of scientific research sometimes has the potential to be used for harm as well as for good.

The dual-use dilemma is worth taking seriously when engineering software for the digital world not because we want to argue that technologies should not be developed if they can be used in harmful ways—probably few technologies would be permissible in that case—but because it focuses attention on the importance of considering harmful uses and ways of mitigating the harm. Further, it draws attention to questions of the responsibilities or accountability of those who develop the technologies.

In the only comprehensive study reported to date, Miller and Selgelid[1] state that the dual-use dilemma "is an ethical dilemma since it is about promoting good in the context of the potential for also causing harm." Further, they argue that the dilemma arises because researchers usually only intend their research to be used for good and it is the actions of others that often cause harm.

The same can probably be said of software engineers developing software systems for the digital world. They intend their products to be used for good, but others sometimes use them for ill. Considering software development in the context of the dual-use dilemma highlights this and raises the question of the software engineers' responsibilities in this area.

EXAMPLES OF ETHICALLY CHALLENGING CONTEXTS

The UK government's Communication Data Bill will require ISPs to log all e-mails and website accesses for all users for up to one year. The key motivation behind this is to improve public safety in a changing communication environment by aiding law enforcement agencies in tackling online criminal activity. Privacy campaigners, on the other hand, have raised serious concerns about privacy and personal freedoms.

A law recently passed in France will allow a special agency to cut off the Internet connection of users who download illegal content via P2P networks based on a three-strikes rule. Offending users will receive an e-mail after the first strike, get a letter upon the second offense, and have their connection cut off on the third. While the law safeguards digital copyrights, it also affects legitimate users of P2P content, for instance, the content provided by Norwegian broadcaster NRK, which uses BitTorrent to share material.

The High Court of Australia ruled that a defamation case could be heard in the State of Victoria even though the offending material was on a server in the US. A prominent Australian businessman, Joseph Gutnik, argued that the case should be heard in Victoria on the grounds that that is where the material was read and his reputation harmed. The company, financial publisher Dow Jones, argued that because the material was on a server in the US, that is where the trial should be held. The digital world not only spans the online-physical world divide but also crosses geographical and jurisdictional boundaries.

ETHICAL CHALLENGES FOR SOFTWARE ENGINEERS

The cases we have highlighted and the legislation described in the "Examples of Ethically Challenging Contexts" sidebar pose three key challenges for software engineers designing systems for the digital world.

Right to privacy versus the need to protect vulnerable user groups

There are increasing concerns over the use of the Internet to organize criminal activities such as the grooming of children by pedophiles, terrorists recruiting impressionable youths, and online credit card fraud, to name a few. This requires software-based solutions to aid police and other law enforcement agencies. The UK government's Communications Data Bill is one such effort. However, such "blanket monitoring" is seen by many as an invasion of the individual's right to privacy. Software engineers developing such systems must balance the need to safeguard the privacy of the individual against the need to protect vulnerable user groups—for example, protecting children from predatory advances.

Freedom of choice versus protection from harm

The ubiquitous access to the Internet has also provided opportunities for businesses and the general public to offer new services that should be designed to be attractive to the consumer. However, there are concerns that the features of online gambling sites, for instance, that make the gaming experience enjoyable are also the ones likely to cause harm to those at risk of gambling addiction. Similarly, there are concerns that online role-playing games or shoot 'em up simulations encourage violent behavior—a concern that has been raised for a long time. Software designs for such systems must account for both an individual's freedom to make choices and the notion of common good.

Context-dependent definitions of ethics

Ethical considerations are often subject to, and perhaps driven by, geographical, cultural, ethnic, religious, and even historical standpoints. Taking images of nude children as an example, there are cases that would obviously be classified as child abuse or exploitative while others would be considered innocuous—for instance, family photographs of children playing in the bath. However, there are cases where the boundary is not clear. For example, Tearney Gearon's photographs of her children nude on the beach wearing masks during a family holiday led to an intense debate about whether these were pornographic.

The cases where the ethical distinctions are clear are easy to tackle, but where the boundaries are fuzzy, subject to the interpretation of specific communities, software engineers face a nontrivial challenge. How can these often conflicting interpretations be effectively managed in the design of relevant software systems?

