Heinrich C. Mayr, Martin Pinzger (Hrsg.): INFORMATIK 2016,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2016 267

A Comparison of Value Sensitive Design and Sustainability Design

Stefanie Betz, Andreas Fritsch[1]

Abstract: Research and practice in different disciplines try to address technical, social, economical and environmental issues in software systems. This includes not only direct effects of a system and its features but also long-term and cummulative effects that are only shown over time. Nevertheless, in traditional software engineering these effects are usually not taken into consideration. This makes it hard to assess these long-term and cumulative impacts of a system. Therefore, solutions are needed to help software engineers to understand and assess the systemic effects of decisions taken in requirements engineering and systems development. In this paper we discuss and analyze two existing approaches, which provide concepts and methods for this task: Value Sensitive Design (VSD) and Sustainability Design (SD). The analysis shows that the two approaches have conceptual similarities such as the different sustainability dimensions (in SD) versus values (in VSD) and hierarchy of effects (in SD) versus direct and indirect stakeholders (in VSD). Altogether, SD provides a more holistic and structured conceptualization, while VSD – looking back at a longer tradition – can provide a range of experiences when it comes to integrating values in software engineering.

Keywords: Value Sensitive Design, Sustainability Design, Software Engineering, Sustainability, Value

1 Introduction

Software is the backbone of modern society as software systems are increasingly embedded in our society [Ch16]. Therefore, software designers need to be aware of the implication of the socio-technical systems they build. However, in traditional software engineering practice the long-term and cumulative impacts of a system are usually not taken into account and treated in isolation [Be15b]. For example, when designing a groupware system that supports knowledge sharing – balancing of human aspects like privacy and reputation of the involved individuals might well decide whether the software is eventually accepted by its intended users [Mi07]. Or think about the design of a procurement system, where a traditional approach would focus on financial return and supplier compliance and define the requirements accordingly. A holistic approach could integrate further aspects like providing visibility of a product's carbon footprint [Be15a]. One can imagine that this decision has far reaching and accumulative effects on the ecological performance of the respective organization. This might in turn affect consumer perception and thereby the long-term financial performance.

These examples illustrate that the discipline of Software Engineering needs to cope with multidimensional and long-term impacts of systems that have traditionally received little

[1] Karlsruhe Institute of Technology, Institute of Applied Informatics and Formal Description Methods (AIFB), Building 11.40, D-76128 Karlsruhe, {firstname}.{lastname}@kit.edu

attention [Be15b]. Thus, concepts and methods are needed to support understanding of possible effects of software systems and to provide a common ground for research and practice.

Several approaches have been proposed to consider (and balance) technical, social, individual, economical, and environemental concerns in software engineering (e.g. [Fr99], [HL07], [Be14], [Ra15]). Also, a number of conferences and workshops are emerging in this research area, e.g. ICT4S (http://ict4s.org), the SEIS Track at ICSE (http://2016.icse.cs.txstate.edu/seis) or the RE4Susy Workshop at the RE (http://web.csulb.edu/~bpenzens/re4susy/). This paper focuses on two approaches that explicitly provide concepts and methods to support understanding and provide a common ground: one is called Value Sensitive Design [Fr99] and the other Sustainability Design [Be14]. In the following, we are going to shortly introduce them and then discuss their similarities and differences. We believe that either of the approaches has its specific strengths (exemplary case studies versus a structured conceptual framework) and hope that a juxtaposition may inspire synergetic advancements in both communities and related research fields.

2 Value Sensitive Design

The term Value Sensitive Design (VSD) was first coined by Friedman to describe an emphasis of human values in technology design projects [Fr99]. It has since become a "branded term" for specific strategies and techniques concerned with human values [DN15] in technology design. In 2015, Davis and Nathan [DN15] published a summary of the field from which we draw in order to outline VSD.

According to VSD, a technology influences humanity in an "emergent and relational process" [DN15, p. 15]. The influential factors of this process are the design of the technology (e.g. product features), the context of its use and the involved stakeholders. A core assumption herein is that technology products can be improved by identifying and addressing human values in the design. Friedman defines the term *human value* as "what a person or group of people consider important in life" [FKB06, p. 349]. This broad notion is accompanied by a non-exclusive list of specific values named Human Welfare, Ownership and Property, Privacy, Freedom from Bias, Universal Usability, Trust, Autonomy, Informed Consent, Accountability, Courtesy, Identity, Calmness and Environmental Sustainability [FKB06]. Davis and Nathan [DN15] describe four *core commitments* of VSD. These can be seen as characteristics that may distinguish VSD from other design approaches. According to the authors, these characteristics are **proactive stance**, **interactional perspective**, **direct and indirect stakeholders** and **tripartite methodology**. VSD is **proactive** in that it asks researchers to consider human values during the design of a technology – rather than merely criticizing and analyzing existing technologies. The **interactional perspective** is taken by acknowledging that technology and values influence each other bidirectionally: on the one hand, a technology's design supports (or hinders) certain values and on the other hand, the usage of a technology is dependent on people's values. Furthermore, VSD distinguishes between **direct stakeholders and indirect stakeholders**. Direct stakehold-

