On the Role of Value Sensitive Concerns in Software Engineering Practice

Balbir Barn*, Ravinder Barn†, Franco Raimondi*
*School of Science and Technology, Middlesex University, London
{b.barn, f.raimondi}@mdx.ac.uk
†Royal Holloway University of London
{r.barn}@rhul.ac.uk

***Abstract*—The role of software systems on societal sustainability has generally not been the subject of substantive research activity. In this paper we examine the role of software engineering practice as an agent of change/impact for societal sustainability through the manifestation of value sensitive concerns. These concerns remain relatively neglected by software design processes except at early stages of user interface design. Here, we propose a conceptual framework and language concepts that will translate value sensitive design from its current focus in participatory design to one located in mainstream software engineering processes. Addressing this need will have an impact of societal sustainability and we outline some of the key research challenges for that journey.**

***Index Terms*—Value Sensitive Design, Values, Co-Design, Requirements elicitation**

I. Introduction

Sustainability concerns have been substantively critiqued from an environmental perspective since the 1960s, with the dominant, normative definition in the literature centering from that published in the United Nations Commission on Environment and Development (UNCED) report "Our Common Future" [1]: "*Sustainable development is development that meets the needs of the present without compromising the ability of future generations to meet their own needs*".

Debate around this definition has seen sustainability further discussed in terms of economic and social / societal sustainability [2] where the latter includes a focus on processes, systems and structures to support healthy and liveable communities. Others[1] have noted that sustainability is a social goal and ultimately there is no dichotomy between environmental and societal sustainability and this requires that "*social scientists working on these issues cooperate in interdisciplinary projects with scientists who assess the life-cycle-wide material and energy flows attributable to products and services, as well as ICT developers who understand the resulting requirements to be placed on ICT applications.*" [3, ,p19].

The National Research Council of USA reported on the centrality of the role of Computer Science research to support societal sustainability and identified a number of strategies and recommendations [4] based on the assumption that most sustainability issues require inter-connection and human interactions with systems. The report noted research in socio-technical systems is needed to encompass society, organisations and individuals and their behaviour rather than remain focussed on a narrow environment related definition of sustainability. Further, the role of information systems especially in their new guise of "apps" delivered through smartphones is particularly pertinent. Organisations responsible for these apps should recognise that their corporate actions with respect to design and deployment of such systems have a profound impact on all aspects of societal welfare.

Given that corporate actions are really an aggregation of individual behaviours, an examination of the root causes of human behaviour, particularly within the corporate context and in relation to the types of corporate actions that either benefit or harm society, is important [5]. A basic driver of human behaviour may at least in part be explained by a notion of "Value" [6]. Hence, a discussion about values in the design of socio-technical systems is necessary.

Following from this, we hypothesise that part of the problem of systems design is that the software engineering process (in any of its variants) does not include a first class representation of the concept of value in any of its constituent elements. A simple test of searching for the term "value" in one of the leading texts on software engineering (Somerville [7]) demonstrates its absence.

In this context, this paper suggests that human values are influencing factors in acceptance of socio-technical systems that are part of the necessary fabric for societal sustainability. Therefore consideration of values in the design of such systems can encourage societal sustainability. The paper examines the current evidence on how values are integrated in a *very* limited software engineering practice and offers essential characteristics for an approach for these integration requirements.

II. Related Work

The foundations for discussing how values could be integrated into the software engineering process has largely been initiated through the domain of human-computer interaction (HCI) in the seminal article by Suchman [8] and has influenced the advent of Co-Design, where questions about values "become easier since the collaboration between designer and client (or user) is explicitly recognised as a goal of the process."[9].

[1]Following communication from an anonymous reviewer.

A notable and sustained contribution to understanding and accounting for values in the design process is the work by Friedman and her colleagues on Value Sensitive Design (VSD)[10]. Research has informed the debate on the unintended consequences of systems and the detrimental effect or compromise of moral values that users may believe in.

