Digital platforms and responsible innovation
expanding value sensitive design to overcome ontological uncertainty

de Reuver, Mark; van Wynsberghe, Aimee; Janssen, Marijn; van de Poel, Ibo

DOI
10.1007/s10676-020-09537-z

Publication date
2020

Document Version
Final published version

Published in
Ethics and Information Technology

Citation (APA)
de Reuver, M., van Wynsberghe, A., Janssen, M., & van de Poel, I. (2020). Digital platforms and responsible innovation: expanding value sensitive design to overcome ontological uncertainty. *Ethics and Information Technology*, *22*(3), 257-267. https://doi.org/10.1007/s10676-020-09537-z

Ethics and Information Technology
https://doi.org/10.1007/s10676-020-09537-z

ORIGINAL PAPER

Digital platforms and responsible innovation: expanding value sensitive design to overcome ontological uncertainty

Mark de Reuver[1] · **Aimee van Wynsberghe**[1] · **Marijn Janssen**[1] · **Ibo van de Poel**[1]

Abstract
In this paper, we argue that the characteristics of digital platforms challenge the fundamental assumptions of value sensitive design (VSD). Traditionally, VSD methods assume that we can identify relevant values during the design phase of new technologies. The underlying assumption is that there is only *epistemic uncertainty* about which values will be impacted by a technology. VSD methods suggest that one can predict which values will be affected by new technologies by increasing knowledge about how values are interpreted or understood in context. In contrast, digital platforms exhibit a novel form of uncertainty, namely, *ontological uncertainty*: even with full information and overview, it cannot be foreseen what users or developers will do with digital platforms. Hence, predictions about which values are affected might not hold. In this paper, we suggest expanding VSD methods to account for value dynamism resulting from ontological uncertainty. Our expansions involve (1) extending VSD to the entire lifecycle of a platform, (2) broadening VSD through the addition of reflexivity, i.e. second-order learning about what values to aim at, and (3) adding specific tools of moral sandboxing and moral prototyping to enhance such reflexivity. While we illustrate our approach with a short case study about ride-sharing platforms such as Uber, our approach is relevant for other technologies exhibiting ontological uncertainty as well, such as machine learning, robotics and artificial intelligence.

Keywords Value sensitive design · Digital platforms · Emergent values · Responsible innovation · Value dynamism

Introduction

In today's society, digital platforms are ubiquitous, ranging from social media platforms to smartphone operating systems. Through digital platforms, organizations make their technologies, data and user base available to third parties (De Reuver et al. 2018). For instance, platforms allow consumers to find taxi drivers, or municipalities to share city data (Janssen and Estevez 2013). While digital platforms offer convenience and stimulate innovation, they also produce undesired societal consequences. For instance, mobile platforms such as Android and iOS have spawned millions of apps, which have positive implications (e.g. productivity apps to organize work more efficiently) but also negative ones (e.g. productivity apps negatively affect employees' work-life balance) (Yun et al. 2012). Other scholars warn of the increased risks to privacy and security (Mineraud et al. 2016), and broader societal impacts on markets, democracy and social life (Dow Jones Newswires (2019).

One approach to anticipate and account for the negative value implications of new technologies is *value sensitive design* (VSD). VSD has been applied to a variety of mainly ICT and robotics projects (Friedman and Kahn 2003; Davis and Nathan 2015; van den Hoven et al. 2015; van Wynsberghe 2013), and various modifications to the original method have been proposed. The VSD approach assumes that value implications of a technology should be identified, prior to creating the technology (Friedman et al. 2006). The identified value implications should then be addressed in implementations of the technology to mitigate negative impacts. A fundamental assumption of VSD is that uncertainties about which values will be affected by a technology are *epistemic*, which means that they can be resolved by increasing knowledge and understanding.

Two characteristics of digital platforms challenge the fundamental assumption of epistemic uncertainty in VSD. First, digital platforms often *mediate* between users: they

✉ Mark de Reuver
g.a.dereuver@tudelft.nl

[1] Faculty Technology, Policy and Management, Delft University of Technology, Delft, The Netherlands

Published online: 13 May 2020

are intermediaries between consumers (e.g. social media) or between consumers and third parties (e.g. game consoles) (Rochet and Tirole 2003). In which ways users interact through a platform, and for what purposes, is often beyond the control of the designer of the platform. Second, digital platforms are extensible, which means that third parties (e.g. app developers) can add new modules (e.g. apps) over time, without actively involving the platform owner (e.g. operating system provider) (Hanseth and Lyytinen 2010; Tilson et al. 2010). This makes digital platforms generative (Zittrain 2006): one can principally not foresee what modules will be added to a platform in the future (Boudreau 2012). The mediating role and generativity of digital platforms, we suggest here, results in *ontological* uncertainty about value implications. Ontological uncertainty entails that it is indeterminate how others will use a digital platform and what values are impacted, which makes it difficult to foresee which values are relevant prior to implementing the technology.

Given the need for the responsible design of digital platforms, evidenced in recent scandals, this paper explores how VSD methods can be expanded to be suitable for digital platforms. Specifically, we suggest expansions to VSD methods to address the ontological uncertainty in value implications of digital platforms. In "Background" section, we discuss the key characteristics of digital platforms and VSD. "Expanding VSD for digital platforms: the TERC model" section outlines our three main expansions to VSD: increased iterations, introduction of reflexivity, and moral sandboxing. In "Illustration: ride-sharing platforms" section we illustrate our expansions through a case study. Although the focus of this paper is on the application of VSD to platform design, our approach may also be valuable for other technologies that exhibit ontological uncertainty, such as machine learning algorithms.

Background

Key characteristics of digital platforms

Many definitions of platforms can be found in the literature, each emphasizing different characteristics. We broadly define platforms as foundations upon which unrelated actors can interact and offer services or products (Gawer 2009). In the case of digital platforms, these foundations consist of hardware and software modules, as well as rules and standards on how to interact with these modules (Tilson et al. 2010). Platforms have two main characteristics that are important for our argument: mediation and extensibility.

