Accepted to the International Conference on Product-Focused Software Process Improvement (PROFES 2024).
Accepted preprint. Article first published in Lecture Notes in Computer Science (LNCS) by Springer Nature.

Supporting value-aware software engineering through traceability and value tactics

Rebekka Wohlrab[1,2], Marc Herrmann[3], Christopher Lazik[4], Marvin Wyrich[5], Inês Nunes[6], Kurt Schneider[3], Lucas Gren[1], and Robert Heinrich[7]

[1] Chalmers University of Technology and University of Gothenburg, Sweden
wohlrab@chalmers.se, lucas.gren@cse.gu.se
[2] Software and Societal Systems Department, Carnegie Mellon University, Pittsburgh, PA, USA
[3] Software Engineering Group, Leibniz University Hannover, Hannover, Germany
marc.herrmann@inf.uni-hannover.de, kurt.schneider@inf.uni-hannover.de
[4] Software Engineering Group, Humboldt-Universität zu Berlin, Berlin, Germany
lazikchr@informatik.hu-berlin.de
[5] Chair of Software Engineering, Saarland University, Saarbrücken, Germany
wyrich@cs.uni-saarland.de
[6] Unaffiliated Researcher, Frankfurt, Germany
ir.nunes@campus.fct.unl.pt
[7] KASTEL - Institute of Information Security and Dependability, Karlsruhe Insitute of Technology, Karlsruhe, Germany
robert.heinrich@kit.edu

Abstract. Understanding human values like honesty, security, power, and fairness is crucial for designing trustworthy software systems. Recently, the software engineering community has recognized the importance of considering human values when developing software systems. However, there is no mature approach to considering human values as a first-order concern in software engineering. The exact meaning of such values is often vague or unclear, which makes it difficult to treat them systematically and break them down into traceable requirements, implementation, and testing artifacts.
In this vision paper, we propose a systematic approach to seamlessly integrate human values into the requirements engineering process. Concretely, we introduce the concept of "value tactics", which are tangible design decisions crafted to uphold certain values explicitly. We illustrate the approach with a running example and conclude with a research agenda for advancing value-aware software engineering.

Keywords: human values · software process · requirements engineering

1 Introduction

Eliciting high-quality software requirements from end users is complex and crucial [5], requiring a deep understanding of their desires and objectives [19]. Those

desires and objectives often relate to human values like honesty, security, or fairness [15]. Psychology has long studied human values, exemplified by Schwartz's theory, which includes ten fundamental values and their subvalues [10, 12].

Human values are also important throughout the software development lifecycle and in particular, in requirements engineering. In this paper, we focus on requirements engineering, as it is crucial to adequately understand human values when trying to uncover users' needs and expectations. Yet, in current requirements engineering practice, the underlying values of users are often neglected during elicitation [15], partly due to their imprecise and intangible nature and because in the past, requirements engineers tended to quickly move from problems to solutions. We recognize that understanding stakeholders' needs and aligning them with system requirements is complex and time-consuming. However, grasping the *why* behind requirements can offer significant advantages throughout the software development lifecycle.

In this vision paper, we describe how human values can be considered as a first-order concern throughout the requirements engineering process. We propose an approach that can be integrated into existing software processes and ensures that requirements, implementation, and testing artifacts are traceable to *why* a system's behavior and structure are designed in a certain way. We call our approach "value-aware requirements engineering", emphasizing its unique focus on enhancing the awareness of stakeholders' values, surpassing previous efforts in human-centered design [2]. Currently, formulated system requirements are usually decoupled from a description of the initial stakeholders' intentions. Developers and testers who were not involved in eliciting requirements could, however, benefit from an understanding of *why* those requirements were formulated; such understanding can lead to a system more aligned with end users' desires. Additionally, explicit consideration of human values can enhance requirements elicitation itself [1, 16]: As different values can be in conflict or interrelated [10], this can explain conflicts in the requirements of different stakeholders and in turn support an informed negotiation and prioritization of these requirements. Finally, business stakeholders increasingly prioritize values like trustworthiness and fairness, seeking association with positive values [4]. Human values vary both individually and culturally [8], underscoring the need for multinational corporations to adapt their software engineering process accordingly.