ers are those who use a product or technology. Those who do not interact directly with a product or technology, but are influenced by others' use are indirect stakeholders.

The **tripartite methodology** of VSD consists of *conceptual, empirical and technical* "investigations" [FK02]. These investigations are seen as "iterative" and "integrative". During *conceptual investigations*, the affected stakeholders and values affected by a technology's use are to be identified. *Empirical investigations* apply methods like surveys or questionnaires (among others) to examine the relationship of technologies and their stakeholders with respect to human values. Lastly, *technical investigations* focus on the relationship between specific technological features and values – a concrete approach might be the design of a software that explicitly supports one of the human values listed above.

Several methods have been developed to support VSD [DN15]. Most of them are adaptations of design or social science methods, like fore example value-oriented mock-ups or value-oriented semi-structured interviews. Further methods were described by Davis and Nathan [DN15] as values-oriented analyses (*Direct and Indirect Stakeholder Analysis*, *Value Dams and Flows*) and a values-oriented toolkit (*Envisioning Criteria and Cards*).

In the following, we shortly illustrate some of the described concepts and methods using our first introductory example of a groupware system for the purpose of knowledge sharing (see [Mi07]). The system CodeCOOP was developed together with an industry partner to support software engineering knowledge sharing. During development of the system conceptual and empirical VSD-methods were employed: During *conceptual investigations*, *direct stakeholders* were identified by their roles as (among others) those who submit questions to the systems or those who answer questions. Examples for *indirect stakeholders* are managers or executives of the firm. The *empirical investigation* method *Value Dams and Flows* served to identify several tensions between critical values like privacy, reputation or trust. One outcome was for example that "queriers" fear that their reputation could be harmed when they ask a poor question. This insight led to the implementation of a feature that allows editing of posts to correct errors and improve quality.

3 Sustainability Design

Sustainability Design (SD) in the context of software engineering has been coined by Becker et al. in the Karlskrona Manifesto of Sustainability Design [Be14]. The manifesto is the central paper of this approach providing the main concepts, which are presented in the following. The core definition of *sustainability* that has been adopted for the manifesto is the simple and common "capacity to endure" [Be15b].

The main concepts the manifesto is presenting to approach sustainability are the **five dimensions** and the **three orders of effects**. The **five dimensions** are (1) individual sustainability, which aims at maintaining individual human resources (e.g. health, education), (2) social sustainability, which aims at preserving and improving the societies in their solidarity and services, (3) environmental sustainability, which aims at preserving the natural resources, (4) economical sustainability, which aims at retaining capital and added value, and (5) technical sustainability, which aims at maintaining and evolving information, sys-

tems, and infrastructure. These dimensions can be in conflict with each other [Be15c]. Thus, being technologically sustainable may have a negative effect on the economical sustainability.

The **three orders of effects** are: (1) the direct effects of the software system development and use; (2) the enabling effects that result from the ongoing use of the software system, and (3) the systemic changes caused by the long-time software system usage on a larger scale [Be15c]. The manifesto also follows the idea of not presenting concrete techniques but rather a set of *principles and commitments* for SD [Be14]. These are for example that sustainability is systemic, multidimensional and applies to a system and its wider context. Moreover, sustainability needs to be addressed interdisciplinarily and requires action from several levels, which interact with each other. Finally, sustainability is independent of the purpose of the system, requires long-term thinking, and can be achieved without cutting the needs of the future generations.

The manifesto ends with several suggestions for different stakeholders how to get started to achieve sustainable design. Researchers could identify and discuss research questions, customers and users can put the concern on the table and try to use sustainable products, education may revise the curricula and codes of ethics to include sustainability and finally, software practitioners should raise awareness and try to identify the effects on the different dimensions [Be14].

There exists some papers providing initial methods and techniques (e.g.*system scoping*, *stakeholder participation, stakeholder impact analysis*, *goal modelling*) to support sustainability design (see for example [Be15a], [Be15c], [Ch15]). These papers are focusing on requirements engineering as the authors of the manifesto see "requirements as the key to sustainability" [Be15a].