The first characteristic is that most digital platforms *mediate between user groups* (Rochet and Tirole 2003). For instance, a ridehailing platform mediates between taxi drivers and consumers, and operating systems mediate between consumers and app developers. By mediating between groups, platforms reduce transaction costs, for instance search and contracting costs in ridehailing services. Platforms that mediate between users exhibit network effects: they become more valuable as more users join (Katz and Shapiro 1985). However, how and for what purpose users will interact through a digital platform cannot always be foreseen when a platform is being designed.

A second characteristic is that digital platforms can be *extended with applications* (Tilson et al. 2010). For instance, an operating system platform can be extended with apps that provide additional functionality. The modules in the platform are relatively generic and stable, whereas the add-on modules are more specialized (Baldwin and Woodard 2009). Innovations can be realized by developing new add-on modules, or by recombining platform and add-on modules in new ways (Henderson and Clark 1990). As a consequence, platforms are *generative*: they enable unanticipated add-ons (Boudreau 2012) without active involvement of the platform provider (Tilson et al. 2010; Bygstad 2017). Add-ons on digital platforms can even change the functionality of the platform itself (Yoo et al. 2010).

Both characteristics of digital platforms give rise to ontological uncertainty for the platform provider: how users interact on the platform, and what add-on modules third parties will build on top of the platform, cannot, in principle, be foreseen while designing the platform.

Control of digital platforms

Digital platforms can be used to interact and develop applications in ways that cannot be foreseen while designing the platform (Boudreau 2012). For digital platform providers, this poses a risk. For instance, add-on applications can be harmful or ill-performing (Wareham et al. 2014) that can even destabilize the platform on which they run (Wessel et al. 2017). Consequently, there is a wealth of literature available exploring measures of control and governance to prevent the undesirable usage or add-ons (e.g. Wareham et al 2014).

Borrowing from control theory (Ouchi 1979), platform providers may exert control over third parties through: input, output, behavioural, or normative measures (Tiwana et al. 2010). For instance, in the case of app stores, the provider may impose conditions for registering as an app developer (i.e. input control), give revenue sharing incentives to well-performing apps (i.e. output control), dictate the way apps are developed through software development kits (i.e. behavioural control) or use reputational systems (i.e. normative control). Platform providers can exercise control through so-called boundary resources, which make a platform accessible for applications (Ghazawneh and Henfridsson 2013).

Some boundary resources, such as application programming interfaces (APIs), help to *resource* add-on applications, while others, such as terms and conditions for usage, help to *secure* the platform. Empirical studies have shown that platform control helps safeguard relations between platform provider and customer(s) (Mukhopadhyay et al. 2016).

Although control mechanisms may help to mitigate undesirable behaviour by third parties, platform providers face challenges in exercising control. First, tight control can have negative implications on the viability of a platform. The degree to which control is exerted may negatively affect motivations of app developers to contribute to a platform (Goldbach et al. 2014; Schaarschmidt et al. 2018). In fact, giving up control makes it more likely for a platform provider to attract users and developers, and hence survive on the market (Ondrus et al. 2015). Second, platform providers struggle to exercise control effectively. An extensive case study on the Apple iOS platform showed that control over boundary resources is continuously contested by third parties, for instance by pressuring the platform owner to open up its APIs (Eaton et al. 2015). Third, platform providers face time pressure to launch platforms on the market, which prevents them from contemplating control mechanisms extensively. The pressure to launch new platforms quickly is growing as large providers are `enveloping' their existing platforms into new markets (see e.g. Amazon and Apple) (Eisenmann et al. 2011). Further, the software industry has adopted agile and scrum development methods, based on the notion of trying out minimum viable products quickly and testing while on the market.

In sum, although control and governance of digital platforms is desirable to mitigate undesirable outcomes, implementing control mechanisms is challenging for multiple reasons. What is needed are design approaches that allow a rapid and agile approach to designing digital platforms, while at the same time provide space for consideration of societal values and how they may be impacted by one design choice or another.

Value sensitive design

As a theory, VSD aims at systematically integrating values of ethical importance into technological designs (Friedman et al. 2006). While today various methods exist to achieve this goal, original VSD methods were developed by design scholar Batya Friedmann. In the last decades, VSD methods have been refined through contributions from scholars from multiple countries and disciplines (Friedman and Kahn 2003; Friedman et al. 2006; Davis and Nathan 2015; van den Hoven et al. 2015; van Wynsberghe 2013). VSD has been successfully applied to a variety of ICT and robotics projects. Similarly, a range of approaches have been articulated and developed that go by other names such as values in design (VID), values at play, and design for values (DfV) (e.g. van den Hoven et al. 2015; Flanagan et al. 2008). Specific variations to VSD have also been proposed, for instance care centered value sensitive design (CCVSD) aimed at using care practices and care values for evaluating the design of robots in care settings (Van Wynsberghe 2013). Some of these approaches call specifically for dynamically adapting to values that emerge through interaction with patients, after the technology is put in use (Poulsen and Burmeister 2019).

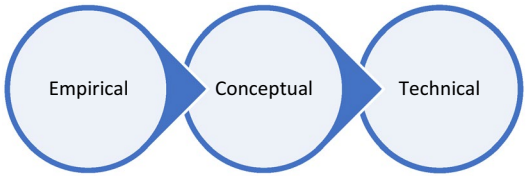

Fig. 1 VSD trip-partite methodology

At the core of VSD is a tripartite methodology of empirical, conceptual, and technical investigations (Friedman et al. 2006). *Empirical investigations* involve, among others, value elicitation and empirically investigating how relevant stakeholders perceive the values at stake. *Conceptual investigations* aim at conceptualizing the relevant values and identifying and dealing with possible value trade-offs. *Technical investigations* concern identifying value issues on the basis of existing technical designs as well as translating values into technical features. Although the three types of investigations can take place in parallel or iteratively, their description nevertheless seems to suggest a certain order, as shown in Fig. 1.

A somewhat similar process has been sketched by Flanagan et al. (2008) who distinguish three types of activities: (1) value discovery, aimed at finding out the most important values for a design task, (2) translation, aimed at operationalizing values, and translation values into technical features while addressing trade-offs and (3) verification, aimed at testing whether or not values have been embedded in the design as intended. While the VSD approach sketched by Flanagan et al. recognizes the iterative and non-linear character of the mentioned activities and the design process, it is still based on two assumptions that are important for our argument. The first assumption is that the mentioned activities are mainly to be undertaken during the design phase. The second (related) assumption is that the relevant values can be fully identified beforehand and can also be sufficiently embedded in the technology during the design phase.