In the broader software engineering community, the influence of human values has been discussed lately [13, 17, 18]. However, a concept of integrating human values into the requirements engineering process is currently missing. In this paper, we provide an approach to systematically integrate human values into the requirements engineering process as explicit and traceable artifacts. The approach creates a traceable chain of high-level goals to human values, which are analyzed, prioritized, and connected to personas. To bridge the gap from requirements engineering to design and implementation, we introduce the concept of "value tactics", which map values to functional and non-functional requirements. Overall, our approach ensures traceability from high-level system goals to system requirements, with a focus on human values as a first-order concern.

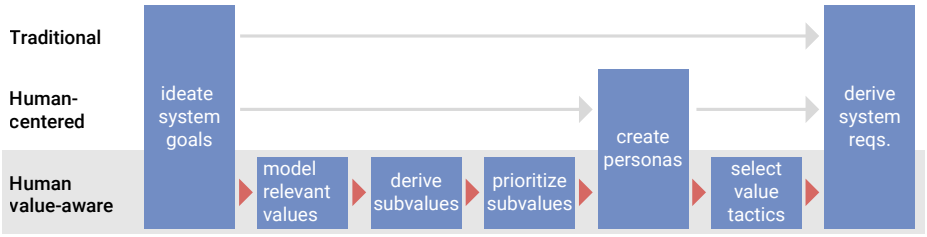

Fig. 1. Variants of deriving system requirements from goals

2 An Approach for Integrating Human Values into Requirements Analysis

Fig. 1 shows an overview of the initial process steps in (a) a traditional, (b) a human-centered, and (c) our new human value-aware approach. Each variant starts with an idea of the system goals and leads to deriving system requirements. Human-centered development introduces concepts such as personas to consider human stakeholders. Human value-aware requirements engineering adds explicit consideration of human values, such as resilience, fairness, or privacy. It involves (c) the explicit modeling of human values, a new technique for deriving personas, and—in particular—selecting value tactics to narrow the gap between identified human values and respective requirements. The steps of deriving and prioritizing subvalues are crucial but rely on existing practices.

Running Example: A Chat Application

We consider a fictitious chat application, *ChatApp*. It shall support communication between users to message friends, schedule meetings, or describe health symptoms and get help. In some situations, the communication shall be supported by AI chatbots. The company aims to develop ChatApp in a way that is in line with its end users' values. This example scenario will be used to illustrate our approach. If human values are considered relevant, this will occur soon after defining the first vision of the app. Therefore, we assume that stakeholders have agreed on (i) the high-level goal or purpose of the system and (ii) rough demographics of the humans that are to interact with the system. In our running example, the involved stakeholders know that a ChatApp shall be developed and used by adult humans for private communication (i.e., the vision). Relevant human values are then elicited and modeled. For example, design thinking and value-sensitive design techniques have proven useful for this step [3, 14]. While the model of Schwartz et al. [10] is very well-known in both psychology and Software Engineering, other value models can be used in that step as well.

Derive and prioritize subvalues. To elicit the associated subvalues, stakeholders are interviewed or asked in workshops. The process starts from the overall value model (e.g., Schwartz [10]). Stakeholders are asked to identify relevant subvalues. Conflicting subvalues are common in practice [11], and it is important to

be aware of such conflicts. Following our example, benevolence could mean different things to different stakeholders, depending on their specific individual and social experiences. Involved stakeholders can then rank the identified subvalues with respect to the relevance of a value for the end users. Several stakeholders participate in those discussions, in order to consolidate the identified values and subvalues in personas. Personas are fictional but realistic descriptions of users that are widely used in human-computer interaction and requirements engineering. We propose adding value-related information and ensuring that each persona is plausible in itself, i.e., does not have severe conflicts. If certain identified subvalues are in conflict, they are assigned to different personas. It is important to cover all selected subvalues by the resulting set of personas. Prioritization among the personas may be needed: if personas have conflicting sets of values, they are negotiated and prioritized by the participating stakeholders.

