VALUE SENSITIVE DESIGN AND AGILE DEVELOPMENT: POTENTIAL METHODS FOR VALUE PRIORITIZATION

Till Winkler

Institute for Information Systems and Society,
Vienna University of Economics and Business (Austria)

till.winkler@wu.ac.at

ABSTRACT

A promising pathway towards increasing the recognition of value-oriented approaches outside of academia is to facilitate their integration into state-of-the-art agile development processes. This paper provides an overview on the similarities between value-oriented approaches and agile development processes. To facilitate an integration into agile development, light-weight value prioritization methods are needed. Several approaches to value prioritization are presented and insights from using a Likert-scale, as a scalable and light-weight method, in the context of a mobile navigation application are reported. It is the aim of this paper to inspire more empirical work in this area and to foster discussions about an integration into agile development processes and about the challenges of prioritizing values.

KEYWORDS: value sensitive design, value-based engineering, agile development, value prioritization, mobile navigation application.

1. INTRODUCTION

Software is a crucial element of digital technology and has become an integral part of our society (Andreessen, 2011). However, software is not neutral, but acts as a mediator for human values and biases, molding its own operational context, which in return shapes human perception and actions, creates new practices and subsequently ways of living (Verbeek, 2008). Mediation through software can have negative effects such as biases introduced by algorithms (O'Neill, 2016; Obermeyer & Mullainathan, 2019). This suggests that software developers have the responsibility to consider human values, potential for biases and ethical concerns during the development process. The idea to consider instrumental values during technology development, such as *efficiency* and *reliability*, is as old as technology itself (van de Poel, 2015). However, such instrumental values only represent a fraction of relevant human values. A value list aggregated by Winkler and Spiekermann (2019) mentions 355 values potentially important for sustainable technology development. Due to the context-sensitive nature of values (Steen & van de Poel, 2012) even such extensive lists can never be complete. This raises the question on how to choose and prioritize values during a value-oriented project.

Value sensitive Design (VSD) provides a framework for the integration of values with ethical importance into technology design (Friedman, Kahn, Borning, & Huldtgren, 2013; van de Poel, 2015). To achieve successful integration of human values into the design process, VSD employs an integrative and iterative tripartite methodology, consisting of conceptual, empirical and technical investigation (Friedman & Hendry, 2019). All three investigations are interdependent and inform each other mutually (Manders-Huits, 2011). Since its initial conceptualization, VSD has inspired several value-oriented approaches such as *Values at Play* (Flanagan et al. 2005), *Value-oriented and Culturally Informed Approach* (Pereira & Baranauskas, 2015), *Value-based Engineering* (Spiekermann & Winkler, 2020) and several others. All these subsequent approaches contribute to a growing body of knowledge on designing technology in accordance with human values.

Despite these methodological advancements, value-oriented approaches have not found widespread deployment in industry (Miller, Friedman, Jancke, Gill, 2007). Some scholars attribute this to a lack of light-weight methods (e.g. compatible with agile development), a shortage of methods for important tasks (e.g. for prioritizing values) and a lack of consistent methodological description (Miller et al. 2007; van de Poel, 2015; Burmeister, 2016). Additionally, the reliance on exhaustive stakeholder identification and participation, one key feature of value-oriented approaches, is also considered as a major obstacle (Manders-Huits, 2011).

In this paper, I argue for an integration of value-oriented approaches into agile software development processes to increase value consciousness within industry. As a first step, I point out similarities between value-oriented approaches and agile development processes and emphasise the need for value prioritization methods to facilitate an integration. Afterwards, I will present several potential methods for value prioritization. Finally, I report on results and insights from applying a light-weight method to value prioritization in the context of mobile navigation application development.

2. CONSIDERING VALUES DURING AGILE SOFTWARE DEVELOPMENT

The once popular sequential waterfall software development process is based on the assumption that software requirements can be fully specified upfront and that optimal solutions are predictable and plannable in advance; design and sub-sequential coding in this case does not start before a clear set of requirements is defined (Royce 1970). In practice, 4 out of 10 factors for project failure are related to problems with software requirements (Clancy 2014). The volatility requirements, the inability of users to formulate them upfront and the impossibility to know all software details in the beginning, are major obstacles for upfront planning (Mellis, Loebbecke, & Baskerville, 2010; Schmidt, 2016). Changing requirements emphasize the need for less formal and more flexible processes (Sommerville, 2010).

