Incorporating Ethics in Software Engineering: Challenges and Opportunities

Authors Anonymised

***Abstract*—Ethics is recognised as an important concern in the development and operation of software systems. While there are codes of ethics and sets of ethical principles available to software professionals, there is a lack of tool and process support for systematic ethical deliberation at most stages of the software lifecycle. To create and deploy ethical software, it is vital that ethical concerns of software systems are reflected in their artefacts, such as requirements, software architecture, code and test suites, and that software professionals are supported in considering the ethical as well as technical consequences of their decisions. This paper reports on some early work in identifying the challenges of ethical decision making and opportunities for addressing these challenges in the context of software engineering.**

***Index Terms*—software ethics, ethical deliberation, software engineering**

I. Introduction

As software becomes more pervasive and critical for the functioning of society, there are increasing reports of ethical risks and violations related to software use [1]–[3], such as algorithmic bias, privacy violations, lack of accessibility and misuse of software technology. Ethics is now recognised as an important concern throughout the software lifecycle, including software development, operation and maintenance. It is also recognised that software practitioners should consider the ethical as well technical consequences of the decisions they make in the course of their work.

Professional bodies such as ACM and IEEE and academic researchers have produced codes of ethics to offer guidance to software professionals in ethical and professional conduct [4], [5]. However, most recent research in this area has focused on the ethics of using AI and ethical concerns related to the collection and use of data. Despite the availability of codes of ethics and sets of ethical principles, particularly aimed at AI, and the recognition that ethics should be a design level concern, there is a lack of tool and process support for systematic and applied ethical deliberation during the lifecycle of software systems in general.

The hypothesis of our broader work is that, if ethical concerns and decisions are adequately represented by software engineering artefacts, and supported by processes and tools integrated into the software lifecycle, then the behaviour of practitioners, and systems themselves, will better conform to ethical principles.

The overall aim of the work introduced in this paper is to facilitate the explicit consideration of ethical concerns in software engineering artefacts and processes. We explore related academic work on computer and software ethics. In order gain an understanding of the practical challenges of ethical deliberation in industry, we interview software professionals on their experience of ethics guidance and practice.

The main contributions of this paper are the insights gained from this process and possible avenues for future work to improve ethical practice in software development. The paper is structured as follows. Background and related work on ethical theories and ethics in the context of software engineering are discussed in Section II. Details of the interviews with software professionals, including objectives, design and process, are presented in Section III. The results of the interviews are summarised in Section IV. Section V brings together the insights gained from related work and interviews. Limitations of the study are outlined in Section VI. Section VII builds on the insights gained to discuss the challenges and opportunities for incorporating ethics in software engineering. Conclusions and some avenues for future work are outlined in Section VIII.

II. Background and Related Work

In this section, we present an overview of the background and some existing work relating to ethical concepts, codes of ethics, sets of ethical principles and current solutions for addressing ethical issues in Computer Science and software engineering.

The Encyclopaedia Britannica[1] defines ethics, or moral philosophy, as "the discipline concerned with what is morally good and bad and morally right and wrong." The study of ethics can be conducted at three different levels of abstraction: metaethics, which is the study of the nature of ethics, normative ethics, which is concerned with providing an account of the standards for morally right or wrong actions, and applied ethics, which is the study of specific ethical cases. Normative ethics is divided into three dominant approaches, individuated according to whether duties (deontology), outcomes (consequentialism) or virtues (virtue ethics) are prioritised. Most existing work on ethics in Computer Science is concerned with the practical and normative question of how to direct actions in ways which promote ethical decision making.

Researchers have suggested that computing ethics should be considered an infrastructural ethics or 'infraethics', which is "a first-order framework of implicit expectations, attitudes, and practices that can facilitate and promote morally good decisions and actions" [6]. An infraethics is implicit within any complex society, and for it to effectively promote moral action, there must also exist an appropriate moral code. Democracy

[1]https://www.britannica.com

is an illustrative example, the principles of autonomy and responsibility being important to a democratic society. These principles are difficult to balance, and it is the infraethics (the political structure in place) which both enables this balance, and allows the moral action which arises from autonomy and responsibility to be promoted within a democratic society.