These challenges cannot be met by software engineers alone. Close collaboration is required with ethicists, lawmakers, and social scientists. Good engineering practices can solve some problems, but more diverse expertise is required for others. Perhaps this collaboration is yet another challenge.

ETHICS-AWARE SOFTWARE ENGINEERING

It is not clear that those developing software systems for the digital world in fact worry too much about any harmful consequences their products might have, but it is our contention that true professionals must be concerned about these issues. As such, we advocate the notion of ethics-aware software engineering, where ethical considerations are explicitly taken into account across the software development life cycle and are an integral part of risk assessment and acceptance criteria.

Consider again Google Street View. At one level, the benefits of Street View seem rather trivial. We can already go house hunting, plan holidays, and work out where to meet friends without too much trouble. Despite Google's claim that it "believed it was acting within the law and that the benefits of the service outweighed concerns about its intrusiveness. The reason we are doing it is because we think it has a lot of benefits," these benefits hardly out-

weigh invasions of privacy or aiding criminals or terrorists. More considerations than this, however, are relevant. In general, we do consider that more information is better than less. Even if the information provided by Street View is not particularly important in itself, unless compelling reasons exist to prohibit it, prohibition could set a dangerous precedent.

According to Miller and Selgelid,[1] ethical dilemmas should be solved not by simply weighing the potential benefits against the potential harms, but rather by finding other alternatives. This also applies to dual-use dilemmas. Alternative options should be found if possible, and this is where software engineers have a role to play. Can ways be found to change the software so that the benefits remain but the potential harm is avoided or minimized? At least some of the privacy concerns have been overcome by the blurring of faces and license plates on Street View, and people can have the image of their own house deleted quite quickly. Appropriate software engineering can mitigate the problems in cases like this.

The case of chat rooms and P2P file sharing, and Internet communications and online footprints of users in general, poses interesting challenges. On one end of the spectrum, we have the "blanket monitoring" approach of the UK Communications Data Bill or the complete ban as in the French Net Piracy law, while on the other extreme there are privacy and personal liberties advocates against any such measures.

In this case also, the Miller and Selgelid approach can aid software engineers. Instead of an all or nothing approach taken by lawmakers, specific criminal contexts can be tackled by solutions that address the need to protect vulnerable users while safeguarding the rights of the public in general.

An example of this is the UK Isis project (www.comp.lancs.ac.uk/isis). Jointly funded by the Engineering and Physical Sciences Research Council and the Economic and Social Sciences Research Council, the project is executed in collaboration with specialist law enforcement agencies and is developing tools for the analysis of online pedophile activity using natural-language processing techniques. However, instead of taking a blanket monitoring approach, the project is focusing on reducing the workload of law enforcement officers who have large amounts of text to process—for example, chat room logs—which requires numerous person-hours, causing law enforcement agencies to be consistently backlogged.

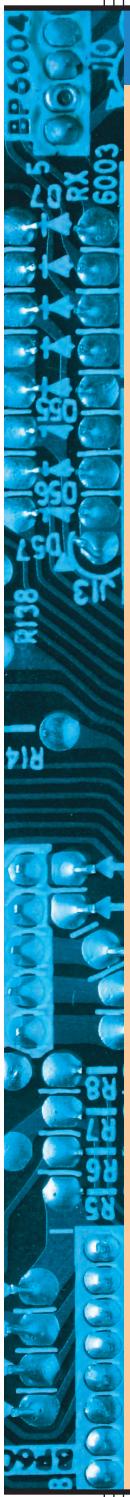
The Isis toolset flags potentially suspicious communication by detecting typical patterns of grooming behavior online and identifying when an adult is masquerading as a child. Significantly, however, the tools are being developed with the close involvement of ethics researchers, who are investigating the privacy concerns via a series of stakeholder meetings with industry experts, ISPs, parents, and young persons, to inform the development process and ensure that ethical considerations remain at the forefront of the developers' minds. At the same time, the tools are not aimed at being the judge, jury, and executioner. Instead, they provide additional evidence to allow trained law enforcement officers to make informed judgments about whether or not pedophile activity is taking place. Any evidence the tools gather still must comply with UK protocols and legal requirements for evidence efficacy.

