A Roadmap for Ethics-Aware Software Engineering

Fatma Başak Aydemir  
Utrecht University  
The Netherlands  
f.b.aydemir@uu.nl

Fabiano Dalpiaz  
Utrecht University  
The Netherlands  
f.dalpiaz@uu.nl

ABSTRACT

Today’s software is highly intertwined with our lives, and it possesses an increasing ability to act and influence us. Besides the renown example of self-driving cars and their harmfulness, more mundane software such as social networks can introduce bias, break privacy preferences, lead to digital addiction, etc. Additionally, the SE process itself is highly affected by ethical issues, such as diversity and business ethics. This paper introduces ethics-aware SE, a version of SE in which the ethical values of the stakeholders (including developers and users) are captured, analyzed, and reflected in software specifications and in the SE processes. We propose an analytical framework that assists stakeholders in analyzing ethical issues in terms of subject (software artifact or SE process), relevant value (diversity, privacy, autonomy, ...), and threatened object (user, developer, ...). We also define a roadmap that illustrates the necessary steps for the SE research and practice community in order to fully realize ethics-aware SE.

KEYWORDS

ethics, ethical software engineering, ethical design, ethics compliance, requirements engineering

ACM Reference Format:


1 INTRODUCTION

Humans are the centerpiece of most software engineering (SE) phases, including requirements elicitation and specification, development, verification and validation, maintenance and evolution. Moreover, software projects and products arise to fulfill stakeholders’ needs and wishes, and the resulting systems are utilized (directly or indirectly) by human users. Finally, the pervasiveness of software and hardware in our lives offer new possibilities that transform our lives and can even enable enhancing; think of smart watches, sleep therapy apps, autonomous vehicles, and the like. Researchers have coined different terms to study the many facets of this interplay, including social software engineering [1], socio-technical systems (engineering) [2–4], and social computing [5, 6].

Existing codes of conducts and ethics standards (e.g., ACM’s Software Engineering Code of Ethics [16]) provide high-level guidance to software professionals on one particular aspect, that is, professional ethics. However, they do not capture the many other facets of ethics such as those in the questions above. Recent initiatives were established to cope with ethical issues related to the rise of artificial intelligence (e.g., self-driving cars and war robots), yet we lack such initiatives for the rest of the software products and we miss an overall framework for understanding and managing ethical concerns throughout SE, operation, and maintenance. The SE world

For the selection and purchase of non-software goods and services, people are increasingly taking into serious account ethical aspects [7, 8] besides the more traditional criterion of cost-effectiveness. People are rightfully concerned about how goods are produced (e.g., do subcontractors use child labor? Is fair payment guaranteed?), what is the cost for the environment (e.g., what is the carbon footprint? Are the raw materials obtained without poisoning or destroying natural ecosystems?), and what substances goods contain that may have negative side-effects on people’s health (e.g., are there any pesticides in the food? Do any of the ingredients lead to some kind of addiction?).

We advocate that similar questions should be asked for the software products we use, and ethics should play a role in the decisions of end-users, customers, industry professionals, and companies regarding the production and use of software. There are numerous ethics concerns, each giving birth to justified questions that one may pose about a software product, including:

- What are the working conditions of software professionals (think of the use of outsourcing in software development)?
- What are the costs of developing and running specific software on the environment [9, 10]?
- Does the system encourage unhealthy behavior that leads to digital addiction [11]?
- To what extent is the software aligned with the privacy preferences of individual users [12]?
- Are gender and diversity issues considered by the software producing organization?
- Is the system fair [13] and does it avoid discrimination of users with a given gender, race, age range, or income?
- To what extent is software evolution driven by a democratic need analysis of the crowd of users [14]?

In summary, we put forward that the decisions of humans regarding a given software—Should a user buy a certain software? Should a developer accept a job at that company? Should a company switch to a different information system (provider)?—should be based on ethics concerns. These aspects add up to the most basic criteria that determine suitability, i.e., functional requirements as well as quality requirements [15], and cost considerations.
is lagging behind: for physical goods and services, standardized artifacts are arising such as the Common Good balance sheet [17] and frameworks for making ethical decisions in business [18].

In this paper, we introduce ethics-aware SE, the ethics-aware version of SE that fosters the elicitation and analysis of stakeholders’ values and their inclusion both in the socio-technical process through which software is built, and in the resulting software product. We present ethics-aware SE by means of an analytical framework, which assists stakeholders in analyzing ethical issues in terms of the subject (software artifact or SE process), the ethics value to be preserved (diversity, privacy, autonomy, eco-sustainability, discrimination, etc.), and the object that is threatened when compliance with the value is not ensured (user, developer, etc.).