Codes of ethics are sets of principles designed to facilitate ethical decision-making. Such codes are published by regulatory bodies, governments, companies and academia in a diverse range of disciplines. The ACM / IEEE Software Engineering Code of Ethics can be used as a useful foundation for framing different ethical concerns, such as public, employer and client, product, judgement and self [4], although there is some evidence that knowledge of the Code of Ethics does not influence decision making on ethical matters [8].

The Algorithm Watch Inventory[2] contains a collection of over 150 AI ethics guidelines. Floridi and Cowl note that this proliferation of guidelines and codes of ethics can lead to either redundant or conflicting principles, and propose a unified framework of five principles by systematically analysing and condensing six high-profile sets of ethical AI principles [5]. The principles of this framework are beneficence, non-maleficence, autonomy, justice and explicability. Researchers have also argued for the shift from codes of conduct to ethical deliberation throughout the software development process [9].

The principles of accountability, responsibility, and transparency (ART), proposed to inform the ethical attitudes of professionals working with AI systems, particularly in the context of coding ethics into artificially moral agents (AMAs) [10], [11], can be more widely applied to the field of software engineering. Accountability is the requirement to explain and justify decisions and actions. Decisions must be derivable from data and algorithms, which must represent the moral values and societal norms to be used in deliberation. There must be a clear chain of responsibility, including human stakeholders, for the decisions and actions taken by the system. Transparency is the requirement to describe, inspect and reproduce the mechanisms through which the system makes decisions. This includes concerns such as data provenance.

Vakkuri et al. use the ART principles in a research study to explore current ethical practices in start-up like environments where AI systems for the health care sector are developed [12]. This study found that software developers who took part largely felt a personal sense of responsibility for creating ethical systems. However, the value of creating a working prototype was considered higher than that of incorporating ethics into the early stages of a project. Furthermore, participants did not have a clear sense of their ethical responsibilities. A related finding of this study is that developers did not have plans to deal with unexpected consequences of technical decisions or use contexts of systems. The study thus found that there is a gap between ethical research and ethical practice in AI projects.

[2]https://inventory.algorithmwatch.org

Ethics is now acknowledged as a concern that should be addressed at design level. Concepts such as Value-Sensitive Design [13], Ethics by Design in the context of AI [14] and Choice Architecture [15] highlight the need for tools, methods and design guidelines to support ethical decision making by human and autonomous agents. Lurie and Mark [16] propose an Ethical-Driven Software Development (EDSD) framework that attempts to connect software developers' ethical responsibilities to their professional standards. They use EDSD index cards with yes / no questions at different phases of software development to improve ethical awareness in stakeholders. Alidoosti [17] describes planned work in creating ethical guidance for architecture decision making through the use of a stakeholders map and a value model.

Human-centric model-driven software engineering [18] has been proposed to address the lack of consideration of differences in human characteristics (such as age, gender and culture) between users of software systems. Value-based software engineering [19] aims to explicitly consider and reconcile value objectives, which are not necessarily economic, from all stakeholders of a system.

Our broader work aims to build on the insights and experiences presented in these works to explore how ethical deliberation can be supported in software engineering. The work presented in this paper uses the ART principles as the framework for discussing ethical concerns with software practitioners. The design of our interviews was inspired by the one used by Vakkuri et al and we compare the insights we gained with their work in a more specific context.

III. Interviews with Practitioners

Our review of academic literature on ethics in Computer Science, outlined in Sections I and II, highlights both the importance placed on ethical awareness and deliberation in software engineering and the lack of integrated tool and process support to facilitate such deliberation. To augment our understanding of the issues and to gain insights into the practical challenges of ethical deliberation in industry, we interviewed software professionals.