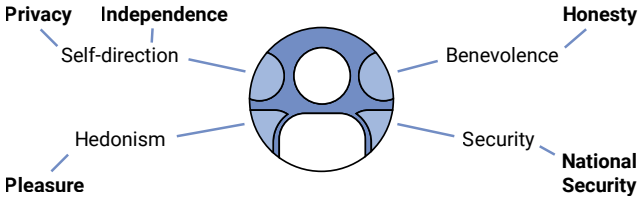

Fig. 2. Selected values and subvalues for ChatApp, based on Schwartz et al. [10]

Fig. 2 shows an overview of the selected values for ChatApp. Within the Schwartz model, they are the basis for deriving subvalues in the subsequent step. Values on opposing positions of Fig. 2 tend to conflict with each other. For example, self-direction is a potentially relevant value, indicating the freedom to choose one's own goals and actions.

Select value tactics After having developed a catalog of personas and subvalues, the stakeholders aim to ensure that values can be addressed in subsequent requirements analysis, system design, and the implementation of the system. Therefore, values need to be linked to functional or non-functional system requirements and design decisions that make them more actionable and concrete. We call these links *value tactics*, similarly to architectural tactics, that address specific quality attributes and suggest mechanisms to consider them in architectural design [6]. For instance, the quality attribute *reliability* can be addressed using the architectural tactics "*heartbeat*" and "*ping/echo*". Those tactics can be used to check whether a device is still running and responsive. By selecting architectural tactics in the architecture design, reliability can be improved. Instead of focusing on quality attributes, value tactics focus on values or subvalues and suggest how they can be addressed by more concrete design decisions.

Value tactics can be functional or non-functional design decisions. They are not limited to one specific system, but are generally valid to support a certain value. For instance, to ensure that the value "*honesty*" is represented, require-

ments engineers might choose to "*anonymize transactions*", which is a cybersecurity design decision, ensuring that people can share their opinions without fearing repercussions. To ensure that the value "*independence*" is represented, requirements engineers might choose to design the system in a way that does not dictate a strict workflow, but allows users to work on tasks in a customized way (e.g., during the installation or configuration process). That decision is a decision that impacts a function of the system.

Tab. 1 provides examples of values, subvalues, and value tactics for the ChatApp. Note that the value tactic is only listed with a concise name in the table. In practice, we envision that each tactic comes with a short description that highlights how the tactic works, what influence it has on different values and subvalues, and what needs to be considered when applying it in practice.

Table 1. Examples of values, subvalues, and value tactics

Value	Subvalue	Value Tactic
Benevolence	Honesty	Anonymize transactions
Benevolence	Honesty	Provide explicit information about the use of AI
Benevolence	Honesty	Describe the origin of data
Self-Direction	Privacy	Keep user information private by default
Self-Direction	Privacy	Disclose what personal data is stored
Self-Direction	Independence	Allow users to determine the workflow

Derive system requirements This step focuses on eliciting detailed requirements on the system design based on the value tactics. In an iterative process, requirements derived from the selected value tactics are refined until detailed design requirements have been identified. This refinement is based on traditional requirements engineering techniques [7], however, enriched by the outcomes of the previous steps—values, personas, and value tactics. In many cases, given that requirements engineering and design often go hand in hand, design alternatives are considered, and design decisions are made at this point. For instance, for the value tactic "anonymize transactions", concrete privacy-preserving methods are chosen in this step. For each refinement step, traceability is ensured by making input and output relations explicit. The identified detailed system design requirements will form the basis for system analysis and development.