The iterative (or spiral) model to software development employs several adapted waterfall phases and adds risk assessment processes and prototyping activities (MacCormack, Verganti, & Iansiti, 2001). While iterative software processes are more flexible, they are still based on the assumption that software development can be a predictable and upfront plannable process (Schwaber, 1997). As the source of uncertainties is often outside of the development team's control, flexibility and adaptability to new situations is vital for successful software development (Schmidt 2016). Furthermore, upfront planning becomes nearly impossible as the fast evolution of technology often leads to the emergence of new practices and required skills during project realization (Schmidt 2016).

Agile development can be seen as a counter-reaction to the habit of adding more processes (e.g. risk management) to improve development, as it tries to avoid heavy-weight processes and complex documentation (Fowler & Highsmith, 2001; De Lucia & Qusef, 2010). While traditional approaches advocate extensive planning and codified processes, agile approaches rely on the development team's creativity to deal with unpredictable challenges (Dybå & Dingsøyr, 2008; Nerur, Mahapatra, & Mangalaraj, 2005). An initial goal of agile development, as stated in the "Agile Manifest" (Fowler & Highsmith, 2001), was to give developers the autonomy and flexibility needed to deal with challenges (e.g. changing requirements, uncontrollable uncertainties, emerging new practices) and to enable high quality software development. Furthermore, the goal was to strengthen the role of individuals (Fowler & Highsmith, 2001) by ascending them above being the entourage of a development process. Instead, agile development focuses on human collaboration during a development project. These goals produced desired results by increasing software and code quality, improving job satisfaction, raising productivity and customer satisfaction (Dybå & Dingsøyr, 2008).

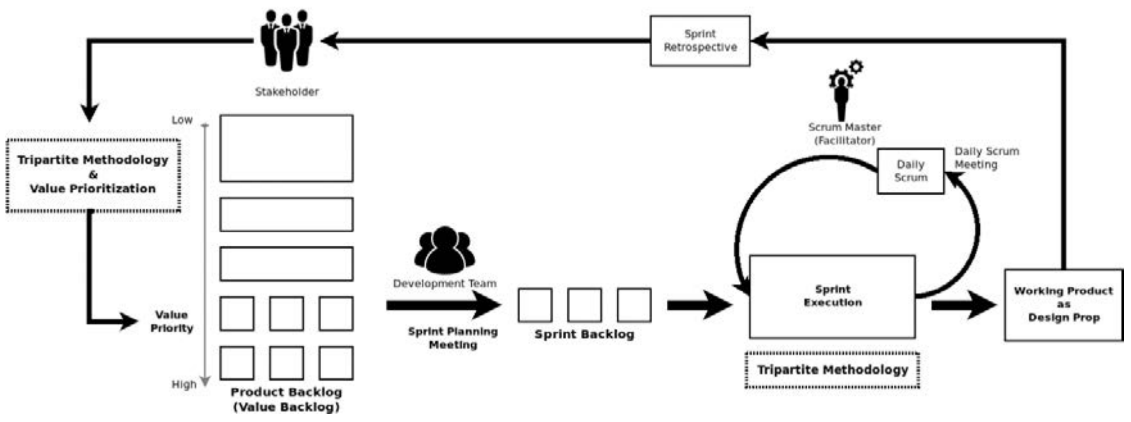
2.1. Value-oriented approaches and agile development processes

In agile development, essential requirement engineering activities are mingled together and performed iteratively throughout the whole development cycle (Paetsch, Eberlein, & Maurer, 2003). Such an iterative and integrative nature is also characteristic for the tripartite methodology of VSD (Friedman & Hendry, 2019). The general goal of any requirement engineering activity is to identify stakeholders and elicit, analyse, specify, validate, document and manage their requirements (Abran, Moore, Bourque, Dupuis, & Tripp, 2004; Sommerville, 2010). Achieving similar goals, the VSD tripartite methodology (conceptual, empirical and technical investigation) can be considered as a requirement engineering activity for human values. In VSD, direct and indirect stakeholders are identified and relevant values elicited and analysed during the conceptual investigation. The empirical investigation further analyses stakeholder perception, the use context of a system and validates relevant values. The technical investigation specifies how technology can support certain values, or what values are implicated by already existing features (Manders-Huits, 2011; Friedman & Hendry, 2019).