Our goal is to foster the community to fill the existing gaps: current codes of conducts lack of breadth and depth, there are no languages for stakeholders to express their ethical standpoints, and software organizations do not reveal details on ethical issues. As a result, it is practically impossible for a human to check her/his ethical alignment with a given software product or its producing organization. Notably, we do not try to promote one or another ethical concern; our main aim is to support transparency.

Specifically, we make three contributions toward to the establishment of a body of work on ethics in SE:

- We review white and gray literature about ethics & SE that motivates the creation of systematic approaches for making progress in the field (Sec. 2);
- We introduce our vision on ethics-aware SE that includes a framework for understanding and analyzing ethical aspects (Sec. 3);
- We define a research roadmap to realize our vision that shows the milestones to achieve in order to move from today’s practice to a fully ethics-aware practice of SE (Sec. 4).

Finally, we conclude the paper with a discussion of the key message and with an outline of our own future work (Sec. 5).

2 BACKGROUND

Our research problem is related to professional and applied ethics in SE, normative multiagent and sociotechnical systems, compliance of requirements, and requirements specialized for certain good conducts such as privacy and transparency.

2.1 Related Work

Pierce and Henry [19] argue that proper ethical behavior of software professionals is influenced by the individual’s own code, the code that exists in the work place, and the formal code of conduct and discusses the degree of influence of these three codes. Hall [20] discusses good professional conducts for software engineers. Gotterbarn [21] provides a list of ethical issues concerning software engineering. Allhoff [22] approaches to applied ethics from a science and engineering perspective. Software Engineering Code of Ethics and Professional Practice is the output of the joint task force by ACM and IEEE-CS [16].

The logical basis of ethics has been discussed by Prior [23], Nowell-Smith and Lemmon [24], and Rickman [25]. Later, Anderson revisited the discussion [26]. Deontic logic deals with modeling and reasoning on obligation, duty, permission, right and other related normative concepts [27].

Boella et al. [28] introduce normative multiagent systems (MASs) where norms regulate and structure social order among agents. Singh [29] proposes an approach for governing sociotechnical system based on norms, where stakeholder needs are taken into consideration for governance decisions.

Compliance is a topic well explored in requirements engineering (RE). Breaux et al. [30] check whether a set of requirements is compliant with the rights and obligations extracted from legal documents. Darimont et al. [31] use goal-oriented RE method KAOS [32] to model regulations.

Chung et al. propose a framework to capture and analyze non-functional requirements [33]. Several non-functional requirements have been specifically studied in requirements engineering. For example, Hosseini et al. [34] propose a requirements modeling language to model transparency. Several scholars worked on privacy [35, 36].

Other areas of SE has also studied specific good behavior. Gallotra et al. introduce fairness testing techniques that are able to identify whether and when software exhibits discriminatory behavior. Shenoy and Erratta [37] discuss how software can be developed in an environmental friendly fashion. Winschier and Paterson [38] promote re-use to foster sustainability. Keith et al. collect end user values regarding privacy for quantified-self applications [39]. Collins et al. [40] discuss ethical considerations regarding the delivery of software products. Hyman [41] argues the role of software to ensure fairness among crowd-sourced workers.

2.2 Gray literature discussions over SE ethics

Ethics issues in tech industry, software development processes, and software artifacts have been discussed also in the gray literature. One popular topic is unethical user interface design patterns, referred as “dark patterns”. Dark Patterns web site [42] provides a pattern library for unethical user interface design and examples from real web sites and applications. Mesibov [43] considers dark patterns as a way to learn good design. Brigham [44] discusses honesty and deception over user interface design, and ends with success story where Audible\(^1\) changed its design based on negative feedbacks due to dark patterns. Dark patterns are also banned by EU in the updated European Consumer Rights Directive [45]. Dark patterns are not only used to deceive consumers but also to encourage addiction [46].

The variety of the ethics issues in SE can be seen in the content of various documents. Sgroi [47] points out topics such as software bugs, open source code, piracy, intellectual rights, and liability. Wayner [48] lists 12 ethical dilemmas for software professional which include privacy of users as well as software professionals, business ethics, open source software, and bug fixes. Pollice [49] indicates the lack of awareness on ethics issues in software development and discusses privacy, trust, encryption, intellectual property, and freedom of speech.