Our aim was to gain an understanding of ethical awareness in software practitioners and the ethical infrastructure, such as codes of ethics, ethics processes and supporting tools, that is available to software practitioners to facilitate ethical deliberation in their work. The insights gained from this process will be used in future research to improve the state of the art in providing tools and processes for ethical deliberation that are integrated into the software lifecycle.

A. Interview Objectives

Interviews with software practitioners were conducted with the following objectives designed to achieve the aim of gaining insights into ethical awareness and support for ethical deliberation in software industry:

1) Establish whether organisations provide, or suggest, a comprehensive code of ethics to enable their employees to incorporate ethics into their work.

2) Investigate the ethical opinions of practitioners with respect to accountability, responsibility, and transparency, within their work.
3) Establish whether practitioners find their ethical responsibilities clear within their role.
4) Establish whether, and to what extent, ethical practices are formalised in technical roles.
5) Investigate the current tools which are available to assist the incorporation of ethics into software development.
6) Investigate which factors may influence the extent to which ethics is considered within an organisation.
7) Collate a set of requirements for a tool which would enable the incorporation of ethics into software engineering.

The ART principles were chosen as the basis for discussing ethics in software engineering since they are well-established in literature, having been included as part of the IEEE Guidance on Ethically Aligned Design[3], and can be considered in the context of software systems in general.

B. Recruitment of Participants

The Covid-19 pandemic affected both the recruitment process and the interview schedule. Participants were recruited for interviews mainly via the LinkedIn platform. The advertisement described the purpose of the research and stated that interviews would focus upon accountability, responsibility and transparency with respect to participants' work. A total of 10 participants took part.

Each participant was a software professional working for a different organisation. These organisations included startups, SMEs, and large enterprises. Participants were at diverse stages of their careers.

C. Interview Process

Participants were asked to take part in an interview lasting approximately 30 minutes regarding how they addressed ethical concerns in their work. Ethical approval for this process was obtained from the authors' academic institution.

Interviews were conducted on Microsoft Teams and recorded to allow for automated transcription. The interviews were semi-structured, with a list of initial questions. Further clarifications were sought as necessary. Software professionals at an early stage in their career and working on a wide range of projects sometimes chose to answer questions based upon the most recent projects they had worked on. Others, who were more established in their role, chose to answer based upon their career as a whole, or their previous role if they had recently moved company. This decision was made by the interviewees themselves.

D. Interview Design

14 questions were used as the basis for the semi-structured interviews. The first two questions, related to the interviewee's role in their organisation, were adapted from [20]. The remaining questions were inspired by the interviews conducted by Vakkuri et al. [12], although the aim of our interviews was broader and therefore participants were recruited from a variety of organisations, both in terms of sector and development methodology.

The interview questions were:

1) Which of the following describes your role in your organisation?
2) Which of the following processes are you generally involved in during a project?
3) Where would you describe your team's methodologies as being on the spectrum from completely agile to completely plan-driven?
4) How would you describe the size of your organisation? (e.g., startup, SME, large)
5) Does your organisation have a code of ethics?
6) What do you feel are your responsibilities within a project?
   - Do you feel you have ethical responsibilities, and if so, what are they?
7) Which aspects of your work are you accountable for?
   - For example, do you feel accountable for the future use of software you create? Do you feel accountable if software you have worked on goes on to be used for a different use-case?
8) Do you feel it is clear what your ethical responsibilities are with respect to your role?
9) Do you consider transparency during your work? (e.g., through code documentation/code review)
   - Are there any other ways you ensure transparency?
10) Black box issues are an example of a widely discussed issue of transparency. Are there any others which you consider during your work specifically?
11) Are the current ethical practices you undertake in your work formalised? (e.g., documentation, code review)
12) Are there any other practices currently in place in your organisation which promote ethical considerations?
13) Do you track data or software provenance or include this information when documenting software?
14) What kind of tool support do you currently have to help you integrate ethical considerations?
    - If you do, then do you find it adequate?
    - If not, or you don't find it adequate, what would your requirements be?