The VSD approach and its various iterations have proven useful but run into limitations in the case of platform design. As explained in "Key characteristics of digital platforms" section, platforms are different from other technologies as they introduce not only new epistemic uncertainties, but also ontological uncertainties. How a platform will be used and

what ethical issues that usage will raise, and henceforth what values should be addressed in design, is, at least in part, indeterminate at the design stage. Moreover, although often the importance of certain values can be anticipated, it may turn out to be impossible to reliably embed these values into the technical features of the platform because, as we have sketched in "Control of digital platforms" section, use of digital platforms is in part beyond the direct control of the platform designers. Taking of all these considerations together, what Flanagan et al. (2008) call verification (of all the embedded values at play) may therefore not only be practically difficult but principally impossible. Given that VSD has shown promise in providing guidance to designers and engineers in an ICT context, we suggest expanding VSD to account for the additional uncertainties introduced by digital platforms.

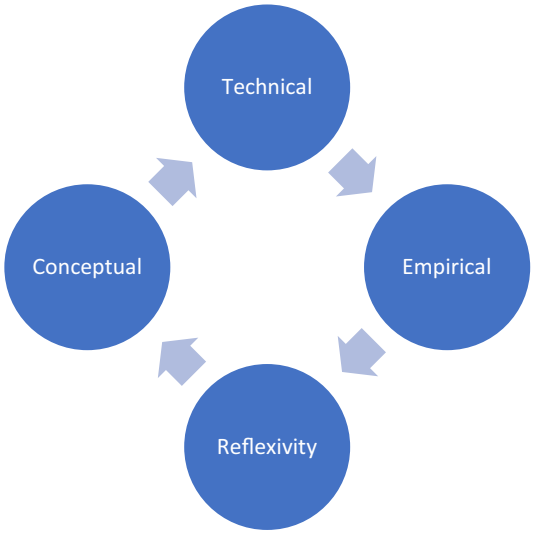

Fig. 2 The TERC model illustrating double loop learning for platform design

Expanding VSD for digital platforms: the TERC model

As we have discussed in "Background" section, platforms are characterized by modularity. This modularity gives platforms a certain flexibility, which seems to make it easier to adapt the capabilities of the platform during the later phases of their life cycle. On the other hand, platforms are also susceptible to what is known as 'increasing returns to adoption' (Arthur 1989), i.e. the more a platform is adopted the more other users have reasons to adopt it too, even if the platform is perhaps technologically inferior to competing ones. The reason for this is that platforms coordinate transactions between actors, which makes it attractive to use the platform that is already commonly used (or used more than others) (Katz and Shapiro 1985).

Technologies that are characterized by economies of increasing returns to adoption are also more likely to get locked-in, i.e. it becomes harder to switch to another technology (Arthur 1989). We witness this in the case of Facebook; there are not many competing social networking platforms and users stick to the platform, even if some of them might be critical of how Facebook, for example, addresses privacy. At the same time, the Cambridge Analytics scandal seems to make it clear that there are high costs to not address such ethical issues, or to do so too late. The benefits of VSD, and other similar approaches under the umbrella of responsible innovation (Owen et al. 2013), have been articulated in response to practices where ethical issues raised by new technology were only addressed after they had arisen. The idea was (and is) that VSD leads to a more proactive attitude in which ethical issues are addressed or even avoided by addressing values upfront, to address the ethical issues prior to the development and use of a platform.

In Sect. 2, we laid out our argument about why VSD falls short of addressing the specific challenges of platform design. Our argument does not imply that we should give up on addressing values in the design of platforms, but rather that some methodological innovations in the VSD methodology are required. More specifically, we suggest the following three expansions to the VSD methodology in the context of platform development (see Fig. 2):

1. *Extending* the required VSD investigations (empirical, conceptual, technical) to the *entire life cycle* of a platform rather than limiting them to the design phase.
2. *Introducing* a fourth type of activity (or investigation) into the tripartite framework that we call *reflexivity*. This new activity involves a reflective inquiry into whether or not current applications still help realize important values and/or raise new value issues that need to be addressed.
3. Introducing *new methods* for supporting the reflexivity. This is achieved through what we call *moral prototyping* or *moral sandboxing*.

We use the label *TERC model* to refer to the combination of the VSD skeleton and our three extensions. The remainder of this section discusses the three expansions of VSD.

Extending VSD investigations to the full life cycle of platforms

The exact way users and developers will interact with and build upon a platform is not pre-determined during the design stage. This means that new or unexpected uses and

add-on modules may arise over time, which create novel ethical issues or make new values relevant that were not (and could hardly have been) foreseen beforehand. Our suggestion is to conduct multiple iterations of VSD investigations, not only during the design, but throughout the entire life cycle of a platform.

VSD is already an iterative approach; however it is not necessarily an approach that extends to the whole life cycle of products (or platforms in our case). For example, Friedman et al. (2006, pp. 360–361) list eight essential features of VSD, but they do not include a whole-of-life-cycle approach as one of them.[1] Still we believe that the idea to extend VSD to the whole life cycle sits well with the seventh feature of VSD they mention, namely what they call the interactional stance: values are realized in the interaction between technology and society. Indeed, many VSD scholars already seem committed to a whole-life cycle approach. Nevertheless, we believe it is important to make this commitment more explicit and to formulate it as an essential part of the value-sensitive design of platforms.

There are multiple reasons for making this suggestion. First, extending VSD investigations to the full life cycle will allow platform providers to uncover relevant information, in real time, to inform decisions of platform developers. The constant updating of information is not only necessary for epistemic reasons but also because processes (like usage) that are still indeterminate at the design stage may become determinate (or change) in a later phase of the life cycle. Consider for example a platform like eBay that allows individuals or stores to sell goods to other individuals (or stores). In recent years there have been advertisements on the site to sell people: children (New York Post 2016), spouses (Mirror 2018), or boy/girlfriends (Metro 2018). Accordingly, continual iterations whereby values are a central feature in the acceptability of the platform (as opposed to functionality exclusively) enable the platform designers (and others) to identify ethical, and sometimes legal, issues and values that were not foreseen at the design stage, and to take additional measures to address these properly.