In 2017, diversity has been discussed widely due to two major incidents. The first is the reflection of a female engineer on her employment at Uber [50]. The company has also been criticised.
for its lack of business ethics [51]. The second is an anti-diversity manifesto written by an engineer employed at Google, who later got fired over it [52].

Another popular topic of discussion is the ethics of intelligent systems such as self-driving vehicles [53, 54] and unmanned arms [55, 56]. A list of 20 guidelines for ethics of automated driving is issued by German ethics commission [57]. On the other hand, crowd-based solutions are adopted to solve social dilemmas regarding self-driving vehicles [14, 58].

3 OUR VISION OF ETHICS-AWARE SE

Ethics in SE has started gaining more attention. Initial discussions [59] focus on professional ethics which later lead to IEEE/ACM code of conduct for SE. However, with the rapid integration of software in the lives of individuals as well as in the processes of organizations, software artifacts and software development processes are subject to more numerous and more heterogeneous ethical concerns than professional ethics alone.

A code of conduct is a set of guidelines for its intended audience. In the case of SE, a code of conduct for software engineers is a set of rules for the software professionals to follow. However, many other stakeholders are involved in the development, release, and usage of software including customers, software development organizations, and users. There is therefore a need for tools and methods to reflect, implement, and validate the ethical values of these parties, who are not disciplined by the existing code of conducts.

3.1 The Principle of Harmony

Our goal is to lay the foundations for ethics-aware software engineering to establish harmony with respect to ethical values and behavior in SE. This includes (i) creating awareness of ethical issues regarding software artifacts and software development processes, (ii) providing stakeholders with tools to let them articulate their ethical values, and (iii) building methods to monitor, verify, and validate SE processes and software artifacts to ensure compliance with those values. Our vision for the long term is to help policy makers to create regulations for software-engineering processes and software artifacts based on the ethical values and the corresponding “right” behavior identified by the stakeholders. Different stakeholders have distinct ways of interpreting such harmony. Users want to be able to articulate their ethics requirements so that they can choose the software that better aligns with their values. For SE professionals, harmony means that they can also state their ethics values and work for organizations that respect them. Software development organizations declare their ethics principles, and reflect these principles in the software engineering processes they follow and in the software artifacts they build.

3.2 The Ethics-Aware SE Method

Figure 1 illustrates our high-level method for ethics-aware SE. It involves all stakeholders regarding software development, from customer and software users to developers, organizations, and others. The method is agnostic of specific development paradigms; its activities can be embedded in any method including agile, waterfall, iterative, and V-model. Five distinct phases of the method are shown in a cycle which indicates that being ethics-aware is not a one-shot activity, but rather requires continuous effort as long as the software artifact is being built, maintained, and used. Figure 1 also lists four distinct enablers of being ethics-aware in software engineering practices (E0–E3).

Figure 1: Method for Ethics-Aware Software Engineering

Articulation This activity involves eliciting, modeling, and analyzing ethics values, what we call ethics requirements for software artifacts and their development processes. One source of such requirements is, naturally, the traditional stakeholders of software artifacts, such as customers, users, legislative organizations.

However, the voices of SE professionals also need to be heard to ensure the harmony between them and the organizations they work for, as well as the products they build. Similarly, software organizations should declare their ethics values to ensure that their processes and products are aligned with their values in the subsequent activities. Articulation can be included in the traditional requirements engineering activities for software stakeholders. In current practices, some ethical concerns such as privacy are already treated as quality requirements [12, 35, 36, 60].

We aim to bring other ethical issues to attention of SE stakeholders. Professionals may articulate their ethics values when they are hired and update their statement any time. Organizations may state and update their ethics values at any time as well.

The subject of ethics requirements could be either the software artifact or the software development process, each requirement concerns an ethics value, and a set of objects that are affected in case of violation or satisfaction. Ethics values can be high level such as “diversity” however related requirements can be refined to be more specific such as “user interfaces shall attract both female and male users equally”. We have provided some example issues in Table 1. One can argue about the overlap among certain items. For example, transparency, diversity, and common good are all related

### Table 1

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Specification</td>
<td>Articulation</td>
<td>Implementation</td>
<td>Verification</td>
</tr>
</tbody>
</table>

Although legislative bodies define right and wrong in terms of norms that should be obeyed by everybody, ethical values are personal and individual. When we say “right”, we mean “in compliance with a stakeholder’s preferences”.
to fairness, which is yet another issue. Ensuring the completeness of such a table is beyond the scope of our paper, and in fact, it is one of our long term objectives to create a catalog of ethics issues regarding software products and processes to support stakeholders.