For questions 1 and 2, a list of pre-defined options was provided along with the option to specify an 'Other' response.

A glossary of the terms accountability, responsibility, transparency and data provenance was prepared and provided to participants in case clarifications were required.

The first 4 questions were used to gain an understanding of the working environment of participants and to determine whether this influenced ethical awareness or support for ethical deliberation. The remaining questions address the interview objectives outlined in Section III-A.

[3]https://ethicsinaction.ieee.org/wp-content/uploads/ead1e-overview.pdf

E. Analysis of Interview Data

Interview recordings were transcribed to text and anonymised. Since this was a small study, the data analysis was carried out manually. Responses to questions that were not open-ended (such as 1, 2, 4 and 5) were aggregated to determine their distribution. A thematic content analysis on data from the interviews was carried out by one of the authors with support from other authors. The following broad themes were used to analyse the responses: Codes of Ethics, Responsibility, Accountability, Transparency, Ethical Cases, and Current Tooling and Requirements.

IV. Results

In compliance with the ethics approval granted for the study, responses have been anonymised or summarised in the following sections to maintain the anonymity of participants. For the same reason, specific examples that can potentially identify individual persons or organisations have not been included.

Participants' roles in their organisations related to the following aspects of the software lifecycle: Software requirements, software architecture and design, software development, software testing, software maintenance, software configuration management and project management. One participant was additionally a user/user representative.

Five of the ten participants either definitively stated that their organisation did not have a code of ethics, or said that they were uncertain but had never been made aware of a code. Upon researching the organisations, it is clear that where employees were uncertain about the existence of a code, there is no code publicly available.

Four participants work for large organisations which have a technical department but are not technical organisations. These organisations do have codes of ethics, but they are largely legal or regulatory based and are not helpful when making decisions concerning software engineering. One participant works for a technical organisation which has a code of conduct. However this code does not focus on computing ethics apart from briefly describing some potential legal issues.

Only one of the ten organisations pointed their developers towards a specific computing code of ethics. The interviewee from this organisation works in security, and all employees are encouraged to be members of a professional organisation, and are therefore expected to comply with that organisation's code of ethics. They also noted that their company's own code of ethics did not aid technical decision-making.

Seven interviewees either do not believe that they made ethical decisions in their work, or believe that these decisions are minor, and therefore feel little responsibility. This group includes all five of the interviewees who work within the set of organisations which have legal and regulatory focused codes of ethics.

Regardless of interviewees' personal awareness and understanding of ethics in computing, they all felt either that their ethical responsibilities were not clear, or that they do not have ethical responsibilities at all. The clarity of participants' ethical responsibilities can therefore be categorised into three groups: those who believe that they do not have ethical responsibilities; those who believe that their ethical responsibilities are limited enough to not require clarification; and those who believe that they have a degree of ethical responsibility, but within their workplace it is not clear what these responsibilities are. Even interviewees who did not believe their work had ethical implications were concerned about ethics in computing as an industry.

Regarding the incorporation of transparency into their work, having already spoken about accountability and responsibility, one interviewee suggested that their work might be transparent for the wrong reasons because transparency is treated as a practical but not an ethical requirement. This focus on the practical benefits of good practices, such as clear documentation allowing for more efficient use of time, understandable algorithms creating a better product, and code review verifying that people are completing work, or speeding up the development process, or fulfilling legal obligations, was mentioned in every interview.

All interviewees discussed the use of version control, documentation, and code review to promote transparency in their work. However, only the four interviewees who work in security said that these practices are pursued formally. The decision to pursue these practices is up to the discretion of the team or individual. Leaving the decision to pursue transparency up to individuals can lead to software developers cutting corners to save time, particularly during agile development or when software artefacts are developed for internal use.