3 Research Agenda

The goal of integrating human needs into software development has always been a driver in software engineering. For many years now, awareness of human values has grown in the academic community. Spreading awareness of values in the public and private sectors is key to creating systems that better meet stakeholders' needs. It requires transferring knowledge from academia to practice,

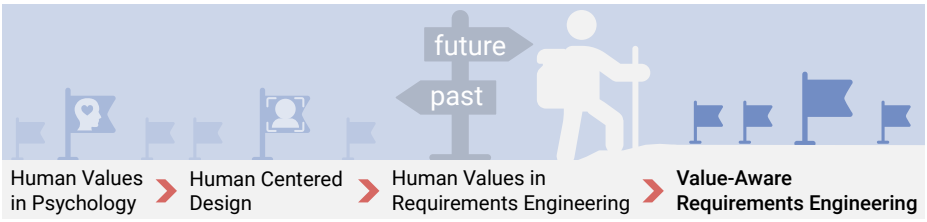

Fig. 3. From researching human values, via user-centricity, to our proposed approach for value-aware requirements engineering

e.g., supported by science communication. Our work proposes an approach that explicitly considers human values in a traceable way that does not stop at a high level, but breaks them down with the use of value tactics. Our approach highlights gaps that open up new areas for future research.

Fig. 3 depicts the path that research has traveled to date and indicates our proposed research roadmap for the future. In the following, we describe what is needed to develop mature contributions for value-aware software engineering:

RQ 1 **Communication:** What specific communication methods (e.g., the selection of a value framework) are most effective for engaging stakeholders in discussions about complex human values?

RQ 2 **Prioritization:** How can we develop a prioritization framework that helps stakeholders weigh different human values (e.g., national security vs. privacy) in the context of requirements engineering?

RQ 3 **Personas:** What aspects beyond traditional demographics (e.g., age, gender) should be considered when creating human value personas that accurately represent stakeholder concerns?

RQ 4 **Value tactics:** How can we create a reusable catalog of tactics that address different human values in requirements engineering?

RQ 5 **Industry adoption:** What practical tools and training programs can be developed to empower stakeholders to adopt value-aware requirements engineering processes in practice?

RQ 6 **Impact:** What impact does value-aware requirements engineering have on the software lifecycle, e.g., testing, maintenance, and operations?

RQ1 is concerned with finding an appropriate language to communicate human values. Most related work relies on the Schwartz [10] model. While it has been validated and used widely, it seems worthwhile to analyze if it is the best model in a system's context. A software engineering-focused model could be beneficial, providing crisp and context-dependent descriptions of values to ensure that people understand them consistently. For instance, should human values be described using natural language or formalized to avoid ambiguity?

The prioritization of human values is in itself a challenge to be tackled (RQ2). Deciding which values are relevant to include and which values need more focus during development requires a profound understanding of the human values

themselves. To be able to argue which values are more important and how values interact with each other, we need some guidance on understanding the meaning of specific values and how they translate to practical requirements.

Human value personas need to be further developed (RQ3). We only roughly proposed using the concept of personas to represent human values, where they emerged from, and how they are bundled. Future work should investigate how human value personas can best be represented and how to deal with conflicts in values between personas. Such personas may be developed and augmented using Large Language Models also to enable a conversation between developers and personas [9]. One possibility could be to develop adaptive functionality so that systems behave differently depending on the personas they interact with.

RQ4 is concerned with the development of reusable catalogs of value tactics. We need to understand how detailed and rigorous the catalogs should be. Should there be a standardized meta-catalog that includes all relevant human values from which stakeholders can select the ones that are relevant to them? How can stakeholders manage and maintain the catalogs as they grow in knowledge and dependencies? How can and should values be evaluated?

RQ5 addresses how practitioners could be supported when transitioning to value-aware requirements engineering. We are confident that values will play an even greater role in the future, but this does not automatically mean that it will be easier to translate them into specific system requirements. Therefore, RQ5 aims to identify hurdles and ultimately offer practitioners an adaptable process.

Finally, RQ6 is concerned with the empirical evaluation of how value-aware requirements engineering affects the development and use of software products. Concretely, we envision an investigation of the hypothesized positive effects in both the short and the long term.

4 Conclusion

The interaction between humans and software-intensive systems is significantly influenced by values such as honesty, security, and fairness. Despite the growing recognition of human values in software engineering, there is a need for concrete approaches that can be integrated into software processes.