The concept of *value* has been investigated in its role to systems design. Friedman is generally credited with introducing the study of values to IS practice through the Value-sensitive design (VSD) [11]. She defines value as: *what a person or group of people consider important in life*. Values that are particularly pertinent to information systems include: ownership and property; privacy, freedom from bias, universal usability, trust, autonomy, informed consent, identity and others. In systems design, *values* have, to-date, been integrated mostly with participatory design approaches [12] or more recently Co-Design. Co-Design involves potential (un-trained) end users working jointly with researchers and designers to jointly create artefacts that lead directly to the end product and, as Yoo et al. [13] note: "become a dominant user study methodology in the fields of product design, service design, interaction design and HCI [14]". Several researchers have commented that whilst participatory design is a dominant mode of technological design, end-users still struggle to influence the direction of design within the participatory process. Furthermore, end-users may not fully understand the ecology of available technologies. It may be that the reductionist principles of software engineering could be argued to have hindered the integration of values approaches into mainstream practice and so making it harder to monitor such concerns.

Value-sensitive design (VSD) emerged to integrate moral values (and more broadly ethics) with the design of systems. A key premise of VSD is that it seeks to design technology that accounts for human values throughout the design process (over and beyond the identification of functionality and visual appearance) of systems. Thus VSD has a stated goal that there should be freedom from bias in systems. That is: computer systems should not systematically and unfairly discriminate against certain individuals or groups of individuals in favour of others [11]. VSD has developed both methods and theory that incorporate particular values into technologies through conceptual, empirical, and technical investigations.

In the Yoo et al. model, the traditional co-design core blends methods from value-sensitive design to structure the co-design engagement with inputs from stakeholders and considerations of values. The co-design process may be initiated by free-form thinking, but their key innovation is in the introduction of two types of structured interventions. Designer prompts entail materials that originate from expert designers and may comprise personas, scenarios or the use of envisioning cards. Stakeholder prompts originate from the end-users and may utilise value based scenarios addressing concerns such as unintended uses of the system; changes of the use of the system over time and so on. Values will be, typically, those derived from the list suggested by Friedman in [15]. The reflection element of the model provides a way of representing how prompts may be generated by either a stakeholder or a designer as a result of joint participation in the co-design space.

In software engineering, Goguen's latter work in requirements elicitiation has emerged from formal computer science but presents a similar discourse on the importance of values that could be seen as a plea to move from a reductionist principle to one that proposes "semi-formal approaches that take account of social processes can be valuable. Values are the key to unlocking ...the enormous dangers of contemporary technologies." [16]. Goguen recognized that there are limitations to formalisms but also noted that ethno-methodologies present challenges in collection of data and the subsequent grounding of data in the design of the target technical artefact. Thus "methods based on ethnomethodology cannot be applied directly to systems that have not yet been built". Other approaches to consideration of values in software enginnering focus on return on investment calculations and therefore are not the type (moral) values that we propose also need to be addressed in the software engineering process [17], [18].

III. A Proposal for a Conceptual Framework for Integrating Values and Requirements

Currently there are no formal or semi-formal models and their associated processes for integrating values and requirements into the software engineering design process. Building on the work of Yoo et al. [13], we have developed a co-design workshop method that exposes value concerns such as security and privacy. The approach has been applied on a research project (Mobile Apps for Youth Offending Teams – MAYOT) aimed at developing social technology for use by young people and their caseworkers in youth offending teams in the UK. The project raised requirements on design methods to incorporate the voice of stakeholders with respect to privacy issues and an outcome of that work is the conceptual framework by deriving values and their resolution to value conflicts between stakeholders for integration into the software engineering design process.