A second aspect of the Isis project focuses on tackling the distribution of child abuse media via P2P networks by detecting core distributors—the people who have access to children and hence pose the biggest threat. The project is developing natural-language analysis techniques that can detect the specialist vocabulary used by pedophiles to share such material, remaining in step with changes to this vocabulary as it evolves (which is normally the case).[3] In this case, the ethical considerations have been narrowed down by focusing on a specific domain where features of the software (monitoring of P2P activity) that may otherwise raise wider ethical considerations can be geared toward a specific problem—identifying the vocabulary used by pedophiles to share material online and remaining in step with changes to that vocabulary.

The distinction between effect and use, even if not a sharp one, is useful when considering both the role and the responsibility of the software engineer.

In the Facebook case, it is not so obvious what a software engineer can do to mitigate the problem apart, perhaps, from limiting the time that anyone can use Facebook in one session. Clearly, Facebook poses a social problem but, in this case, it is arguably not an instance of the dual-use dilemma in the way that Street View is. In the Street View example, the issue is not that the technology has different effects on different people but that it can be used differently. Invading someone's privacy is different from house hunting.

The dual-use dilemma is concerned with uses rather than with effects. In the case of Facebook and similar systems, the dual-use dilemma becomes more pronounced when information shared on such systems can be exploited by criminal elements. This distinction between effect and use, even if not a sharp one, is useful when considering both the role and the responsibility of the software engineer. The dual-use dilemma suggests that software engineers have a duty to take care with respect to harmful uses of their systems. Harmful effects, as in the case of addictive software, may be more difficult to counter.



POINT/COUNTERPOINT

DO SOFTWARE ENGINEERING AND ETHICS MIX?

James Walkerdine, *Isis Forensics*

Most would agree that considering ethical issues during software engineering is something that all developers should strive for, and certainly in our organization this is encouraged. In reality, however, it is rarely considered in most software development projects. In our experience, this is not due to a lack of will on the part of the developers, but more often due to the practical challenges that make the consideration of ethical dimensions difficult.

Managing complexity

Software development is already a complex process, whether resolving conflicting requirements, developing efficient designs, or ensuring that an implementation is sufficiently verified and validated. The broad range of involved activities is typically performed under the constraints of limited resources or a tight deadline.

Given the fact that many software projects do overrun, it is not surprising that project managers are unwilling to add additional complexity to the task and run the risk of compromising other areas of the software development process. Unless ethical considerations are a core requirement for the development, managers are always likely to prioritize efforts to ensure that the functional aspects of the software are delivered and that quality levels are maintained.

Certainly, an approach that encourages ethical considerations within software engineering should be promoted; however, it also needs to be suitably supported. The Software Engineering Code of Ethics provided by the IEEE Computer Society and the ACM offers a foundation, but it is also too abstract when considering a development's finer-grained ethical implications.

For example, consider developers who need to make decisions about the interface for an online gambling game. On the one hand, it would improve the gaming experience, but on the other it might lead to potential risk for vulnerable users. Another example is developers who are building a social networking system and must make a tradeoff between ensuring their users are safe by monitoring activity, while at the same time maintaining user privacy. This requires software engineers who are trained and experienced in ethics and can convert ethical theory into a form that can be practically applied. However, such expertise is scarce, and the number of suitable training courses in this area is limited.

Practical challenges

The rapidly evolving nature of software engineering and software technologies can also pose practical challenges. Software systems are becoming increasingly distributed. With technologies such as service-oriented computing becoming popular, the nature of software is slowly moving away from single large-scale developments to developments that are based on compositions of software modules. These modules are often provided by third parties, can be geographically dispersed, and are typically beyond the control of the software developer who intends to use them.

These "systems of systems" raise several interesting issues when considering the ethical aspects of a development. For example, a developer may be required to use a remote third-party software module that is located in a country that has different legal and moral rules and thus has no say in how it operates. Or if a third party module used within a system fails and results in harm, who is deemed to be accountable for it?