Only interviewees who worked in security formally considered the future operational life of the system out on the field. Only one other interviewee had considered possible product misuse. Similarly, only interviewees working in security had formal plans to deal with, and prevent, unexpected behaviour of the system.

When questioned about accountability, interviewees provided examples of instances when they have acted, or considered acting, morally. Regarding their ethical responsibilities, interviewees gave examples of instances when they were expected to perform an ethically significant task, but were not given any guidance. These anecdotes related to special category data, incomplete data sets, post-decision justification, potential whistleblowing, and unverified code.

All interviewees said that they use software engineering practices such as version control and pair programming. They also all mentioned the use of project management software such as Git or virtual Kanban boards to delegate tasks and maintain transparency. However, these tools were primarily used for practical reasons such as limiting the length of meetings and checking that employees are contributing satisfactorily to their team. Only four of the interviewees were able to describe specific tools or formalised processes which assist the integration of ethical decisions into their work, two of which are the formalised documentation processes described above. Again, three of these interviewees work in security. The tool described automates part of the consideration of data provenance.

Five interviewees responded that their ethical tooling was adequate, despite the fact that they had described several ethical concerns, from issues with documentation to concrete examples of ethical dilemmas.

Suggested features for a tool to assist the incorporation of ethical considerations were a formalised protocol for documentation, a way to reliably identify when there are ethical considerations in a project, and automatic identification of unsafe data handling.

V. Insights from Interviews

The insights gained from related work and the interviews are outlined in this section in the form of empirical conclusions (EC).

EC1 (Theme: Codes of Ethics, Corresponding objective: 1) Most organisations do not provide, or recommend, a computing or software-specific code of ethics.

EC2 (Theme: Codes of Ethics, Corresponding objective: 1) When software professionals are provided with a code of ethics, this is often a non-technical code.

EC3 (Theme: Responsibility, Corresponding objective: 2) A majority of software professionals believe either that they do not have ethical responsibility, or that this responsibility is very limited.

EC4 (Theme: Responsibility, Corresponding objective: 2) Software professionals feel most responsibility towards tackling problems related to software development, such as finding bugs and meeting project goals.

EC5 (Theme: Responsibility, Corresponding objective: 2) On a personal level, software professionals are concerned about the ethical aspects of their work. However, little is done to tackle these concerns.

EC6 (Theme: Codes of Ethics, Corresponding objective: 2) If software professionals are only provided with a non-technical code of ethics, they are more likely to assume that ethical concerns are not relevant to their work.

EC7 (Theme: Transparency, Corresponding objective: 2) Established software engineering practices, such as code documentation and code review, support transparency in systems development. However, these practices are not typically used to support transparency for ethical reasons.

EC8 (Theme: Accountability, Corresponding objective: 2) Product misuse and error scenarios, which may have ethical implications, are only considered during development outside security organisations. They are rarely considered in terms of the future operational life of the system out in the field.

EC9 (Themes: ART, Corresponding objective: 2) Software professionals who work outwith security do not have plans to deal with unexpected behaviour of the system, for example, resulting from machine learning or the future expansion of the use of the system.

EC10 (Theme: Responsibility, Corresponding objective: 3) The ethical responsibilities of software professionals in their work environment are not clear to them.

EC11 (Theme: Transparency, Corresponding objective: 4) All software professionals recognise transparency as a goal, but it is not formally pursued outwith security. Even in security, compromises may be made, particularly if the software is for internal use.

EC12 (Themes: ART, Corresponding objective: 5) There are few tools currently being used by software professionals to integrate ethics into their work.

EC13 (Theme: Accountability, Ethical Cases, Corresponding objective: 6) A software professional's influence within an organisation correlates with their likelihood to raise, or personally mitigate, ethical concerns, and to feel a greater degree of personal responsibility.

EC14 (Themes: ART, Current Tooling and Requirements, Corresponding objectives: 2, 7) Software professionals would like to be explicitly made aware if there are potential ethical issues at stake within any project they are working on and would like guidance in documenting ethical concerns.