Second, extending VSD to the entire life cycle is a way to address to so-called Collingridge dilemma (Collingridge 1980). This dilemma states that in the early phases of technological development, technology is still malleable, but the consequences are largely unknown. In the later phases when the consequences become known, technology is often so entrenched that it becomes hard to change. Extending VSD to the entire life cycle allows for learning about consequences and addressing ethical issues along the way. The question that arises, then, is how to avoid that the technology has by then become so entrenched that it cannot be changed any longer.

We suggest that the way out of the Collingridge dilemma here is found in both extending VSD to the entire life cycle of platforms and ensuring enough flexibility in platform design. In this way, new insights can lead to adaptions along the way. This may not be easy to achieve but seems the only viable way forward if one wishes to address ontological uncertainties while recognizing the conundrum posed by the Collingridge dilemma.

Reflexivity throughout the platform life cycle

In order to account for some of the issues raised above (e.g. the discovery of new or competing values), alongside the difficulty in knowing how to deal with such value discoveries, we suggest an addition to the current tripartite framework of VSD, a stage we call 'Reflexivity'.

We understand 'reflexivity' here as a form of second-order learning. The distinction between first order and second-order learning is now common in the literature on learning and goes back to authors such as Schön and Argyris (e.g. Argyris and Schön 1978). First-order learning is learning within the bounds of existing belief and value systems. While first-order learning takes existing belief and value systems as given, second-order learning also involves adaptions to such beliefs and value systems.

We apply the distinction between first and second-order learning to technological design and the development of software systems. Then, first-order learning is basically learning how to better achieve the (taken for granted) goals of a technology (van de Poel and Zwart 2010). It may, for example, involve learning about the needs of stakeholders, or learning about how a technology is used in practice, or about what technical configurations work better than others. Yet, first-order learning does not imply learning about the goals, or values, of technological development. Second-order learning would imply a reflection on, and possibly a change in, the goals for which a technology is developed.

Similarly, in VSD first-order learning involves better understanding of the values of stakeholders, learning how certain technologies contribute (or do not) to certain values, among others. However it does not yet involve learning about the values revealed through the use of the technology; rather, it takes these values as given. Conversely, second-order learning involves learning about the values that designers should aim at in VSD. Such second-order learning, or reflexivity, may for example be triggered when it is revealed, for example, that a digital platform has unintended consequences and these consequences require attention to values that have not yet been addressed.

We propose to make reflexivity a separate activity within the expanded VSD methodology presented here, i.e. the

[1] Similarly the recent book by Friedman and Hendry (2019) nowhere explicitly discusses VSD as a whole-of-life cycle approach.

TERC model. The central question for this activity is: What are the values that we should aim at in the design of, or in future iterations of, this technology? By adding reflexivity as a separate activity, we draw attention to the fact that in this iterative process for carrying out VSD activities throughout the life cycle of the platform, it is not just about an update of information; rather, it is about the need to reflect on what new ethical issues have arisen (for example, the platform makes it possible to sell family members online), what new or additional values should be upheld, and how to do that.

Attaining reflexivity may also have consequences for who is to be involved in the design process. It is not a guarantee that engaging in multiple iterations will lead to the discovery of ethical issues in need of attention if the platform design team does not have the skills to do so or is 'too close' to the project. If, for example, the same design team is involved every step of the way, it may be difficult for these individuals to deep dive into the unintended consequences that may arise from a small change in user behavior. Such deep dives often require the skills of trained ethicists. There is already work done on the role of the 'values advocate' when using VSD approaches (Manders-Huits and Zimmer 2009) and on roles for ethicists as design team members in general (Van Wynsberghe and Robbins 2014). Each of these contributions point out the necessity of having an individual capable, and responsible, for investigating the impact of a technology (or design choice) on the expression and/or limitation of values. Hence, extending VSD investigations to include reflexivity also implies that ethicists should be involved not only in the design, but during the entire life cycle of the platform in order to attain reflexivity in a more systematic manner.

Methods for reflexivity: moral sandboxing

We suggest different mechanisms to implement reflexivity, depending on the platform life cycle. The methods originate from software development in a turbulent environment.

In the *design phase* of a platform, we suggest reflexivity through an activity of *moral sandboxing* (or moral prototyping): a mechanism to uncover value implications of a novel platform in an early stage, in a controlled environment. In software development, sandboxes are commonly used as a separate software environment that isolates the effect of experimentation (Wahbe et al. 1994). A sandbox is self-contained and should ensure that no damage can be done to the operational environment. To date, sandboxing is a term used in a technical capacity that refers to the testing of new additions in a safe environment before actually using it in a real-life environment. In this way, mistakes and unintended consequences can be identified and resolved before exposing the additions to the public. The idea of sandboxing is that users or developers can freely experiment with a software product without predefined scenarios or scripts. They are motivated to provide feedback about failures and report issues to developers. The provider of the software product can learn about unforeseen use scenarios and interactions with the system, which feeds into the design. In this way, interdependencies between new software modules and the existing ones can be addressed, and mistakes in the new or existing modules can be identified.

We propose the concept of 'moral sandboxing' as a way of uncovering value implications of a novel platform in an early stage and throughout the various VSD iterations. Whereas sandboxing is traditionally used to evaluate technical outcomes or user performance or usability, we suggest here to use it to discover moral elements. In this way, unforeseen value implications can be uncovered without real-life risks. In a sandbox environment, platform designers can invite and motivate third parties to develop new innovations, which permits the uncovering of unforeseen value implications in a controlled setting. Thus, in observing user behaviors, a map for understanding is created on which behaviors are aligned with which values, which can be plotted on a larger framework of ethical issues and principles. After a certain number of iterations it becomes more and more difficult to violate the values, or to reveal new undesired behaviors (or disvalues) and at that moment one can consider the iterations to be done.

Moral sandboxing allows for uncovering new values that need to be addressed (as either desirable or undesirable), value changes (in definition or in prioritization) or new threats to existing values. Moral sandboxing can thus contribute to what we have called reflexivity. Depending on how it is carried out, it might be considered a form of 'reflection-in-action', or reflection-on-action (Argyris and Schön 1978). The former is defined as thinking or reflecting in the midst of carrying out an activity while the latter means thinking about the practice *after* the event and turning that information into knowledge. The sandbox is used to observe behaviour and to give designers a chance for either reflection in or on action depending on the time constraints or the ability to make small changes as a solution.