We now go back to our second initial example, a procurement system (see [Be15a]), to illustrate some of the listed methods, dimensions and effects: Discussing the purpose of the project the project team assesses possibilities to support sustainability development of the company emphasizing the effects that the procurement system can have on sustainability in all dimensions (*system scoping*). For example, the system can visualize the carbon footprint of products and facilitate the choice of providers who apply sustainable practice in the environemental dimension. Another action the project team conducts is to extend the number of stakeholders using a *stakeholder impact analysis*. Possible stakeholders involve for example local supplier representatives, service delivery organizations, process analysts, the CTO, and the strategic planning group. Surrogate stakeholder can be introduced to keep the number of stakeholders manageable. The visualization of the carbon footprint of a product within the procurement system qualifies as a *direct effect* along the environmental dimension of sustainability. This in turn enable users of the system to buy products with low carbon footprints (*enabling effect* and *economic dimension*). Taking a long-term perspective, this can lead to the systemic effect of reduced carbon footprints (*systemic effect* and *environmental dimension*) (see [Be15a]).

4 Discussion

First of all, one can identify several similarities between VSD and SD. The focus of both approaches is on ”thinking in a broader context”, when designing technology. Also, both approaches advocate the idea that technology has an impact on humanity. Thus, they not only take into account immediate effects and technical requirements but also think about effects in multiple dimensions for different stakeholders. Interestingly, both approaches use the idea of commitments for their design approaches rather than providing concrete techniques. But the inherent concept of sustainability design is much broader and more holistic than value sensitive design. Of course, this is already indicated when only looking at the names of the two approaches (value versus sustainability). Thus, SD is emphasizing the long-term effects of systems over time. Additionally, VSD is only emphasizing the “human values” such as human welfare, trust, privacy etc. and environmental sustainability. SD, in turn, has the already mentioned broader focus including not only individual, social, and environmental aspects but also technical and economical ones. This might help to not only integrate e.g. social or environmental aspects into software design, but also to identify and address tensions between these aspects and more “traditional” foci like efficiency and cost.

The sustainability dimensions can be understood as a way of structuring different kinds of values: Penzenstadler and Femmer propose a generic model for sustainability where values represent dimensions of sustainability [PF13]. In this view, the value privacy for example would be an aspect of the individual dimension of sustainability.

In this context it is also worth to mention, that the authors of the VSD approach have not determined and defined the human values for VSD. They only provided a seemingly rather arbitrary list of values. However, both approaches acknowledge systems thinking, the “hierarchy” of effects and the possible conflicts between different dimensions or values. Although, systems thinking is not explicitly mentioned in VSD and the hierarchy of the effects is only mentioned indirectly (and up to the second level) by referring to direct and indirect stakeholders.

Finally, there is one big difference of the two approaches and their use as a common ground for sustainability design. The manifesto is explicitly written to provide a definition and common ground for SD. VSD has evolved more and is based on different publications. We have provided a table on page 272 that provides a summary of the described characteristics of VSD and SD.

	Value Sensitive Design	**Sustainability Design**
Values/Dimensions	Human Welfare Ownership and Property Privacy Freedom from Bias Universal Usability Trust Autonomy Informed Consent Accountability Courtesy Identity Calmness Environmental Sustainability	Social Sustainability Economical Sustainability Environmental Sustainability Individual Sustainability Technical Sustainability
Impact	direct stakeholders indirect stakeholders	direct effects enabling effects systemic effects
Principles/Commitments	proactive stance interactional perspective direct/indirect stakeholders tripartite methodology	Sustainability is systemic Sust. has multiple dimensions Sust. transcends multiple disciplines Sust. is a concern independent of the purpose of the system Sust. applies to both a system and its wider contexts Sust. requires action on multiple levels System visibility is a necessary precondition and enabler for sust. design Sust. requires long-term thinking It is possible to meet the needs of future generations without sacrificing the prosperity of the current generation

Tab. 1: Value Sensitive Design versus Sustainability Design

5 Conclusion

We have presented VSD and SD as two approaches with somewhat similar objectives. Altogether, the described differences amount to VSD following a bottom up and SD a top down approach. The first started out with design projects and the task to consider human values in the process. And the latter was initialized with a definition of the challenge to be resolved (sustainability in software design) and is now applying these conceptualizations in design processes. Hereby, SD is more structured and holistic (e.g. by defining the five dimensions of sustainability as a structuring means for different values, where VSD has so far considered a seemingly arbitrary list of human values). However, VSD is looking at a longer tradition and can provide a range of experiences when it comes to integrating values in software engineering.