Empirical conclusions (ECs) 1, 2, 3, 6, 12, 13 and 14 are novel insights from this study. ECs 4, 5, 6 and 10 support empirical validations or conclusions from the work of Vakkuri et al., which suggests that their work may be more widely applicable than AI development in start-ups. The other three conclusions qualify insights from the work of Vakkuri et al., which suggests that while their work on transparency may be more widely applicable, this is not necessarily true with respect to current practices within security.

The study did not aim to consider security as a particular context or concern. However, during interviews, it emerged that there were notable differences in working practices in security and other contexts. These are noted in the findings.

VI. Limitations of Findings

In this section we discuss the potential threats to the validity of the study and its findings. The interviewees were a self-selecting group, who responded to an advertisement to take part in the study. It is therefore plausible that the participants' interest in ethics is not representative of the sector as a whole. To gain a broader insight into how software professionals in general view ethics in their work, a larger dataset would be required. However, there is support for qualitative case study research, particularly for novel research areas [21].

Only ten interviews were carried out in total. Whilst each interviewee worked for a different organisation, some worked within similar sectors. More interviews covering practitioners from a larger range of tech sectors may have yielded different results.

The interview format allowed participants to answer with little guidance. For example, when discussing ethical responsibility, if questions about specific ethical issues had been asked, it is possible that interviewees would have been more likely to recognise the ethical concerns within their work. However, even after discussing ethical issues, most interviewees maintained that ethical issues are not a primary concern in their work.

There are many different frameworks and codes of ethics. An interview based on a different framework to the ART principles may have yielded different results. For example, if interviews had focused on principles such as trust and fairness, the results may have reflected issues concerned with diversity. Similarly, there are broader ethical concerns, such as sustainability, energy consumption and accessibility, which were not explicitly addressed in these interviews.

Some of these limitations were due to circumstances prevalent at the time of this research. The scope of interviews and the basis for questions were determined as appropriate for an initial exploration of the state of practice in ethics. Plans for future work, outlined in Section VIII, aim to address some of these limitations.

VII. Discussion: Challenges and Opportunities

The results of interviews with software professionals broadly support the conclusions from existing work in specific contexts such as the study by Vakkuri et al. on AI systems for health care in start-up environments, but also refine and clarify some of their conclusions.

The structure of the interviews was such that interviewees were asked about their technical / project responsibilities prior to questions about ethical responsibility. The ease with which all interviewees summarised their technical responsibilities compared to their uncertainty with respect to ethical responsibilities is indicative of a lack of ethical clarity, even when interviewees had a personal interest in ethics.

The lack of ethical emphasis and clear ethical guidance from nine out of 10 organisations that employed participants in the study is of concern, particularly given the high profile ethical violations reported in recent years. The responsibility of ethical deliberation is apparently left to individual employees, relying on their knowledge, experience and motivation to pursue these concerns. The finding that an employee's influence within an organisation correlates with their likelihood of pursuing ethical concerns, even though all were concerned about ethics in general, warrants further investigation. This may be due to one or more factors, such as more senior employees having more power, more job security, more time and resources, or more knowledge and experience, to pursue ethical matters.

If ethical deliberation is not part of organisational culture as well as software engineering processes, it is likely to be sacrificed in the interests of business concerns, time constraints and what might be perceived as technical correctness of the system. However, we would suggest that a system is not correct unless it is correct for everyone and this requires ethical deliberation.

The limited responsibility reported by interview participants suggests that traditional Aristotelian virtue ethics can be applied to ethics within software engineering. According to the concept of phronesis (moral or practical wisdom), a person is morally culpable if their understanding of what is beneficial or harmful is mistaken (Aristotle NE 1103a 1–10). According to virtue ethics, it is the responsibility of moral agents to have insight into the effects of their actions and to be able to justify them accordingly.