The Co-Design activities in the various workshops yielded a rich set of data including design and specification of features/functions of the MAYOT app. We focus on one design feature. The *Exclusion Zone* feature is a function that is available on the MAYOT app that allows a case worker to define a geographic region from which a young person is prohibited (with potential risks to violating their youth order with obvious detrimental effects). The feature alerts the young person in possession of the smart phone hosting the app that they are in an exclusion zone. During the course of a number of workshops, the feature went through a number of design changes representing the designer's view, a case worker's view, and views from young people who believed that the prototype app feature violated their privacy. Such concerns were considered seriously by the research team to ensure that a balance was struck between relative privacy and security of information. Further, there was a clear challenge that if we are to incorporate value sensitive requirements into the design

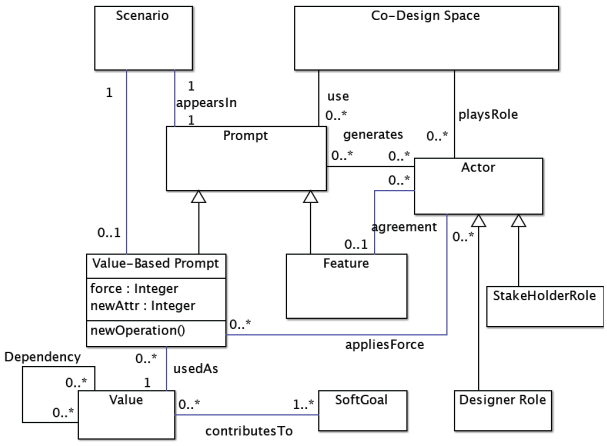

Figure 1. Conceptual Model

of software then we need a way of understanding, managing and exploring the impact of values through the requirements process.

A. Language Model

Our starting proposal posits that a more computationally useful understanding for incorporating values into the design process is to treat the analysis of value sensitive concepts as a model-based language design activity. Such an approach will utilise a suitable meta-language that can represent the various features of language such as the abstract syntax (defining the information structures to represent essential aspects of the language in a form suitable for machine processing); the concrete syntax (the appearance of the language on the screen or page) the various mappings necessary to relate the abstract syntax to the concrete syntax and the semantic domain in a form similar to that in [19]. Simple Unified Modelling Language (UML) class models and associated constraints can be used as a suitable meta-language for representing the language components listed above. Figure 1 shows the abstract syntax for the essential features of value concepts drawn as a UML class diagram based on the language design approach used in [19, pp. 53-54].

The model represents key concepts that we have encountered during our efforts at enacting value sensitive co-design with respect to prompts and how they are utilised.

We limit the semantics of this model to be a collection of traces. A trace is a sequence of co-design events (workshops). Each event is described by a system state change represented by object diagram that are instances of the semantic model. The semantic model comprises objects and slots that contain values. Additionally there are well-formed rules that determine how an instance model is deemed to be correct with respect to the conceptual model.

A *Co-Design Space* is any space where key Ac*tors*, the *Designers* and *Stakeholders* (including indirect stakeholders) participate in joint actions to create or evolve design features of intended systems. Actors are the source of the generation of *Prompts*.

Prompts are mechanisms or props that provoke reflection in the *Co-Design Space*. They are the main concept that needs to be tracked with respect to engendered values in the requirements elicitation and design process. *Prompts* can ultimately become a *Feature* of the system and become tools for describing how functions of a system may behave by appearing in scenarios of use. As noted by Yoo et al [13]. *Scenarios* incorporate *Values* and thereby bring *Values* into the co-design process. A Value may be of a particular type (e.g. Moral, Privacy etc.) and have properties such as name and description and relationships with other concepts that capture the provenance of a value. A *Prompt* may become a *Value Based Prompt* (i.e. it raises concerns or addresses a value of interest to a stakeholder). A *Value Based Prompt* ultimately requires some resolution as different actors may attribute different measures to the same value (c.f. the value of privacy and autonomy associated with the "Exclusion Zone" prompt). To cater for this, we propose that we integrate components from the Contextual Design participatory design method [20]. In particular, the Contextual Design method utilises the notion of a Cultural Model that provides an analytical capability for showing the cultural or political forces in the organisation. The Cultural Model also allows for values and for different forces to be applied by stakeholders to those values in order to arrive at a resolved status for a value. We account for this by *appliesForce* association between *Actor* and *Value-Based Prompt* and a function that computes the net *force* of a *Value Based Prompt*.