At the *end of the design phase*, we suggest using beta versions for increased reflexivity. A beta version is typically understood as the version that is close to release and in some instances is distributed to a selected group of users for testing. The latter is called a closed beta. The beta users are motivated to report any issues. Sometimes financial incentives are provided to report issues. Only after the issues, are resolved the final version is released.

After launching the platform, dynamic adjustment and surveillance is recommended. The lack of predictability creates uncertainties if people will violate values. Tan and Sia (2006) propose 'advanced structuring' and 'dynamic adjustment' as strategies for managing outsourcing. Advanced structuring refers to introducing mechanisms to avoid having

value conflicts, whereas dynamic adjustment is about monitoring possible value conflicts and intervening when necessary. We follow their approach by advocating incremental and empirical approaches to test (undesired) effects and ensure platform structure avoiding undesired behaviors. In addition, we advocate for measures detecting anomalies and undesired behavior and having mechanisms to deal with them. Through dynamic adjustment and surveillance, platform providers continuously monitor and learn from innovations that third parties create. Whereas most platform providers today already monitor threats against known risks, our extension is novel as it allows uncovering novel value implications that may emerge suddenly over time.

Importantly, after the platform has been launched, it becomes difficult to make fundamental changes to the platform, as add-on modules and third parties become dependent on it. Hence, our view on dynamic surveillance entails making incremental changes rather than fundamental ones. Further, these incremental changes are largely related to the boundary of the platform rather than its core, which is to remain stable as much as possible (cf. Tiwana et al 2010). Such incremental changes at the boundary of the platform could for instance be adapting the level of openness of boundary resources (cf. Ghazawneh and Henfridsson 2013). Platform owners could restrict the conditions under which APIs of the platform can be used by third parties or add new APIs that give access to specific functions that are desirable to be promoted (cf. Eaton et al 2015). Alternatively, platform owners could refine or add new control mechanisms that steer the behaviour of third parties, including algorithmic control. One instance would be to change entrance rules or rating systems of third-party offerings, as discussed in the next section.

Illustration: ride-sharing platforms

To give an illustrative example, we consider ride-sharing platforms offered by so-called transportation network companies (TNCs) like Uber, Lyft, and Didi. A range of values is relevant for such platforms including transparency, accountability, safety, environmental sustainability, privacy and freedom from bias. Our focus here will be on the latter value, and other issues related to gender and race. We largely focus on Uber.

Freedom from bias in ride-sharing

Friedman and Nissenbaum (1996, p. 332) use the term bias "to refer to computer systems that systematically and unfairly discriminate against certain individuals or groups of individuals in favor of others." It should be noted that this definition is morally loaded as it refers to *unfair* discrimination. So freedom from bias is here not understood as implying that differential treatment is necessarily wrong (in fact sometimes it is justified or even good), but only that it is wrong if it is unfair or (morally) unjustified. Friedman and Nissenbaum (1996) further distinguish between pre-existing bias (that already exist in society), technical bias (due to certain technical constraints and choices) and emergent bias (that emerges out of use).

There exists some pre-existing bias when it comes to taxi services, e.g. where they operate. In some neighborhoods, it is much harder, or takes much longer, to get a taxi than in others, and there is growing evidence of several forms of discrimination by taxi drivers (e.g. Brown 2018). It might be argued that platforms like Uber have the potential to reduce such discrimination and bias. One particular technical feature of Uber is particularly relevant here: drivers and passengers do not get identifying information about each other (like race and/or gender) at the moment the system proposes certain rides (Uber 2019a, 2019b). It is only after both parties have accepted an offer that this information is exchanged (Uber 2019c). This feature is intended to limit bias on the part of consumer and driver concerning gender and/or race biases against the other.

A study by Brown (2018) suggests that TNCs like Uber and Lyft have improved taxi services in less well-off neighborhoods in Los Angeles. She finds that "ridehailing extends reliable car access to travelers and neighborhoods previously marginalized by the taxi industry" (Brown 2018, p. iii), and she suggests that "ridehailing provides auto-mobility in neighborhoods where many lack reliable access to cars" (Brown 2018, p. iii). She also finds that racial-ethnic differences in service quality (e.g. waiting times) in Los Angeles are much lower for TNCs than for traditional taxis, although some differences remain.

This does not mean that the system has no technical bias at all; one feature that has been mentioned is that the system requires users to have a credit card (Dallas Observer 2013), which obviously creates bias against certain groups, although there are ways to use the system without a credit card (WikiHow 2019). A study by Ge et al. (2016) carried out two randomized control trials in the Boston area and Seattle to investigate whether TNCs treat passengers of all race and gender equally. They found "a pattern of discrimination, which we observed in Seattle through longer waiting times for African American passengers—as much as a 35 percent increase. In Boston, we observed discrimination by Uber drivers via more frequent cancellations against passengers when they used African American-sounding names. Across all trips, the cancellation rate for African American sounding names was more than twice as frequent compared to white sounding names. …. We also find evidence that drivers took female passengers for longer, more expensive, rides in Boston (Ge et al. 2016)."

Interestingly, the study, which looked at UberX, Lyft, and Flywheel, also revealed a difference between Uber and Lyft: Lyft drivers see both the name and photo before accepting, or denying, a ride; while Uber drivers see these only after acceptance of a ride. As a consequence, in the study, Uber drivers much more often cancelled rides for passengers with African American-sounding names. The authors of the study suggest that TNCs like Uber might choose to completely omit personal information about potential passengers. They recognize, however, that this would leave open (and might even exaggerate) others forms of discrimination like not taking rides from certain neighbourhoods.

A study by Hanrahan et al. (2017) relates bias to the rating system of Uber. After a ride, the driver and passenger rate each other. This rating is, however, anonymous and ratings may be given without further explanation or justification. If drivers receive biased ratings from passengers (for example because of their skin color), such a rating will propagate through the system as Uber assigns work, and allow passengers to ask services, based on past ratings of drivers. Consequently, drivers are worried about low ratings and may suspect these to be based on bias. In response, some develop strategies to avoid biased ratings, like avoiding rides from certain areas or demographic groups (Hanrahan et al. 2017, pp. 11–12). In this way, the suspicion of bias by drivers may lead to an increased bias towards passengers, fueled by the rating system.