Without sufficient moral wisdom, practitioners may be unable to consider the ethical concerns and consequences associated with a system. Rather than attempting to enforce ethical decisions through prescriptive principles or specific outcomes, we suggest a focus on highlighting and clarifying the ethically salient features of practitioners' work, such as considering the potential impact of technical decisions on all stakeholders, and providing adequate tool and process support for this undertaking.

Framing computing ethics in terms of virtues shifts the focus from merely providing prescriptive maxims, aimed at constraining the ethical decisions themselves, to providing a framework for developing the moral character of agents. Prioritising either outcomes or duties can result in conflicting interpretations of a prescribed course of action, as justification to a deontologist may not constitute justification to a consequentialist and vice versa. Focusing upon informing practitioners in order to facilitate ethical understanding, rather than prescribing specific context-free duties or outcomes, can further act to mitigate these potential conflicts throughout the software lifecycle.

Prioritising virtues over prescribed actions does not entail that ethical principles should not be applied to software engineering, but rather that these principles should be considered in terms of virtue ethics. For example, the ACM Code of Ethics and Professional Conduct[4] implicitly suggests an Aristotelian approach, with the aim of the Code being to guide professionals, not to provide an algorithmic solution to ethical decision-making.

The insights obtained from interview participants working within non-technical organisations suggest that these practitioners are less aware of ethical concerns. These participants justified reduced responsibility by appealing to the regulatory principles within their organisations' legal or regulatory codes of ethics. One potential approach to facilitating ethical awareness is therefore the promotion of codes such as those of the ACM / IEEE. These codes are designed to apply to all aspects of software engineering and, as such, highlight many relevant ethical issues.

Related work and insights gained through interviews suggest that ethical concerns and requirements should be explicitly specified for systems as part of software development, and this documentation should be accessible to all stakeholders involved so that they are reminded and encouraged to consider the ethical as well as technical consequences of their decisions. In supporting this practice, software engineering artefacts such as system requirements, software architecture, software design and test suites, in addition to software itself, can constitute part of the infraethical environment which promotes and sustains moral behaviour. Explicitly recording ethical concerns in software engineering artefacts can help remind stakeholders

[4]https://www.acm.org/code-of-ethics

of ethical consequences, track outstanding issues and carry out ethical impact analyses.

Integrated tool support for the above mentioned activities will be equally important. Software development and project management tools may be enhanced to remind practitioners to consider ethics, represent or document ethical concerns in artefacts, analyse potential ethical conflicts, identify ethical consequences of decisions and generate reports of outstanding ethical issues.

Education and training of software professionals in software ethics are complementary requirements to ethical infrastructure in achieving these objectives. Practitioners need to be able to anticipate and identify potential ethical concerns as well as make ethical decisions to resolve dilemmas.

VIII. Conclusions and Future Work

As software becomes more pervasive, researchers and practitioners recognise that software ethics becomes correspondingly more important. In this paper, we have provided an overview of related work on the topic and explored the state of practice through interviews conducted with software professionals working in industry. The insights gained from these processes indicate that, while there is awareness of potential ethical issues among practitioners, much more work needs to be done in providing an environment and adequate tools to encourage ethical deliberation throughout the software lifecycle. Thus, we have motivated the incorporation of ethics in software engineering artefacts and processes.

We suggest that facilitating and promoting good ethical practice in and through software engineering processes and artefacts, rather than prescribing specific actions or outcomes, would help inculcate the more generic and transferable habit of ethical deliberation among software professionals.

We envisage many avenues for future work in this important area. A larger study involving software professionals across a broader range of sectors and roles will provide more insights into the state of practice and challenges in incorporating ethics into the software lifecycle. The study reported in this paper used the ART principles as the basis for discussing ethics. While these were deemed appropriate for an initial investigation, a broader study addressing further dimensions of ethics such as fairness, accessibility and sustainability will yield more comprehensive and deeper insights ethical practice in industry.