It is only when the net forces applied are balanced that the prompt becomes a feature. Forces that are applied to "balance" value sensitive concerns can be expressed in an arbitrary unit. To support the model we propose some sample constraints:

Constraint:R1: Not all prompts may become features. In the illustrative case study, a designer prompt aimed at making available individual intervention plans from the MAYOT app was rejected by case workers as potentially an example of a privacy breach and therefore value-sensitive.

Constraint:R2: A feature must be agreed by all participating stakeholders. This raises some other issues such as the availability of all stakeholders throughout the co-design process. To overcome this we propose that a feature is accepted by all stakeholders in at least one co-design workshop.

Constraint:R3: If a feature incorporates a value then the associated value-based prompt must have a net force of nil.

Constraint:R4: The same prompt may have a different generator role in a different co-design space.

IV. Discussion

We discuss our proposal by raising two open questions and offer an initial response. (1) What purpose can a framework for managing value sensitive concerns serve towards a sustainable society? We suggest that developing a clear, causal, traceable link that chains together stakeholders, value concerns raised by stakeholders, the scenarios and space in which concerns were

raised and the feature that is impacted can provide a more nuanced approach for determining priorities found in more traditional requirements engineering approaches. In particular, values that impinge or impede sustainability concerns can be surfaced. Key to this, is recognising that stakeholders can include networks that go beyond the designers and users of the system. For example a system designed for a security organisation may meet the values of both designers and the immediate users. However, (privacy) values that reside amongst in-direct stakeholders could be violated and such stakeholders may make behavioural changes that have an impact on environmental sustainability. Given that sustainability is about multi-generational issues, how to represent un-born generations in the design process is a challenge. Two options are possible: firstly one could involve stewardship from a state advocacy policy and recognise that participatory design is not always appropriate; or secondly, given that we now have inter-generational use of IT systems, research instruments that can identify and evaluate generational concerns can be developed.

(2) How does the framework enhance existing practice of analysis and design? From software engineering, requirements elicitation/management perspective, a benefit of tracking value concerns is the opportunity provided for early evaluation of a particular concern. Other research that we have undertaken examines how values related to privacy can be subject to early evaluation through the use of expert domain knowledge encoded in a Bayesian network [21]. Systems thinking that has evolved from the participatory design school has had a traditional strength in examining the socio-technical elements of systems design but artefacts from that approach are partially lost in translation in the task of implementation. We suggest that such a translation loss may be attributed to a lack of appropriate languages that can bridge different domains. Language based approaches lend themselves to model checking and verification processes.

V. Conclusion

The process of creation of information systems involves stakeholders, both direct and indirect who approach systems design activities with a pre-conceived set of values (such as privacy). These values have a critical influence on the design process and have not yet successfully been translated into current software engineering practice. This paper has posited a conceptual model for beginning this translation. The model has been induced from a single but rich case study from an ongoing research project aimed at developing mobile apps to support engagement between young people and their case workers in the UK youth justice system. The limitations of the model and the case study approach are apparent, none the less, the research presented can be seen as a call to arms and a vision for the SE community to recognise how systems impact on society at large. Currently the language has been formulated as an abstract syntax but preparations are underway to develop appropriate tool support using domain specific modelling technologies such as Meta-edit+ and to also explore how methods such as the Architecture Tradeoff Analysis Method (ATAM) [22] can be extended to include value concerns. A further branch of work is exploring the role of value concerns in enterprise modelling languages and business motivation modelling. Given the role that corporations play in constructing communities we consider that may also be a fruitful avenue for research in societal sustainability.

References

[1] G. H. Brundtland, *World Commission on environment and development: our common future*. Oxford University Press, 1987.

[2] S. McKenzie, *Social sustainability: towards some definitions*. Hawke Research Institute, University of South Australia, 2004.