Another issue that has raised concern is violence against certain groups of passengers, in particular women. In 2017, the US law firm Wigdor LLP filed a lawsuit against Uber on behalf of women sexually assaulted by Uber drivers. They claim that "[s]ince Uber launched in 2010, thousands of female passengers have endured unlawful conduct by their Uber drivers including rape, sexual assault, physical violence and gender-motivated harassment." They do not mention a source for these numbers, although there is indeed quite some anecdotal evidence (e.g. Guo et al. 2018; Independent 2016). Female drivers have faced similar issues (BBC News 2019).

Evidence from other countries also suggests that violence against women is a serious issue. In China two women were raped and killed after using the platform Didi (Quarts 2018). As a consequence the platform suspended its car-pooling service Hitch in August 2018 and started to implement additional safety features (Wikipedia 2019).

The value of TERC

The examples above illustrate the unpredictability of users on a platform in terms of behavior, values manifest, values threatened, and values prioritized. To be sure, scholars and activists have suggested ways in which some of the above concerns could be addressed. Hanrahan et al. (2017) argue that platforms like Uber can become what they call a 'vehicle of basis' due to the lack of transparency and accountability of the rating system. They suggest two design strategies to overcome this: (1) a higher degree of transparency in rating and (2) tracing whether certain users systematically give biased ratings and lower the weight of their ratings in the overall rating score. Wigdor LLP suggests several actions that Uber could take including better and stricter selecting of drivers and tamper-free video cameras in all Uber vehicles. In 2018, Uber announced better screening of drivers, the ability to share rides with trusted contacts, and an emergency button (CNN 2018).

Our goal in developing the TERC model is to suggest that such mechanisms need not be an afterthought, nor should it be the task of outside scholars to suggest such measures for recourse. Instead, it should be the task of the platform provider to consistently and systematically ensure that the platform they have provided is able to account for changing behaviors and ethical norms. This, we suggest, can be achieved through the TERC model.

First, the TERC model clearly underlines the crucial importance of extending VSD to the entire life cycle. Despite the fact that the Uber platform may have some promising technical features to avoid bias and perhaps even reduce discrimination, in practice the platform seems to have become a vehicle for bias and discrimination. Moreover, experience suggests that not only the information exchanged at the moment that a ride is accepted is relevant, but also the rating system. Such insights might have been difficult to gather and fully grasp at the design stage but they seem to provide good reasons for attempts to redesign the platform to further design out bias.

The mentioned study by Hanrahan et al. (2017) is particularly relevant in this respect. It shows how the rating system has become a vehicle of bias. This is in large part due to particular usage practices as outlined in the mentioned study. Addressing these usage practices requires not just a technical redesign of the platform, but also changes in the governance of the platform, i.e. changes in how the rating system operates. By extending VSD to the whole life cycle, our TERC model enables such learning and adaptions as an integral part of platform design and governance rather than as an after-thought.

Second, such redesign would be helped by explicitly adding a reflexivity activity to the VSD methodology. As indicated, we understand reflexivity as second-order learning, i.e. learning that puts into question existing belief and value systems. We suggest that learning from the Uber experiences requires second-order learning in two respects. First, it requires recognizing the importance of a new value that is related to preventing the platform being used as a vehicle for violence against women. The value that is at stake here is neither fully covered by bias (i.e. the violence would not

become acceptable if it equally befell to men) nor is it fully covered by the value of safety (as the discriminatory element is part of what makes the violence so wrong). Second, second-order learning is also needed to recognize that the point is not only to design direct technical bias out (as Uber has attempted to do) but also about design measures to avoid the platform to become a vehicle of bias (e.g. through the rating system) and violence. This would seem to require a change in the belief system.

The importance of the third element, tools that enhance reflexivity, is a bit more speculative in this example. In the case of the Uber, moral sandboxing could be used to test a new version of the rating system, giving it to select groups as a way of capturing all the unethical things people might do (both drivers and passengers) in such situations. Moral sandboxing in such an instance is about rectifying the problem of bias in an earlier iteration of the platform (after the platform has been launched) while also attempting to mitigate negative consequences associated with a newer capability. While further exploration is needed to scope out the depth and breadth of moral sandboxing, nevertheless, there seems to be good reasons to suppose that tools like moral sandboxing and moral prototyping can be helpful in bringing to light some of the ethical and societal issues (and tragic experiences) that providers and citizens wish to avoid.

Conclusion

In this paper, we have argued that digital platforms, although providing value for users, create unforeseen consequences. The sheer complexity and scale of digital platforms may prevent an understanding of how they will evolve. Digital platforms are generative, in the sense that one can principally not foresee what modules will be added to a platform in the future, or how their users may interact in practice. This makes it hard or impossible even, to know which societal values will be affected. Whereas many systems essentially exhibit *epistemic uncertainties* which can, in principle, be reduced by increasing knowledge, digital platforms exhibit *ontological uncertainty*: even with full information and overview, opening up a digital platform to users and developers creates outcomes that can, in principle, not be foreseen. One of the more well-known approaches to design, VSD, aims at creating technologies which promote certain values and minimize threats to societal values. VSD, however, falls short when it comes to the dynamic nature of platform development, namely, its modularity and generality.

To deal with these challenges, we presented here the TERC model. Our model expands traditional VSD methods by: (1) extending VSD to the *entire life cycle* of a platform rather than limiting the focus to the design phase, (2) introducing reflexivity as a required activity for platform development, and (3) introducing moral sandboxing, beta releases and dynamic adjustment as new methods to facilitate reflexivity. Our TERC model is a first approach to acknowledge that value implications of digital platforms cannot be predicted up-front. By adding elements of reflexivity to VSD, it is a first to address the notion of ontological uncertainty, caused by the generative and mediating nature of digital platforms.

In an increasingly digitalized and complex society, our approach provides a basis for a new paradigm of reflexive approaches to VSD. Digital platforms are one example of a technology exhibiting ontological uncertainty, whereas more of those technologies are emerging. Consider for example robotics and artificial intelligence/machine learning for which the rules governing behavior are unknown to the programmers, or the development of open source software. We recommend testing our proposed TERC model in these fields and building a repertoire of resources to help platform developers and providers to deal with the rise in ontological uncertainties inherent to the emerging technologies of our time.