An implementation of support for the representation and deliberation of ethics in software engineering artefacts is an important step in assessing the feasibility of our approach. Its viability and effectiveness in practice will be evaluated with the involvement of software professionals. Given the broad awareness of ethical concerns we identified among practitioners from interviews, we believe that the provision of ethical infrastructure through artefacts will lead to better moral practice overall.

References

[1] Rashid, A., Weckert, J., Lucas, R.:Software Engineering Ethics in a Digital World. Computer 42(6), 34–41 (2009)

[2] Isaak, J., Hanna, M. J.: User Data Privacy: Facebook, Cambridge Analytica, and Privacy Protection. Computer 51(8), 56–59 (2018)

[3] Fry, H.: Hello World: How to be Human in the Age of the Machine. Transworld (2018)

[4] Gotterbarn, D., Miller, K., Rogerson, S.: Software Engineering Code of Ethics. Commun. ACM 40(11), 110–118 (1997)

[5] Floridi, L. Cowl, J.: A Unified Framework of Five Principles for AI in Society. Harvard Data Science Review, 1(1) (2019)

[6] Floridi, L.: Distributed morality in an information society. Science and engineering ethics, 19(3), 727–743 (2013)

[7] Booch, G.: Morality and the Software Architect. IEEE Software 25(1), 8–9 (2008)

[8] McNamara, A., Smith, J., Murphy-Hill, E.: Does ACM's Code of Ethics Change Ethical Decision Making in Software Development?. In: Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, ESEC/FSE 2018, Lake Buena Vista, FL, USA, pp. 729–733. Association for Computing Machinery, New York, NY, USA (2018).

[9] Gogoll, J., Zuber, N., Kacianka, S.,et al.: Ethics in the Software Development Process: from Codes of Conduct to Ethical Deliberation. Philosophy and Technology 34, 1085–1108 (2021)

[10] Dignum, V.: Responsible Autonomy. In: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI-1), pp. 4698–4704, International Joint Conferences on Artificial Intelligence, Melbourne (2017)

[11] Dignum, V., Dignum, F., et al.: Design for Values for Social Robot Architectures. In: Robophilosophy Conference, pp. 43–52, Robophilosophy/TRANSOR, Vienna (2018)

[12] Vakkuri, V., Kemell, K. K., Jantunen, M., Abrahamsson, P.: "This is Just a Prototype": How Ethics Are Ignored in Software Startup-Like Environments. In Lecture Notes in Business Information Processing (LNBIP), 383, pp. 195–210 (2020)

[13] Friedman, B.: Value-sensitive design. Interactions 3(6), 16–23 (1996)

[14] Dignum, V., et al.: Ethics by Design: Necessity or Curse?. In: Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society (AIES '18), pp. 60–66, New York, NY, USA (2018). Association for Computing Machinery.

[15] Thaler, R., Sunstein, C., Balz, J.: Choice architecture. The behavioral foundations of public policy. Princeton University Press (2014)

[16] Lurie, Y. and Mark, S.: Professional Ethics of Software Engineers: An Ethical Framework. Science and Engineering Ethics 22(2), 417–434 (2016)

[17] Alidoosti, R.: Ethics-driven Software Architecture Decision Making. In: 2021 IEEE 18th International Conference on Software Architecture Companion (ICSA-C), pp. 90–91, IEEE, Stuttgart, Germany (2021)

[18] Grundy, J., Khalajzadeh, H., Mcintosh, J.: Towards Human-centric Model-driven Software Engineering. Proceedings of the 15th International Conference on Evaluation of Novel Approaches to Software Engineering - ENASE, 229–238 (2020)

[19] Biffl, S., Aurum, A., Boehm, B., Erdogmus, H., Grünbacher, P. (eds): Value-Based Software Engineering. Springer Berlin (2006)

[20] Tu, Y-C.: Transparency in Software Engineering. PhD thesis. University of Auckland, New Zealand (2014)

[21] Eisenhardt, K. M.: Building theories from case study research. Academy of management review, 14(4), 532–550 (1989)