Agile development starts with a rough approximation of the final requirements and adds details during the whole development process (Rees, 2002). In a similar fashion, VSD's tripartite method adds details on important values during the whole development project. In general, the tripartite methodology seems to be a prime candidate for an integration into agile development processes. Figure 1 exemplifies a potential integration of the tripartite methodology into Scrum, which is one of the most common agile development processes (Sharma, Hasteer, 2016). In Scrum, the tripartite methodology can be used for developing and maintaining the product backlog - or "value backlog" in this case. Additionally, during each sprint the tripartite method can be used to further specify and understand values and related concepts.

In agile development, requirement-related activities are performed during face-to-face meetings in collaboration with stakeholders (Ramesh et al. 2010). In contrast to traditional practices (waterfall or iterative model), the development team and all stakeholders are involved throughout the whole development process in all relevant requirement engineering activities (Paetsch et al. 2003; Sillitti & Succi, 2005). Similarly, value-oriented approaches also heavily rely on stakeholder participation (Manders-Huits, 2011), as human values can best emerge during active stakeholder interaction (Borning & Muller, 2012). Especially after larger development cycles (or "Sprint Execution" in Figure 1), additional requirements are identified by presenting the working product to stakeholders. Presenting a working product as design prop can facilitate discussions about human values (Koch, Proynova, Paech, & Wetter, 2013), which could increase the value consciousness of a project and mitigate drawbacks of using prototypes (Reilly Dearman, Welsman-Dinelle, & Inkpen, 2005).

Figure 1. Integration of tripartite methodology and value prioritization into scrum.



In agile development, requirements are considered as being decoupled from each other, allowing a flexible order of implementation according to their priority (Silliti & Succi, 2005). At the beginning of each new development cycle, requirements (or values in Figure 1) are sorted by their priority (Product Backlog), discussed, and chosen for implementation (Sprint Planning Meeting in Figure 1; Sharma, Hasteer, 2016). It is questionable whether decoupling is suitable for value-oriented approaches, as treating values independently from each other prevents resolving value tensions and threatens the delicate balance between values (Friedman & Hendry, 2019). A solution can be to implement values one after the other according to their priority and resolving value tensions during the implementation. Another solution can be to use the conception of values by *Value-based Engineering* based on material value ethics (Hartmann, 1932; Scheler, 1913-1916/1973). According to Value-based Engineering, values are coming as value clusters consisting of core values and related value qualities. Value qualities are instrumental to their specific core value (Spiekermann & Winkler, 2020). For instance, the core value *trust* is supported by value qualities such as *openness*, *accountability*, *privacy* and others (Spiekermann-Hoff, Winkler, & Bednar, 2019). Implementing one value cluster after the other has the advantage that value tensions and the balance between values is encapsualed within a cluster. This tension is expressed in the relation between core values (e.g. *trust*) and related value qualities (e.g. *openness, accountability, privacy*). In any case, prioritizing values (or value cluster) at the beginning of each development cycle (or sprint) using an appropriate light-weight method is essential for a successful integration of value-oriented approaches into agile development processes.

2.2. The fall from grace of agile development

While in the beginning agile development lived up to the goal of giving developers the necessary autonomy and flexibility and increased software and code quality (Dybå & Dingsøyr, 2008), this is not the case anymore in today's industry practice. The promise of shorter development time and rapid results resonated highly in industry (Howard, 2002; Savolainen, Kuusela, & Vilavaara, 2010) and on the management floor. Especially the management saw agility as a tool to increase productivity and time pressure, which lead to an emphasis of the "lean" mentality of agile development adapted from "lean manufacturing" (Poppendieck and Poppendieck, 2007). Studies show that developers are willing to go beyond traditional functional requirements but do not have the necessary time or autonomy to do so (Bednar, Spiekermann, & Langheinrich, 2019; Spiekermann, Korunovska, & Langheinrich, 2018). The aim of agile development to strengthen the role of individuals and the foster collaboration (Fowler & Highsmith, 2001) has not become part of today's industry practice. While stakeholders are supposed to be included in all development steps, in practice, they are substituted by on-site representatives (Sillitti & Succi, 2005). In previous software development processes, developers were bound to satisfy a process, in agile practice they are bound to meet the deadline in time. *Value-based Engineering* explicitly acknowledges this industry practice by calling for "a more careful use of agile forms of system development" (Spiekermann & Winkler, 2020, p. 16) and requiring value-oriented projects to take the necessary time for value considerations.