Accountability

Accountability, in general, is a significant factor in ethical software engineering, especially when considering the legal environment. If the use of software results in damage or unethical behavior, the developers can be held accountable even if they did not cause the action. For example, YouTube has been held accountable for distributing illegal videos posted by its users.

Rather than encouraging the development of ethical safeguards within future software, the fear of litigation can instead make software developers purposely limit their accountability. This is perhaps best illustrated by peer-to-peer file-sharing software that has become increasingly decentralized and anonymous, so that developers, ISPs, and so on can deny knowledge of illegal material that might be shared using them. In this sort of environment, software developers are always likely to take the route that makes them susceptible to the least amount of risk and requires the least effort—even if this is not the most ethical solution.

Ethical software engineering is complex and raises many practical issues that software developers must consider. This and perhaps the fact that software development suffers from still being a fairly young discipline mean that developers remain reluctant to open the ethics' Pandora's box, even though it is something they may wish to pursue. Until hurdles such as the ones discussed here are removed or guidelines produced, software engineering ethics is still likely to be something that is practiced by the few rather than the majority. C

***James Walkerdine** is managing director of Isis Forensics, an IT forensics company that develops technical solutions to help protect the online activities of individuals and organizations. Contact him at j.walkerdine@isis-forensics.com.*

OUTLOOK AND ROADMAP

Balancing the ethical considerations when engineering software systems for the digital world is nontrivial. That is not to say that it is not the responsibility of software engineers. Software engineers play a fundamental role in the digital world and, as such, must consider the ethical implications of specific software design choices. However, software engineering researchers and software ethics experts must work together to advance the state of the art to provide ethics-aware software engineering practices.

Five key challenges emerge in this context that must be addressed over the next 5 to 10 years:

SORTING OUT THE EXPECTATIONS

Ruth Chadwick, *ESRC Centre for Economic and Social Aspects of Genomics (Cesagen)*

While interpretations of ethics are multiple, it seems clear that ethical issues arise where interests conflict. These can be the interests of different individuals or different groups. Sometimes the interest at stake is the same for all involved, but it cannot be satisfied for everybody—for example, where resources are scarce. At other times, the interests are different, and it is a question of which interests will and should prevail.

Identifying stakeholders

The first step is to identify the interests at stake. The task of ethics is then to provide approaches to resolving potential conflicts between them. It is not necessarily the case that there will be one clearly "right" solution, but the preferred answer must be at least defensible in terms that are recognizable as ethical, being supported by reasons and not, for example, pure assertion of power or blatant self-interest.

In the case of software engineering, the code of ethics provided by the IEEE Computer Society and the ACM identifies several different interests at stake, and the public interest is put first in the list. This is analogous to the codes of other professions that give priority to the interests of users or clients. However, there are issues about how the public interest is to be identified. A problem with this field, as in the case of other new or rapidly developing technologies, is that the technologies themselves have the potential to impact values.

Privacy is a case in point. While there are concerns about protection of privacy on the one hand, on the other what appears to be happening is that the concept and value of privacy is changing. The very fact that people want to engage with a social networking site where they reveal information about themselves in unprecedented ways suggests that notions of privacy are in flux. So there are difficult questions about what exactly is and should be protected when talk of privacy is concerned.

It is true that a code of ethics cannot, typically, be applied in a straightforward way to give answers. A code can perform an educational function so that people develop an "eye" for what would be appropriate in certain situations. In light of recent social networking developments, however, some mechanisms need to be put in place for thinking about what the public interest really is in this domain, such as user engagement in the development process. There are interesting debates to be had here to which user perspectives can make an important contribution. The example of the tradeoff between improving a gaming experience and increasing risk is a case in point.

Practical difficulties

Mention has been made of the practical difficulties of considering ethics, given the constraints of limited resources or a tight deadline. One answer to this is that such a point would be unlikely to impress if it were made by other professionals such as surgeons. However, the point made is more complex, because ethical requirements, it seems to be suggested, can be in tension with the software's functional and quality aspects. From some points of view, however, quality does have ethical aspects. Insofar as development of a product of poor quality has the potential to adversely affect the interests of users, it is an ethical issue. Again, the test is how interests are affected.