We introduced our vision towards *Value-Aware Requirements Engineering*, which builds on existing human-oriented design methodologies. Our approach facilitates a systematic sequence of steps to identify, prioritize, and validate human values and subvalues, and introduces value tactics to bridge the gap from values to design decisions.

Acknowledgments This work is an outcome of the GI-Dagstuhl-Seminar 23473 "Human Factors in Model-driven Engineering". The authors are grateful to the Leibniz Center for Informatics and the seminar's organizers for the opportunity to collaborate and produce these results. This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP)

funded by the Knut and Alice Wallenberg Foundation and by the KASTEL Security Research Labs (46.23.01).

References

1. Detweiler, C.A., Harbers, M.: Value stories: Putting human values into requirements engineering. In: REFSQ Workshops (2014)
2. Farooqui, T., Rana, T., Jafari, F.: Impact of human-centered design process (HCDP) on software development process. In: C-CODE. pp. 110–114 (2019)
3. Friedman, B.: Value-sensitive design. Interactions **3**(6), 16–23 (1996)
4. Jin, K.G., Drozdenko, R.G.: Relationships among perceived organizational core values, corporate social responsibility, ethics, and organizational performance outcomes: An empirical study of information technology professionals. Journal of Business Ethics **92**, 341–359 (2010)
5. van Lamsweerde, A.: Requirements engineering in the year 00: a research perspective. In: ICSE. p. 5–19 (2000)
6. Márquez, G., Astudillo, H., Kazman, R.: Architectural tactics in software architecture: A systematic mapping study. Journal of Systems and Software **197**, 111558 (2023)
7. Pohl, K.: Requirements Engineering: Fundamentals, Principles, and Techniques. Springer Publishing Company, Incorporated, 1st edn. (2010)
8. Sagiv, L.: Personal values, national culture and organizations: Insights applying the schwartz value framework. The Handbook of Organizational Culture and Climate **2**(4), 515–537 (2011)
9. Salminen, J., Liu, C., Pian, W., Chi, J., Häyhänen, E., Jansen, B.J.: Deus ex machina and personas from large language models: Investigating the composition of AI-generated persona descriptions. In: CHI. pp. 1–20 (2024)
10. Schwartz, S.H.: Universals in the content and structure of values: Theoretical advances and empirical tests in 20 countries. In: Zanna, M.P. (ed.) Advances in Experimental Social Psychology, vol. 25, pp. 1–65. Academic Press (1992)
11. Schwartz, S.H.: An overview of the schwartz theory of basic values. Online Readings in Psychology and Culture **2**, 11 (2012)
12. Schwartz, S.H., Bilsky, W.: Toward a universal psychological structure of human values. Journal of Personality and Social Psychology **53**(3) (1987)
13. Shahin, M., Hussain, W., Nurwidyantoro, A., Perera, H., Shams, R., Grundy, J., Whittle, J.: Operationalizing human values in software engineering: A survey. IEEE Access **10**, 75269–75295 (2022)
14. Sjøkvist, N.M., Kjørstad, M.: Eliciting human values by applying design thinking techniques in systems engineering. In: INCOSE International Symposium. vol. 29, pp. 478–499. Wiley Online Library (2019)
15. Sutcliffe, A., Sawyer, P.: Requirements elicitation: Towards the unknown unknowns. In: RE. pp. 92–104 (2013)
16. Thew, S., Sutcliffe, A.: Value-based requirements engineering: Method and experience. Requirements Engineering **23**(4), 443–464 (nov 2018)
17. Whittle, J., Ferrario, M.A., Simm, W., Hussain, W.: A case for human values in software engineering. IEEE Software **38**(1), 106–113 (2021)
18. Winter, E., Forshaw, S., Ferrario, M.A.: Measuring human values in software engineering. In: ESEM (2018)
19. Zowghi, D., Coulin, C.: Requirements Elicitation: A Survey of Techniques, Approaches, and Tools, pp. 19–46. Springer Berlin Heidelberg (2005)