In conclusion, due to the iterative and integrative nature of the tripartite method there is the potential for integrating value-oriented approaches into agile development processes. Such an integration can be of mutual benefit, as it raises the recognition of value-oriented approaches outside of academia and increases the success of software products by considering human values (Spiekermann 2016; van den Hoven, 2017). Furthermore, agile development could gain back its own original focus on stakeholder collaboration. However, caution seems advisable for value-oriented approaches as key aspects, such as strong stakeholder participation and value consciousness, should not be sacrificed.

3. METHODS FOR VALUE PRIORITIZATION

Methods for prioritizing values do not only enable an integration into agile development processes, but are important for any value-oriented project. For instance, considering all 355 values from a value list (Winker & Spiekermann, 2019) can hardly be achieved by a development team; the attempt to fulfil this many value obligations could lead to moral overload (van den Hoven, 2013). None of the unique 17 VSD methods (Friedman & Hendry, 2019) is explicitly recommended for the task of value prioritization. Potentially, the Value Dams and Flows methods (Miller et. al. 2007) could achieve a value prioritization by *excluding* objected design options (value dams) or *including* appealing design options (value flows) and thus implicitly giving priority to related values. The openness of VSD to use the entire range of quantitative and qualitative methods (Friedman & Hendry, 2019) creates the opportunity to employ standard social science methods such as laddering interviews, card sorting tasks or any type of ranking or scaling techniques including Likert-scales. Using a simple Likert-scale sounds especially promising, as it is a light-weight and scalable method, allowing an afford-less inclusion of many stakeholders.

While not explicitly mentioned as a unique VSD method, Burmeister (2016) uses his own approach to value prioritization. The "Burmeister method" focuses on the frequency and emotional intensity with which values are expressed. To my knowledge, this is the only approach that directly includes emotional aspects, therefore recognising that emotions are an important source of moral knowledge, understanding and awareness (Desmet & Roeser, 2015).

Value-based Engineering uses three complementary ethical investigations to prioritize values during its ethical exploration phase (Spiekermann & Winkler, 2020). During these investigations, the development team, the management and ideally all stakeholders are involved in value prioritization. The first investigation assesses the overlap between core values and the business model. During the second duty ethical investigation, personal maxims and values that each prioritizer wants to have as universal law are chosen. Furthermore, values that are not instrumental have a higher priority and the hierarchy of values from material values ethics (Hartmann, 1932; Scheler, 1913-1916/1973) is considered. The third investigation requires to check the values against existing corporate principles, legal frameworks, international human rights agreements or ethical principles. While recognising practical considerations (e.g. involving the management, considering business model) is promising, due to the amount of involved people, the necessary knowledge (e.g. material value ethics, legal frameworks, human rights) and amount of steps, this cannot be considered as a light-weight method.

In their paper, van de Kaa, Rezaei, Taebi, van de Poel, and Kizhakenath (2020) use a two-step approach to value prioritization based on expert opinions. First, experts qualitatively validate values from value lists and create a set of values for each stakeholder group. Secondly, the Best Worst Method (BWM) suitable for highly complex systems (van de Kaa et al. 2020) was used to assign weights (best and worst), determining the most important and least important values of a value set. The most important and subsequently the least important value is used to rate other values of a set using numbers between 1 (equally important) and 9 (extremely more important). Finally, the optimal weight (importance) for each value is derived. While such a type of analysis has been criticized for practically reducing all values into a single value of *utility* (van de Kaa et al. 2020), employing pairwise comparison to reduce the difficulty of the prioritization task seems reasonable. While the reduction of several values to *utility* must be seen critically, this step might make the benefits of value-conscious development graspable for the management and justify additional development time. Other potential methods for value prioritization that also focus on the utility are *discrete choice analysis* and *conjoint analysis* (Breidert, Hahsler, & Reutterer, 2006).