[3] L. M. Hilty and T. F. Ruddy, "Sustainable development and ict interpreted in a natural science context: The resulting research questions for the social sciences," *Information, Communication & Society*, vol. 13, no. 1, pp. 7–22, 2010.

[4] L. I. Millett, D. L. Estrin *et al.*, *Computing Research for Sustainability*. National Academies Press, 2012.

[5] J. Marcus, "Human values and corporate actions propensity: Examining the behavioural roots of societal sustainability," *Business & Society*, p. 0007650312448891, 2012.

[6] E. A. Locke, "The motivation sequence, the motivation hub, and the motivation core," *Organizational behavior and human decision processes*, vol. 50, no. 2, pp. 288–299, 1991.

[7] I. Sommerville, "Software engineering. international computer science series," 2004.

[8] L. Suchman, "Do categories have politics? the language/action perspective reconsidered," in *Human values and the design of computer technology*. Center for the Study of Language and Information, 1997, pp. 91–106.

[9] C. A. Le Dantec and E. Y.-L. Do, "The mechanisms of value transfer in design meetings," *Design Studies*, vol. 30, no. 2, pp. 119–137, 2009.

[10] B. Friedman, "Value-sensitive design," *Interactions*, vol. 3, no. 6, pp. 16–23, 1996.

[11] B. Friedman and H. Nissenbaum, "Bias in computer systems," *ACM Transactions on Information Systems (TOIS)*, vol. 14, no. 3, pp. 330–347, 1996.

[12] G. Bjerknes, P. Ehn, M. Kyng, and K. Nygaard, *Computers and democracy: A Scandinavian challenge*. Gower Pub Co, 1987.

[13] D. Yoo, A. Huldtgren, J. P. Woelfer, D. G. Hendry, and B. Friedman, "A value sensitive action-reflection model: evolving a co-design space with stakeholder and designer prompts," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 2013, pp. 419–428.

[14] M. J. Muller, "Participatory design: the third space in hci," *Human-computer interaction: Development process*, pp. 165–185, 2003.

[15] B. Friedman, P. H. Kahn Jr, and A. Borning, "Value sensitive design and information systems," *Human-computer interaction in management information systems: Foundations*, vol. 5, pp. 348–372, 2006.

[16] J. A. Goguen, "Semiotics, compassion and value-centered design," *Virtual, Distributed and Flexible Organisations: Studies in Organisational Semiotics, Reading, UK*, pp. 11–12, 2003.

[17] B. Boehm, "Value-based software engineering: reinventing," *ACM SIGSOFT Software Engineering Notes*, vol. 28, no. 2, p. 3, 2003.

[18] M. Heindl and S. Biffl, "A case study on value-based requirements tracing," in *Proceedings of the 10th European software engineering conference held jointly with 13th ACM SIGSOFT international symposium on Foundations of software engineering*. ACM, 2005, pp. 60–69.

[19] B. S. Barn and T. Clark, "A domain specific language for contextual design," in *Human-Centred Software Engineering*, R. Bernhaupt, P. Forbrig, J. Gulliksen, and M. Lárusdóttir, Eds. Berlin Heidelberg: Springer, Oct. 2010, vol. 6409, pp. 46–61. [Online]. Available: http://dl.acm.org/citation.cfm?id=1939212.1939219

[20] H. Beyer and K. Holtzblatt, *Contextual design: defining customer-centered systems*. Access Online via Elsevier, 1997.

[21] B. S. Barn, R. Barn, and G. Primiero, "An approach to early evaluation of informational privacy requirements," in *Proceedings of the 30th Annual ACM Symposium on Applied Computing*. ACM, 2015, to appear.

[22] R. Kazman, M. Klein, M. Barbacci, T. Longstaff, H. Lipson, and J. Carriere, "The architecture tradeoff analysis method," in *Engineering of Complex Computer Systems, 1998. ICECCS'98. Proceedings. Fourth IEEE International Conference on*. IEEE, 1998, pp. 68–78.