Risk aversion among software developers is a challenging problem, but not one that is specific to this area. It can be an issue also in other professions—for example, where medical ethics is interpreted as what needs to be done to avoid litigation. The practical questions, however, need to be distinguished from the ones that are genuinely ethical. The debates about what is an ethical conflict and how to set priorities is one issue; incentives and mechanisms for raising ethical awareness constitute another.

The realities of geographical dispersion offer a difficult challenge. In the context of globalization, we are seeing calls for harmonization of ethics and regulation in various contexts, such as international data sharing. But there are certainly problems in holding people accountable in situations of geographical dispersal. Here again, some distinctions are necessary. Actually, holding people accountable for harm they have caused, for example, in some legal process, is one thing. As far as allocating moral responsibility is concerned, much will depend on what type(s) of ethical theory are drawn upon—to what extent people should be held morally responsible for the wider consequences of their actions, or only for what they intend; the link between causal and moral responsibility where causal chains are very complex.

From an ethical point of view, the priority is to sort out what can reasonably be expected of an individual, and after that proceed to look at the possibilities of regulation and control.

In looking at the question of software engineering ethics, in addition to what has been suggested about engagement with users concerning what is in the public interest in this domain, there are conceptual questions to be asked about the role of the activity in question. In the case of professional ethics generally, professions have tended to be associated with some social good. It is not just a question of avoiding harm: Professions typically have a concept of the good or ideal of service to society. Clarity about that may help to further the ethical discussion. C

Ruth Chadwick is a distinguished research professor and director of Cesagen at Cardiff University. Contact her at chadwickR1@cf.ac.uk.

- *How to understand ethics requirements from different stakeholder standpoints.* Digital communities per se are mass-use systems. As such, the ethical standpoints of the various stakeholders involved (users, service providers, regulatory bodies, and so on) are subject to gender, generational, religious, and cultural variation. This, in turn, requires novel requirements engineering methodologies that enable analysis of the highly varied and inherently fuzzy ethical standpoints of the various stakeholders.
- *How to conduct tradeoff analysis involving ethics considerations when making architectural and design choices.* The various ethical considerations must be balanced against each other—for example, the right

to privacy versus the need to protect vulnerable user groups—as well as against other properties of the digital community such as providing enhanced online gambling experience versus providing protection from harm. This requires novel tradeoff analysis methods and decision-support systems that can guide architectural and design choices in the presence of conflicting ethical viewpoints and planned system features.
- *How to validate and test that the implemented system preserves ethical considerations and associated design tradeoffs*. Tracing the architectural and design choices influenced by ethical considerations to their sources is a nontrivial task. This requires understanding which ethical considerations are satisfied by specific design decisions and those that must be checked for and enforced during runtime, which requires novel testing and runtime constraint-checking mechanisms.

Some issues can be addressed by better design, some by better testing, some by legislation, and some by education of users.

- *How to incorporate ethics-aware practices in existing software engineering methodologies and processes*. Although existing software engineering methodologies and processes account for nonfunctional properties, ethical considerations are not at the forefront of such analyses. Incorporation of ethics-aware practices, therefore, requires software and business process improvement frameworks to facilitate an incremental evolution not only of the methodologies and processes but also of organizational practices and codes of conduct.
- *How to train software engineers in ethics-aware software design*. There is an international shortage of personnel trained in the ethical considerations that modern IT systems raise. Training of software engineers in ethics-aware software design is, therefore, essential to overcome this shortage. Any such training requires both adaptation of existing professional training programs and certifications offered by bodies such as the British Computer Society, the Institution of Engineering and Technology, the IEEE, and the ACM, as well as courses at the grassroots level in educational curricula.

IN THIS ISSUE

In attempting to address these challenges, it is useful to keep in mind that they involve various ethical issues. The examples discussed here demonstrate this, and it is reinforced in the articles included in this special issue. Some of the problems are primarily a result of different uses of a system—for example, whether for legitimate communication or for criminal activities. Others arise primarily through different effects on different people—for example, the effect that Facebook appears to have on some students. Problems that are a result of system malfunction comprise a third category. Some issues can be addressed by better design, some by better testing, some by legislation, and some by education of users.