In summary, there are several methods available to prioritize values during a value-oriented development project. As the notion of agile development calls for light-weight methods, I decided to try out a simple Likert-rating scale.

3.1. Using a Likert-rating scale to prioritize values

Based on a value list (Winkler & Spiekermann, 2019), 48 values were identified by the author as relevant for the development of a mobile navigation application. As part of an online survey about mobility behaviour in Vienna, these values were presented in random order to participants. Overall, 264 participants rated values on a Likert-scale ranging from 1 (not important) to 7 (extremely important). The following question was presented to introduce this task: “How important do you consider the following attributes for the development of a mobile navigation app?” Participants were recruited via a university mailing list of the Vienna University of Economics and Business. Participants mean age is 26.34 years (± 6.77) of which 55.3% identified themselves as female and 44.7% as male. Of the initial 48 values, eighteen values had a mean below 4 (neither nor) and therefore were considered as unimportant. Most notably, the values *health*, *human well-being*, *dignity* and *justice*, initially considered as important by the author and considered as high values according to material values ethics, were considered as unimportant by the participants. Of the remaining 30 values, five technical values (reliability, usability, dependability, simplicity and efficiency) were considered as highly important achieving means higher than 6 (very important). Fifteen values, including privacy, trust, security, environmental protection and freedom from bias, received a mid-range importance between 6 (very important) and 5 (slightly important). The remaining 10 values achieved a rating between 5 (slightly important) and 4 (neither nor). Comments stated by the participants indicated high-level frustration and confusion during this rating task. Many participants were not able to see the connection between a mobile navigation application and for instance values such as health or human well-being.

These results are devastating for the idea to use the most light-weight and scalable method for value prioritization. In this case, especially technical values were considered as highly important, which seriously endangers the human and social value consciousness of a project. Simply letting stakeholders rate values, can therefore not be an appropriate method for value prioritization.

This bias towards instrumental values might be a sampling effect, caused by the participants' educational background as economy students. Such a bias could indicate an availability heuristic, leading people to consider recently discussed information or values (e.g. *efficiency*) as being more important than others (Wänke, Schwarz, & Bless, 1995). Implicit association tests assessing subconscious associations between concepts might be a more appropriate approach for further research in this direction (Greenwald, McGhee, & Schwartz, 1998). Rating values in an online setting facilitates fast responses but limits reflection upon the true importance of values. Serious play methods (Garde & van der Voort, 2016) to prioritize values (e.g. using bricks) could be another pathway to increase stakeholder reflection. As values are context-sensitive, setting a generic context (as was done here with the navigation application) could lead to the prioritization of a particular set of values. These preliminary results show the need for further empirical research to assess effects of stakeholder groups, used prioritization methods and established context on stated value priority.

4. CONCLUSION

An integration of value-oriented approaches into agile development processes has the potential to increase the recognition of value-conscious development outside of academia. Such an integration must be pursued with caution as stakeholder collaboration, a key feature of value-oriented approaches, is currently a practice not lived in industry. Furthermore, properly considering values during software development needs time, which is a scarce resource in the area of software development. Value prioritization methods can be a pathway towards successful integration into agile development processes. While light-weight methods would facilitate an integration, results presented here show that using a Likert-scale is not a suitable approach to prioritize values. However, there are several other methods, of which some might be capable of proofing the utility for value consciousness to managers.

REFERENCES

Abran, A., Moore, J. W., Bourque, P., Dupuis, R., & Tripp, L. (2004). Software engineering body of knowledge. IEEE Computer Society, Angela Burgess.

Andreessen, M. (2011). Why software is eating the world. *Wall Street Journal*, 20(2011), C2.

Bednar, K., Spiekermann, S., & Langheinrich, M. (2019). Engineering Privacy by Design: Are engineers ready to live up to the challenge? *The Information Society: An International Journal*, *35*(3), 122–142. https://doi.org/10.1080/01972243.2019.1583296

Burmeister, O. K. (2016). The development of assistive dementia technology that accounts for the values of those affected by its use. *Ethics and Information Technology*, 18(3), 185-198.

Breidert, C., Hahsler, M., & Reutterer, T. (2006). A review of methods for measuring willingness-to-pay. Innovative Marketing, 2(4), 8-32.