We invite researchers leaning towards either of the presented approaches to share experiences and ideas: both approaches offer concepts and methods to address related issues and both promise to offer valuable insights in order to improve the design of software systems. We think it is important that software engineers are aware of the major role software plays in our society and that they are responsible for long term effects of the system they engineer (see also [Be15a]). Future steps for us as researchers and software engineers are to further implement the concepts we presented in current software engineering practices and apply these in exemplary case studies. So, for example one could imagine to develop templates for requirements documentation and sustainability aware modelling languages.

References

[Be14] Becker, Christoph; Chitchyan, Ruzanna; Duboc, Leticia; Easterbrook, Steve; Mahaux, Martin; Penzenstadler, Birgit; Rodriguez-Navas, Guillermo; Salinesi, Camille; Seyff, Norbert; Venters, Colin; Betz, Stefanie: , The Karlskrona manifesto for sustainability design, 2014.

[Be15a] Becker, Christoph; Betz, Stefanie; Chitchyan, Ruzanna; Duboc, Leticia; Easterbrook, Steve M.; Penzenstadler, Birgit; Seyff, Norbert; Venters, Colin C.; Kocak, Sedef Akinli: Requirements: The key to sustainability. IEEE Software, (1):1–1, 2015.

[Be15b] Becker, Christoph; Chitchyan, Ruzanna; Duboc, Leticia; Easterbrook, Steve; Penzenstadler, Birgit; Seyff, Norbert; Venters, Colin C.: Sustainability design and software: The karlskrona manifesto. In: Proceedings of the 37th International Conference on Software Engineering-Volume 2. IEEE Press, pp. 467–476, 2015.

[Be15c] Betz, Stefanie; Becker, Christoph; Chitchyan, Ruzanna; Duboc, Leticia; Easterbrook, Steve; Penzenstadler, Birgit; Seyff, Norbert; Venters, Colin: Sustainability debt: A metaphor to support sustainability design decisions. In: Fourth International Workshop on Requirements Engineering for Sustainable Systems (RE4SuSy). Ottawa, Canada, August 2015.

[Ch15] Chitchyan, Ruzanna; Betz, Stefanie; Duboc, Leticia; Penzenstadler, Birgit; Easterbrook, Steve; Ponsard, Christophe; Venters, Colin: Evidencing Sustainability Design through Examples. 2015.

[Ch16] Chitchyan, Ruzanna; Becker, Christoph; Betz, Stefanie; Duboc, Leticia; Penzenstadler, Birgit; Seyff, Norbert; Venters, Colin: Sustainability Design in Requirements Engineering: State of Practice. In: ICSE'16: 38th International Conference on Software Engineering. Austin, Texas, USA, May 2016.

[DN15] Davis, Janet; Nathan, Lisa P.: Value Sensitive Design: Applications, Adaptations, and Critiques. In (van den Hoven, Jeroen; Vermaas, Pieter E.; van de Poel, Ibo, eds): Handbook of Ethics, Values, and Technological Design, pp. 11–40. Springer Netherlands, Dordrecht, 2015.

[FK02] Friedman, Batya; Kahn, Peter H. JR: Human values, ethics, and design. In: The human-computer interaction handbook. L. Erlbaum Associates Inc., pp. 1177–1201, 2002.

[FKB06] Friedman, Batya; Kahn, Peter H. JR; Borning, Alan: Value Sensitive Design and Information Systems. Human-computer interaction in management information systems: foundations, pp. 348–372, 2006.

[Fr99] Friedman, Batya: Value-sensitive design: A research agenda for information technology. Technical report, 1999.

[HL07] Hochheiser, Harry; Lazar, Jonathan: HCI and Societal Issues: A Framework for Engagement. International Journal of Human-Computer Interaction, 23(3):339–374, December 2007.

[Mi07] Miller, Jessica K.; Friedman, Batya; Jancke, Gavin; Gill, Brian: 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. ACM, pp. 281–290, 2007.

[PF13] Penzenstadler, Birgit; Femmer, Henning: A generic model for sustainability with process- and product-specific instances. In: Proceedings of the 2013 Workshop on Green In Software Engineering. Fukuoka, Japan, p. 3, March 2013.

[Ra15] Rashid, A.; Moore, K.; May-Chahal, C.; Chitchyan, R.: Managing Emergent Ethical Concerns for Software Engineering in Society. In: 2015 IEEE/ACM 37th IEEE International Conference on Software Engineering. volume 2, pp. 523–526, May 2015.