Funding Ibo van de Poel's contribution is part of the project ValueChange that has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme under grant agreement No 788321. Aimee van Wynsberghe's contribution was supported by the Netherlands Organization for Scientific Research (NWO), project number 275-20-054.

References

Argyris, C., & Schön, C. A. (1978). *Organizational learning, Addison-Wesley OD series* (Vol. 2). Reading, MA: Addison-Wesley.

Arthur, W. B. (1989). Competing technologies, increasing returns, and lock-in by historical events. *The Economic Journal, 99*(394), 116–131.

Baldwin, C. Y., & Woodard, C. J. (2009). The architecture of platforms: A unified view. In A. Gawer (Ed.), *Platforms, markets and innovation* (pp. 19–44). Cheltenham, UK: Edward Elgar.

BBC News (2019). 'Thrown to the wolves'—the women who drive for Uber and Lyft. https://www.bbc.com/news/technology-46990533. Accessed Sep 17, 2019.

Boudreau, K. J. (2012). Let a thousand flowers bloom? An early look at large numbers of software app developers and patterns of innovation. *Organization Science, 23*(5), 1409–1427.

Brown, A. E. (2018). Ridehail Revolution: Ridehail travel and equity in Los Angeles. PhD Thesis, UCLA (ProQuest ID: Brown_ucla_0031D_16839. Merritt ID: ark:/13030/m5d847t1.).

Bygstad, B. (2017). Generative innovation: A comparison of lightweight and heavyweight IT. *Journal of Information Technology, 32*(2), 180–193.

CNN (2018). Uber vows again to improve passenger safety. https://money.cnn.com/2018/06/04/technology/uber-passenger-safety/index.html. Accessed Sep 17, 2019.

Collingridge, D. (1980). *The social control of technology*. London: Frances Pinter.

Dallas Observer (2013). Southern dallas leaders say uber is profiling customers, and they want city hall to act. https://www.dallasobserver.com/news/southern-dallas-leaders-say-uber-is-profiling-customers-and-they-want-city-hall-to-act-7134986 . Accessed Sep 17, 2019.

Davis, J., & Nathan, L. P. (2015). Value sensitive design: Applications, adaptations and critiques. In J. van den Hoven, P. E. Vermaas, & I. van de Poel (Eds.), *Handbook of ethics and values in technological design* (pp. 11–40). New York: Springer.

De Reuver, M., Sørensen, C., & Basole, R. C. (2018). The digital platform: A research agenda. *Journal of Information Technology, 33*(2), 124–135.

Dow Jones Newswires (2019). The true costs of social media. https://www.morningstar.com/news/marketwatch/2019091140/the-true-cost-of-social-media. Accessed Sep 17, 2019.

Eaton, B., Elaluf-Calderwood, S., Sorensen, C., & Yoo, Y. (2015). Distributed tuning of boundary resources: The case of Apple's iOS service system. *MIS Quarterly: Management Information Systems, 39*(1), 217–243.

Eisenmann, T., Parker, G., & Van Alstyne, M. (2011). Platform envelopment. *Strategic Management Journal, 32*(12), 1270–1285.

Flanagan, M., Howe, D. C., & Nissenbaum, H. (2008). Embodying values in technology: Theory and practice. In: J. van den Hoven, & J. Weckert, J. (eds) *Information technology and moral philosophy* (pp. 322–335). Cambridge: Cambridge University Press.

Friedman, B., & Kahn, P. H. (2003). Human values, ethics and design. In J. Jacko & A. Sears (eds) *Handbook of human-computer interaction*. Mahwah, NJ: Lawrence Erlbaum Associates.

Friedman, B., & Nissenbaum, H. (1996). Bias in computer systems. *ACM Trans. Inf. Syst., 14*(3), 330–347.

Friedman, B., Kahn, P. H., & Borning, A. (2006). Value sensitive design and information systems. In: P. Zhang & D. Galletta (eds), *Human-computer interaction in management information systems: Foundations*. Armonk, NY: M.E, Sharpe.

Gawer, A. (2009). Platform dynamics and strategies: From products to services. In A. Gawer (Ed.), *Platforms, markets and innovation* (pp. 45–57). Cheltenham, UK: Edward Elgar.

Ge, Y., Knittel, C.R., MacKenzie, D., & Zoepf, S. (2016). Racial and gender discrimination in transportation network companies. *NBER Working Paper No. 22776*.

Ghazawneh, A., & Henfridsson, O. (2013). Balancing platform control and external contribution in third-party development: the boundary resources model. *Information Systems Journal, 23*(2), 173–192.

Goldbach, T., Kemper, V., & Benlian, A. (2014). Mobile application quality and platform stickiness under formal vs. self-control—Evidence from an experimental study. In *International Conference on Information Systems,* Auckland, New Zealand.

Guo, P., Tang, C. S., Tang, Y., & Wang, Y. (2018). Gender-based operational issues arising from on-demand ride-hailing platforms: Safety concerns, service systems, and pricing and wage policy. SSRN: https://ssrn.com/abstract=3260427

Hanrahan, B., Ning, M., & Wen., Y. C. (2017). The roots of bias on uber. *Proceedings of 15th European Conference on Computer-Supported Cooperative Work*.

Hanseth, O., & Lyytinen, K. (2010). Design theory for dynamic complexity in information infrastructures: the case of building internet. *Journal of Information Technology, 25*(1), 1–19.

Henderson, R. M., & Clark, K. B. (1990). Architectural innovation: The reconfiguration of existing product technologies and the failure of established firms. *Administrative Science Quarterly, 35*(1), 9–30.

Independent (2016). Uber driver accused of 32 rapes and sex attacks. Retrieved from https://www.independent.co.uk/news/uk/uber-drivers-accused-of-32-rapes-and-sex-attacks-on-london-passengers-a7037926.html. Accessed Sep 17, 2019.

Janssen, M., & Estevez, E. (2013). Lean government and platform-based governance—doing more with less. *Government Information Quarterly, 30*, S1–S8.