Clancy, T. 2014. "The Standish Group Report," Retrieved from: https://www.projectsmart.co.uk/white-papers/chaos-report.pdf, Jan 15, 2018

De Lucia, A., & Qusef, A. (2010). Requirements engineering in agile software development. Journal of emerging technologies in web intelligence, 2(3), 212-220.

Desmet, P. M., & Roeser, S. (2015). Emotions in design for values. van den Hoven et al, 203-219.

Dybå, T., & Dingsøyr, T. (2008). Empirical studies of agile software development: A systematic review. Information and software technology, 50(9-10), 833-859.

Flanagan, M., Howe, D. C., & Nissenbaum, H. (2005, April). Values at play: Design tradeoffs in socially-oriented game design. In *Proceedings of the SIGCHI conference on human factors in computing systems* (pp. 751-760).

Friedman, B., Kahn, P. H., Borning, A., & Huldtgren, A. (2013). Value sensitive design and information systems. *In Early engagement and new technologies: Opening up the laboratory* (pp. 55-95). Springer, Dordrecht.

Friedman, B., & Hendry, D. G. (2019). *Value sensitive design: Shaping technology with moral imagination.* Mit Press.

Fowler, M., & Highsmith, J. (2001). The agile manifesto. Software Development, 9(8), 28-35.

Garde, J. A., & van der Voort, M. C. (2016). Could LEGO® Serious Play® be a useful technique for product co-design? *Design Research Society*.

Greenwald, A. G., McGhee, D. E., & Schwartz, J. L. (1998). Measuring individual differences in implicit cognition: the implicit association test. *Journal of personality and social psychology*, 74(6), 1464.

Hartmann, N. (1932). *Ethics*. London: George Allen & Unwin.

Howard, A. (2002). Rapid application development: Rough and dirty or value-for-money engineering? *Communications of the ACM*, 45(10), 27-29.

Koch, S. H., Proynova, R., Paech, B., & Wetter, T. (2013). How to approximate users' values while preserving privacy: experiences with using attitudes towards work tasks as proxies for personal value elicitation. Ethics and information technology, 15(1), 45-61.

MacCormack, A., Verganti, R., & Iansiti, M. (2001). Developing products on "Internet time": The anatomy of a flexible development process. Management science, 47(1), 133-150.

Manders-Huits, N. (2011). What values in design? The challenge of incorporating moral values into design. Science and engineering ethics, 17(2), 271-287.

Mellis, W., Loebbecke, C., & Baskerville, R. (2010). Moderating effects of requirements uncertainty on flexible software development techniques. In International Research Workshop on IT Project Management.

Miller, J. K., Friedman, B., Jancke, G., & Gill, B. (2007, November). Value tensions in design: the value sensitive design, development, and appropriation of a corporation's groupware system. *In Proceedings of the 2007 international ACM conference on Supporting group work* (pp. 281-290). ACM.

Nerur, S., Mahapatra, R., & Mangalaraj, G. (2005). Challenges of migrating to agile methodologies. Communications of the ACM, 48(5), 72-78.

Obermeyer, Z., & Mullainathan, S. (2019, January). Dissecting Racial Bias in an Algorithm that Guides Health Decisions for 70 Million People. In *Proceedings of the Conference on Fairness, Accountability, and Transparency* (pp. 89-89). ACM.

O'Neill, C. 2016. *Weapons of Math Destruction. How Big Data Increases Inequality and Threatens Democracy.* Crown Publishing Group.

Reilly, D., Dearman, D., Welsman-Dinelle, M., & Inkpen, K. (2005). Evaluating early prototypes in context: trade-offs, challenges, and successes. IEEE Pervasive Computing, 4(4), 42-50.

Royce, W. W. 1970. "Managing the Development of Large Software Systems," Retrieved from: http://www. cs.umd.edu/class/spring2003/cmsc838p. Aug 08, 2017

Paetsch, F., Eberlein, A., & Maurer, F. (2003, June). Requirements engineering and agile software development. In WET ICE 2003. Proceedings. Twelfth IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, 2003. (pp. 308-313). IEEE.

Pereira, R., & Baranauskas, M. C. C. (2015). "A value-oriented and culturally informed approach to the design of interactive systems," *International Journal of Human-Computer Studies*, pp. 66-82.

Poppendieck, M., & Poppendieck, T. (2007). Implementing lean software development: From concept to cash. Pearson Education.