There are also different emphases depending on whether the systems are primarily stand-alone in one organization or part of the Internet. Some uses and effects are more often problems on networks, particularly the Internet, while stand-alone systems are commonly, although not exclusively, the focus of malfunction problems.

In "Ensuring Trust, Privacy, and Etiquette in Web 2.0 Applications," Amela Karahasanovic and colleagues analyze three empirical studies to highlight trust, privacy, identity, user content control, green technology, and public welfare as requirements to be addressed by software designers. While designing Web 2.0 applications with these features or values in mind will not eliminate all harmful uses or effects, it should help to create a safer and more congenial environment for those who use the applications.

A core point in "The Precautionary Principle in a World of Digital Dependencies" by Wolter Pieters and André van Cleeff, which relates to the design issues discussed by Karahasanovic and colleagues, is that the design of technology can invite or inhibit particular uses. This is in the context of arguing that the precautionary principle should be applied to software design so that potentially harmful uses are discouraged and beneficial uses are encouraged. The authors develop this approach to help cope with problems that arise where there is no longer any "digital fence" surrounding organizations to protect their security.

Two articles in this special issue focus on software malfunction. "Social Impact of Information System Failures" by Tetsuo Tamai provides a detailed analysis of one Japanese case from which the author extracts several useful lessons for developers. Design is an obvious one and so is responsibility, an issue also addressed by Pieters and van Cleeff.

Responsibility is also a theme in "The Public Is the Priority: Making Decisions Using the Software Engineering Code of Ethics" by Donald Gotterbarn and Keith W. Miller. These authors use case studies to demonstrate how a code of ethics, specifically the Software Engineering Code of Ethics, can be used to aid decision making in software engineering.

Finally, in the "Point/Counterpoint" sidebar, we include a short debate between James Walkerdine, managing director of Isis Forensics, an Internet forensics company, and Ruth Chadwick, distinguished research professor at Cardiff

University. They discuss whether software engineering and ethics can mix, make several points that help to clarify some ethical concerns, and expose some of the difficulties in overcoming them.

Ethics has long been recognized as important in software engineering. The growth and development in social networking, ambient computing, cloud computing, outsourcing, and so on has only increased its importance in the digital world.

References

1. S. Miller and M. Selgelid, *Ethical and Philosophical Consideration of the Dual-Use Dilemma in the Biological Sciences*, Springer, 2009.
2. D. Hughes et al., "Peer-to-Peer: Is Deviant Behavior the Norm on P2P File-Sharing Networks?" *IEEE Distributed Systems Online*, vol. 7, no. 2, 2006.
3. D. Hughes et al., "Supporting Law Enforcement in Digital Communities through Natural Language Analysis," *Proc. Int'l Workshop Computational Forensics* (IWCF 08), Nat'l Academy of Sciences, 2008, pp. 122-134.

Awais Rashid *is a professor of software engineering at Lancaster University, UK. His research interests are software for crime prevention, advanced software modularity, and requirements engineering. He received a PhD in computer science from Lancaster University. He is a member of the IEEE, the IEEE Computer Society, and ACM Sigplan. Contact him at awais@comp.lancs.ac.uk.*

John Weckert *is a Professorial Fellow in the Centre for Applied Philosophy and Public Ethics at Charles Sturt University, Australia. His research interests are in the ethics of new technologies, particularly information and communication technology, nanotechnology, and synthetic biology. Weckert received a PhD in philosophy from the University of Melbourne. He is a member of the Australian Computer Society. Contact him at jweckert@csu.edu.au.*

Richard Lucas *is a Research Fellow in the Centre for Applied Philosophy and Public Ethics at the Australian National University. His research interests include ethics and professionalism in ICT, philosophy and ethics in the context of science fiction, and the conjunction of ethics, technology, and sport. Lucas received a PhD in philosophy from Charles Sturt University. He is a member of the Australian Computer Society. Contact him at richard.lucas@anu.edu.au.*