**Specification** This activity fills the gap between the ethics requirements and the corresponding functional and quality requirements on software artifacts and processes. This step requires transferable knowledge from the discipline of ethics to SE. For example, if the customer states that “the system shall be gender neutral”, the specialist must be able to identify the related use cases, and the compliant system behavior.

**Table 1: Examples of ethics issues in software engineering**

<table>
<thead>
<tr>
<th>Value</th>
<th>Issue</th>
</tr>
</thead>
<tbody>
<tr>
<td>Privacy</td>
<td>Handling, storing, sharing user data only under the circumstances and the purposes that the user sets</td>
</tr>
<tr>
<td>Sustainability</td>
<td>Energy consumption of the software artifact, caring about energy throughout the SE process and in the documentation</td>
</tr>
<tr>
<td>Transparency</td>
<td>Transparent decision-making procedures of intelligent systems, publicly available ethics policies by software development organizations</td>
</tr>
<tr>
<td>Diversity</td>
<td>Gender, race, and age distribution of professionals in a development team</td>
</tr>
<tr>
<td>Work ethics</td>
<td>Decisions on which bugs to fix and how quickly, ensuring quality of the code before release</td>
</tr>
<tr>
<td>Business ethics</td>
<td>Informing users of a changed business model, including revenue models</td>
</tr>
<tr>
<td>Accountability</td>
<td>Who should be held responsible for the harm caused by software?</td>
</tr>
<tr>
<td>Dependability</td>
<td>Decision to maintain and/or keep a software product in the market</td>
</tr>
<tr>
<td>Common goods</td>
<td>Contributing to, using, promoting open source software</td>
</tr>
</tbody>
</table>

**Implementation** From the ethics-aware software engineering perspective, the implementation activity not only refers to writing code and documents for software products, but also building software development processes based on the ethics specification. A key challenge is to prevent the way-too-common situation in which quality requirements are disregarded because of time-to-market constraints.

**Verification** These activities involve both the software artifact and the software development processes, and require organizations to put in place appropriate means to continuously check that the software is being built according to the ethics specification.

**Validation** These activities check whether the software artifact and the development processes the ethics requirements imposed by the stakeholders at the beginning of the process. The outcomes of validation may result in a certificate that testifies how well a software system aligns with its intended ethics requirements, and explains deviations, if any.

We have identified four enablers for the adoption of ethics-aware SE by a range of stakeholders including users with no technical background to software development organizations:

**E0: Ethics knowledge.** First and foremost, this is required for each step involved in the method. Users, professionals, and organizations should know about ethical issues related to SE to decide whether they care about these issues, and what is their desired “right” behavior for each issue. Especially for the articulation and specification steps, professional knowledge on both ethics and software engineering is required to identify use and misuse cases and translate ethical concerns into concrete system behavior.

**E1: Awareness.** All stakeholders must be aware of the ethical issues and their consequences within the realm of SE. Issues are discussed and treated only when there is a wide awareness about them. For example, there has been a great awareness on privacy issues in software, as a result companies have provided several tools and customization options for regulating user privacy, and legislative institutions have regulated the specific issues. For the issues where there has been less awareness in the public such as sustainability in SE, we have not seen such customization options from companies and regulations from governments.

**E2: Conscious valuing.** Awareness alone is not enough. The stakeholders must develop a conscious value for issue in discussion. If a certain issue, such as accountability of the software engineers for algorithmic bias does not have a value for the public, they will not articulate their requirements for the issue for they do not care about it. However, this situation might change depending on the course of events, for example allegations on social media for biased timeline presentations might create first awareness (E1) and then conscious value (E2) for an ethical issue in software engineering. Once there is enough interest for an issue, stakeholders articulate their ethics requirements and specialists build the specifications for them.

**E3: Transparency.** In order to ensure that the behavior of the artifact as well as software development processes follow the specification, transparency is key. Software development processes and the behavior of the artifact must be transparent so that they can be validated against the ethics requirements.

Aligned with Footnote 2, we do not impose a right or wrong behavior for SE. We advocate for tools and methods to enable stakeholders to be aware of ethics issues, articulate their requirements, and use ethics as one of the criteria for choosing appropriate software. Our vision is that, in the long term, “right behavior” will emerge based on the ethics requirements of stakeholders and will be regulated so that the products and the processes must comply with it.