Ramesh, B., Cao, L., & Baskerville, R. (2010). Agile requirements engineering practices and challenges: an empirical study. *Information Systems Journal*, 20(5), 449-480.

Rees, M. J. (2002, December). A feasible user story tool for agile software development? In Ninth Asia-Pacific Software Engineering Conference, 2002. (pp. 22-30). IEEE.

Savolainen, J., Kuusela, J., & Vilavaara, A. (2010, September). Transition to agile development-rediscovery of important requirements engineering practices. In 2010 18th IEEE International Requirements Engineering Conference (pp. 289-294). IEEE.

Schwaber, K. (1997). Scrum development process. In Business object design and implementation (pp. 117-134). Springer, London.

Steen, M., & van de Poel, I. (2012). Making values explicit during the design process. *IEEE Technology and Society Magazine*, *31*(4), 63–72. https://doi.org/10.1109/MTS.2012.2225671

Sharma, S., & Hasteer, N. (2016, April). A comprehensive study on state of Scrum development. In *2016 International Conference on Computing, Communication and Automation (ICCCA)* (pp. 867-872). IEEE.

Sommerville, I. 2010. *Software engineering,* New York: Addison-Wesley.

Spiekermann, S. (2016). *Ethical IT innovation: A value-based system design approach*. Boca Raton: CRC Press.

Spiekermann-Hoff, S., Winkler, T., & Bednar, K. (2019). *A telemedicine case study for the early phases of value based engineering* (Working Paper Series/Institute for IS & Society No. 001_vs 1). *Working Paper Series/Institute for IS & Society*. Vienna: WU Vienna University of Economics and Business. Retrieved from https://epub.wu.ac.at/7119/

Spiekermann-Hoff, S., Winkler, T., & Bednar, K. (2019). *A telemedicine case study for the early phases of value based engineering* (Working Paper Series/Institute for IS & Society No. 001_vs 1). *Working Paper Series/Institute for IS & Society*. Vienna: WU Vienna University of Economics and Business. Retrieved from https://epub.wu.ac.at/7119/

Spiekermann, S., & Winkler, T. (2020). Value-based Engineering for Ethics by Design. *ArXiv, 2004.13676*. Retrieved from https://arxiv.org/abs/2004.13676

Scheler, M. (1973). *Formalism in ethics and non-formal ethics of values: A new attempt toward the foundation of an ethical personalism [1913-1916]*. (M. S. Frings & R. L. Funk, Eds.). Evanston, Ill: Northwestern University Press. https://doi.org/10.2307/2707101

Schmidt, C. (2016). Agile software development teams. Springer International Publishing.

Sillitti, A., & Succi, G. (2005). Requirements engineering for agile methods. In *Engineering and Managing Software Requirements* (pp. 309-326). Springer, Berlin, Heidelberg.

van de Poel, I. (2015). Design for values in engineering. *Handbook of Ethics, Values, and Technological Design: Sources, Theory, Values and Application Domains*, 667-690.

van de Kaa, G., Rezaei, J., Taebi, B., van de Poel, I., & Kizhakenath, A. (2020). How to weigh values in value sensitive design: A best worst method approach for the case of smart metering. Science and Engineering Ethics, 26(1), 475-494.

van den Hoven, J. (2017). Ethics for the digital age: Where are the moral specs? In Informatics in the Future (pp. 65-76). Springer, Cham.

Van den Hoven, J. (2013). Value sensitive design and responsible innovation. Responsible innovation: Managing the responsible emergence of science and innovation in society, 47, 75-83.

Verbeek, P. P. (2008). Morality in design: Design ethics and the morality of technological artifacts. In *Philosophy and design* (pp. 91-103). Springer, Dordrecht.

Wänke, M., Schwarz, N., & Bless, H. (1995). The availability heuristic revisited: Experienced ease of retrieval in mundane frequency estimates. Acta Psychologica, 89(1), 83-90.

Winkler, T., & Spiekermann, S. (2019). Human Values as the Basis for Sustainable Information System Design. *IEEE Technology and Society Magazine*, 38(3), 34-43.

Winkler, T., & Spiekermann, S. (2018). Twenty years of value sensitive design: a review of methodological practices in VSD projects. *Ethics and Information Technology*, 1-5.