Katz, M. L., & Shapiro, C. (1985). Network externalities, competition, and compatibility. *The American Economic Review*, *75*(3), 424–440.

Manders-Huits, N., & Zimmer, M. (2009). Values and pragmatic action: The challenges of introducing ethical intelligence in technical design communities. *International Review of Information Ethics, 10*(2), 37–45.

Metro (2018). Man's joke about selling his girlfriend on Ebay backfires when she gets bids of £70,200. Retrieved from https://metro.co.uk/2018/10/06/mans-joke-about-selling-his-girlfriend-on-ebay-backfires-when-she-gets-bids-of-70200-8011201/?ito=cbshare. Accessed 17 September 2019.

Mineraud, J., Mazhelis, O., Su, X., & Tarkoma, S. (2016). A gap analysis of internet-of-things platforms. *Computer Communications, 89*, 5–16.

Mirror (2018). Fed up woman is selling her 'used husband' on eBay—and she doesn't want much money for him. Retrieved from https://www.mirror.co.uk/news/weird-news/fed-up-woman-selling-used-13788886. Accessed Sep 17, 2019.

Mukhopadhyay, S., De Reuver, M., & Bouwman, H. (2016). Effectiveness of control mechanisms in mobile platform ecosystem. *Telematics and Informatics, 33*(3), 848–859.

New York Post (2016). Someone tried to sell a baby on eBay for $5K. Retrieved from https://nypost.com/2016/10/12/someone-tried-to-sell-a-baby-on-ebay-for-5k/. Accessed Sep 17, 2019.

Ondrus, J., Gannamaneni, A., & Lyytinen, K. (2015). The impact of openness on the market potential of multi-sided platforms: A case study of mobile payment platforms. *Journal of Information Technology, 30*(3), 260–275.

Ouchi, W. G. (1979). A conceptual framework for the design of organizational control mechanisms. *Management Science, 25*(9), 833–848.

Owen, R., Bessant, J. R., & Heintz, M. (2013). *Responsible innovation: managing the responsible emergence of science and innovation in society*. Chichester: Wiley.

Poulsen, A., & Burmeister, O. K. (2019). Overcoming carer shortages with care robots: Dynamic value trade-offs in run-time. *Australasian Journal of Information Systems, 23*, 1–18.

Quarts (2018). Another woman was murdered while using Didi's carpooling service. Retrieved from https://qz.com/1370345/another-didi-hitch-ride-hailing-driver-murdered-a-female-passenger-in-china/. Accessed Sep 17, 2019.

Rochet, J. C., & Tirole, J. (2003). Platform competition in two-sided markets. *Journal of the European Economic Association, 1*(4), 990–1029.

Schaarschmidt, M., Homscheid, D., & Killian, T. (2018). Application developer engagement in open source platforms: An empirical

study of Apple iOS and Google Android developers. *International Journal of Innovation Management*. https://doi.org/10.1142/S1363919619500336.

Tan, C., & Sia, S. K. (2006). Managing flexibility in outsourcing. *Journal of the Association for Information Systems, 7*(4), 10.

Tilson, D., Lyytinen, K., & Sørensen, C. (2010). Research commentary—Digital infrastructures: The missing IS research agenda. *Information Systems Research, 21*(4), 748–759.

Tiwana, A., Konsynski, B., & Bush, A. A. (2010). Research commentary—Platform evolution: Coevolution of platform architecture, governance, and environmental dynamics. *Information Systems Research, 21*(4), 675–687.

Uber (2019a). Requesting a specific driver. Retrieved from https://help.uber.com/riders/article/requesting-a-specific-driver?nodeId=1aaf0913-484f-4695-9042-e61fc7613f24. Accessed Sep 17, 2019.

Uber (2019b). Getting a trip request. Retrieved from https://help.uber.com/partners/article/getting-a-trip-request?nodeId=e7228ac8-7c7f-4ad6-b120-086d39f2c94c. Accessed Sep 17, 2019.

Uber (2019c). How to identify a driver and vehicle. Retrieved from https://help.uber.com/riders/article/how-to-identify-a-driver-and-vehicle?nodeId=02746faf-1bc6-4d3f-8ba2-ab35f36d7191. Accessed Sep 17, 2019.

van de Poel, I., & Zwart, S. D. (2010). Reflective equilibrium in R&D networks. *Science, Technology & Human Values, 35*, 174–199.

van den Hoven, J., Vermaas, P. E., & Van de Poel, I. (2015). Handbook of ethics and values in technological design. Sources, theory, values and application domains. Dordrecht: Springer.

Van Wynsberghe, A. (2013). Designing robots for care: Care centered value sensitive design. *Science and Engineering Ethics, 19*(2), 407–433.

Van Wynsberghe, A., & Robbins, S. (2014). Ethicist as designer: A pragmatic approach to ethics in the lab. *Science and Engineering Ethics, 20*(4), 947–961.

Wahbe, R., Lucco, S., Anderson, T. E., & Graham, S. L. (1994). Efficient software-based fault isolation. *ACM SIGOPS Operating Systems Review, 27*(5), 203–216.

Wareham, J., Fox, P. B., & Cano Giner, J. L. (2014). Technology ecosystem governance. *Organization Science, 25*(4), 1195–1215.

Wessel, M., Thies, F., & Benlian, A. (2017). Opening the floodgates: The implications of increasing platform openness in crowdfunding. *Journal of Information Technology, 32*(4), 344–360.

WikiHow (2019). How to use Uber without a credit card. Retrieved from https://www.wikihow.com/Use-Uber-Without-a-Credit-Card. Accessed Sep 17, 2019.

Wikipedia (2019). DiDi. Retrieved from https://en.wikipedia.org/wiki/DiDi. Accessed Sep 17, 2019.

Yoo, Y., Henfridsson, O., & Lyytinen, K. (2010). Research commentary—the new organizing logic of digital innovation: an agenda for information systems research. *Information Systems Research, 21*(4), 724–735.

Yun, H., Kettinger, W. J., & Lee, C. C. (2012). A new open door: The smartphone's impact on work-to-life conflict, stress, and resistance. *International Journal of Electronic Commerce, 16*(4), 121–152.

Zittrain, J. L. (2006). The generative internet. *Harvard Law Review*, 1974–2040.