**4 A ROADMAP FOR THE FIELD: RESEARCH QUESTIONS**

We present the research questions to realize our research vision presented in Section 3, describe the artifacts we aim to produce, and discuss our plans for the validation of the framework. Table 2 lists the artifacts to be produced upon answering each research question.

**RQ1:** What are the relevant ethics issues for software engineering? There is no catalog of ethics issues regarding SE. Our first
Table 2: Artifacts of the Ethics-Aware SE Framework

<table>
<thead>
<tr>
<th>RQ</th>
<th>Artifact</th>
</tr>
</thead>
</table>
| RQ1 | i. Catalog of ethics issues in SE  
     | ii. Crowd-driven ethics standards and code of conduct  
     | iii. Elicitation methods for ethics requirements |
| RQ2 | iv. Language for expressing ethics requirements  
     | v. NLP tool for deriving ethics requirements from natural language text |
| RQ3 | vi. Visual notation for the ethics requirements language |
| RQ4 | vii. Analysis tool kit for ethics statements |
| RQ5 | viii. Design patterns to map ethics requirements to ethics design  
     | ix. Traceability techniques for ethics requirements |
| RQ6 | x. Verification techniques for ethics requirements |
| RQ7 | xi. Ethics acceptance tests |

Research question aims to identify a classification of ethics issues in SE. Our starting point is ethics issues in engineering in general, and our next step is to complement these issues by interviewing and surveying software engineering professionals to discover issues related to professional and business ethics. To maximize reach, we plan to build a crowd-sourcing platform to collect, categorize, and validate current ethical issues for SE. The crowd participants are SE professionals, company representatives, and other professional stakeholders. Their tasks include entering ethics issues and dilemmas they face during software development. As we collect these issues, the crowd will cross-validate the collected issues. The wisdom of the crowd is also used to collect the stance of the individuals when faced with these ethical issues to generate an ethical behavior catalog. In this case, the participants are presented short, realistic scenarios, and answer questions such as “What would you do?” or “What would happen in your company?” given that scenario. The output of this effort is then used to create comprehensive ethics standards and a code of conduct including realistic cases and possible course of actions. By creating the standards in collaboration with the crowd, we intend to align the standards with ethical values and with the current practices, and create awareness among software engineering stakeholders regarding ethics issues in software engineering.

We also envision another crowd-based platform for end users to collect their ethics requirements for various types of software products and processes. However, the articulation of ethics requirements in such setting requires a certain level of technical and philosophical knowledge. So, we plan to complement it with transdisciplinary research with sociology and philosophy researchers to identify ethics issues that rise due to the integration of software in our daily lives. It is also important to be well-informed of the latest events that involves ethical dilemmas and decisions and as well as personal experience reports published as blog posts, social media discussions and other grey literature. Surveying such resources will enable us to gather and process more information regarding ethics requirements, and what is considered “right” and “wrong” behavior.

Once we have the catalogs of ethics issues in SE, their triggering events, sets of possible right and wrong behaviors identified by the crowd, our aim is to turn this information into tools that support requirements elicitation. The catalogs can serve as checklists for requirements analysts to inquire after, but also educated users can benefit from them when making their decisions regarding software. Catalogs can also be used as an anchor for transparency, and organizations use them as guidelines to reveal their ethical stance for certain issues and disclose organization policies. Building appropriate elicitation techniques, providing requirements analysts with adequate tools to capture ethics requirements are other challenges that need to be addressed to achieve ethics-aware SE.

**RQ2: What are adequate modeling primitives to capture ethics requirements?** Conceptual models are useful to abstract, comprehend and communicate information. In case of ethics requirements such information may include the mental model of stakeholders, domain models to understand the field of ethics as a whole, business process models to animate the existing practices with ethics requirements, and possibly many others to represent different perspectives on ethics & SE. Our aim is to develop a language that effectively captures ethics requirements based on the information collected for RQ1.

Considering the amount of information that would be gathered from interviews and surveys employing natural language processing (NLP) techniques to automatically draft ethics requirements from various sources would greatly support specialists.

**RQ3: Which visual notations can help capture ethics requirements?** Ethics is a gray area. Various stakeholders would probably have different perspectives for the same ethics issues regarding the same software product and development processes; therefore, their requirements would differ. It is difficult to communicate all these differing perspectives and possible conflicts in textual format. Visual notations communicate information better than text, so our plan is to devise a visual notation to capture ethics requirements. A key requirement for the notation is to be usable by any type of stakeholder, including those who do not have a strong ethics or SE backgrounds.

**RQ4: How to analyze the interplay of ethics requirements and other requirements?** One of our research goals is to establish ethical harmony among stakeholders involved in software engineering. The outputs of RQ2 and RQ3 enable us to capture and communicate the ethics requirements of the stakeholders, but do not provide any resolution strategy in case conflicts occur. RQ4 focuses on analyzing requirements to identify conflict not only among ethics requirements, but also with other functional and quality requirements. Once the conflicts are identified, they may be eliminated through negotiation and relaxation techniques, or decisions can be made by trade-off analysis, and other decision-making techniques.

**RQ5: Which techniques can help trace ethics requirements?** A significant research challenge is to refine ethics requirements and relate them to more concrete functional and quality requirements. For example, “the software system shall be accessible” is a high-level statement. It is the responsibility of the analysts to identify with which user interfaces, features, design choices this...
We have proposed a holistic ethics framework for the SE discipline. Our vision is that each category of ethics requirements that is outputted from RQ1 may require their own specialist for this mapping between ethics requirements to functional and quality requirements. Still, over time, ethical design patterns will emerge and become prevalent as a result of efforts to realize ethics-aware SE. The mapping from ethics requirements to other artifacts of the software development processes is necessary but not sufficient. It is still important to trace the initial requirements to later understand to which degree they are satisfied along the software development processes.

**RQ6:** How can we verify software artifacts and processes against ethics specifications? This research question concerns testing the behavior of the software artifacts and development processes against ethics specification. How can we construct acceptance tests? What are the unit tests for ethics? Can we generalize and re-use such tests for different software products? Are existing testing techniques applicable for verifying ethics requirements? A good starting point for the value of *fairness* is the work by Galhotra et al. [13], which proposes testing techniques that detect discriminatory behavior exhibited by a system.

**RQ7:** How can we validate software against ethics requirements? Similar challenges exist for software validation. In order to check whether the software product is ethically the right product of all stakeholders we need have a strong comprehension of the requirements through the articulation of the stakeholders, and all the following steps must be successfully completed which demonstrates the cyclic nature of the method proposed in Figure 1. Ethics acceptance tests for customers are significant artifacts for this phase.

5 DISCUSSION AND CONCLUSIONS

We have proposed a holistic ethics framework for the SE discipline. Our framework aims to capture and analyze ethics requirements to achieve *ethical harmony* that is reflected in software artifacts and development processes. Our vision is to fully realize such framework by building its components, starting from the creation of extensive catalogs of ethics issues, requirements, and good behavior for individuals, artifacts, and organizations.

The key novelty of the framework lies within the problem we investigate. Although ethics discussions related to software systems have gained popularity due to the rapid rise of autonomous intelligent systems, the discussions are shaped around the ethics of artificial intelligence and neglect the process of creating such systems, including the individual SE professionals and the followed software development methods. Other ethics issues such as sustainable software systems and green IT, as well as privacy have also received the attention of scholars. Our aim to provide an *umbrella ethics framework* that educates end users as well professionals on ethics, and that provides alternative solutions based on their individual and organizational values as well as their requirements. Our goal is not to declare right and wrong behavior ourselves, but rather to create awareness on the existence of ethics issues and possible behaviors that respect ethics requirements of stakeholders involved in building and using software.

We are inspired by other phenomena that are occurring in our society. For example, the responsible and sustainable food and clothing initiatives, where not only responsible businesses but also regular companies shift towards good behavior due to public awareness, conscious valuing, and demand of ethical behavior. In this light, we would also like to stress the importance of ethics education for SE students and professionals to promote right behavior.

Our immediate future work is to collect input from software professionals. Our goal is to expand the list of ethics issues and to create a classification to use as the basic input for further elicitation and prioritization via crowd-sourcing. Another input we are working on is the creation of short scenarios to provide as examples to the SE crowd to collect their preferred behavior. Also, we are refining the modeling primitives for capturing ethics statements and requirements. Our next publications planned on ethics-aware SE will present the artifacts produced for RQ1 and RQ3.

We plan to conduct research in close collaboration with software professionals, SE researchers, and other disciplines such as sociology, philosophy, and education to ensure relevance and usefulness. As we obtain answers to our research questions through our developed artifacts, our results will be disseminated to the SE community in order to create awareness on ethics issues in SE and foster collaborations.

Ultimately, we hope that our work will trigger the attention of SE community and that ethics will become a first-class aspect to inform decisions on engineering new software systems, their evolution, the selection among alternative software systems, and the decision of which software development company to rely on or to work for.

REFERENCES


