Design Requirements for Value-Critical Systems

Marc Herrmann*, Elisa Schmid, Michael Mircea and Kurt Schneider

Leibniz University Hannover, Software Engineering Group, Hannover, Germany

Abstract

Software systems increasingly need to account for stakeholder values, yet existing approaches provide limited support for integrating them into system design. We analyze these limitations for a class of systems we define as Value-Critical Systems. We find that oversight and governance approaches rely on high-level principles that are difficult to translate into concrete system behavior and often provide only limited or illusory control. Software quality models capture system properties but fail to represent the underlying stakeholder values they are intended to support. Value-oriented design methodologies, while effective at eliciting values, lack mechanisms to operationalize them, reason about trade-offs, and integrate them throughout the system lifecycle.

To address these limitations, we derive five design requirements for the operationalization of stakeholder values in Value-Critical Systems. These requirements support making values explicit, comparable, and measurable, while facilitating the identification and negotiation of trade-offs and their continuous revision over time.

Keywords

Requirements, Stakeholder Values, Meta-Design, Value Trade-offs, Responsible AI, Socio-Technical Systems

1. Introduction

Software systems increasingly shape human life, influencing how we work, communicate, learn, and make decisions. While traditional quality attributes such as performance, reliability, or security remain important, they are not sufficient to capture the full spectrum of stakeholder concerns. In particular, values [1] play a central role in shaping stakeholder expectations, trust, and acceptance of systems.

This is especially evident in systems that process personal data (e.g., health tracking), mediate decisions (e.g., credit scoring), or influence user behavior (e.g., social media). We refer to such systems as Value-Critical Systems (VCSs).

> **Value-Critical System (VCS).** A system in which the alignment with stakeholder values is central to its design, operation, and acceptance.

Misalignment between a VCS's design and stakeholder values can result in outcomes that harm stakeholders and erode trust [2]. This is amplified by the fact that values operate across multiple levels, which can be abstracted into cultural, organizational, and individual contexts (cf. Table 1). Individual values are shaped by organizational norms, which in turn are embedded in broader cultural contexts.

Table 1: Abstraction levels of values, with example frameworks/models and their relevance to software design.

Abstraction Level	Framework/Model	Relevant Design Aspects
Cultural	Hofstede [3]	Societal norms and expectations across regions.
Organizational	Cameron & Quinn [4]	Shared priorities, and governance practices.
Individual	Schwartz [1]	Personal rights, preferences, and ethics.

Proceedings of the 10th International Workshop on Cultures of Participation in the Digital Age (CoPDA 2026): Exploring the Relationship between EUD, AI-Assisted Development, and Meta-Design, June 2026, Venice, Italy.

*Corresponding author.

marc.herrmann@inf.uni-hannover.de (M. Herrmann); elisa.schmid@inf.uni-hannover.de (E. Schmid); michael.mircea@inf.uni-hannover.de (M. Mircea); kurt.schneider@inf.uni-hannover.de (K. Schneider)

0000-0002-3951-3300 (M. Herrmann); 0009-0006-2498-9986 (E. Schmid); 0009-0007-6783-4981 (M. Mircea); 0000-0002-7456-8323 (K. Schneider)

This introduces key challenges for VCS design: values are context-sensitive (e.g., trust in banking vs. trust in autonomous driving), may conflict across levels (e.g., individual privacy vs. national security), evolve over time (e.g., increasing emphasis on sustainability), and be interpreted differently by stakeholders (e.g., varying definitions of fairness). Therefore, addressing stakeholder values in system design inherently exhibits the defining properties of wicked problems [5]. Consequently, VCSs cannot be designed through one-time optimization but require continuous negotiation and adaptation.

Recent high-profile cases highlight the consequences of neglecting stakeholder values in system design. For example, the *Volkswagen* emissions scandal [6] revealed how software systems can be deliberately engineered to obscure environmental impact, undermining values such as sustainability and transparency. Similarly, the *Robinhood* trading platform prevented retail investors from accessing financial markets [7], raising concerns about fairness and trust. While trade-offs between business objectives and stakeholder values are often intentional, long-term consequences are difficult to anticipate.

However, existing approaches like quality models [8] provide limited guidance on how to systematically represent, reason about, and evaluate heterogeneous and often conflicting stakeholder values during system design. This gap underscores the need for design approaches that operationalize values alongside traditional quality attributes like safety and security.

In this paper, we analyze the limitations of existing approaches and derive five design requirements to guide the systematic operationalization of stakeholder values. These requirements include (i) capturing the diversity of stakeholder values across abstraction levels, (ii) representing values as explicit and traceable elements, (iii) supporting the negotiation of value trade-offs, (iv) evaluating systems based on how values are realized in behavior, and (v) enabling continuous adaptation to reflect evolving stakeholder values. Together, these requirements provide a foundation for systematically integrating stakeholder values into the design and evolution of VCSs.

2. Limitations of Existing Approaches

Existing approaches that are not necessarily designed to address stakeholder values explicitly but relate to similar concerns can be grouped into oversight and governance approaches, software quality models, and value-oriented design approaches, each with their own limitations.

2.1. Oversight and Governance Approaches

Oversight and governance approaches aim to ensure alignment with stakeholder values by introducing human control or high-level normative guidance into system design and operation but often struggle to translate these values into concrete system behavior.

Responsible AI and Ethics Frameworks aim to promote ethical principles derived from underlying values in system design. However, these approaches often exhibit a gap between those principles and their practical implementation in real-world systems. Their translation into practice is hindered by factors such as the complexity of AI impacts, unclear distributions of responsibility, disciplinary divides, and organizational constraints [9].

In addition, the fundamental principles often converge on a relatively small and standardized set of values, such as fairness, accountability, and transparency. While this may not address all stakeholder concerns, even within these values, their interpretation is not uniform across different stakeholders. For example, different notions of fairness can be inherently incompatible, making the pursuit of one fairness criterion contingent on trade-offs with others [10].

Human-In-The-Loop (HITL), and related paradigms such as **Human-On-The-Loop (HOTL)**, which emphasizes human involvement either through direct interaction or supervisory control at runtime, are often proposed to support values such as human autonomy, trust, and informed consent.

However, in end-user settings, this assumption breaks down, as users typically lack the information, authority, and context required to meaningfully oversee algorithmic decisions. While HITL enables users to directly intervene (e.g., by correcting or overriding outputs), HOTL assumes a supervisory role in which humans monitor system behavior and intervene only when necessary.

In practice, both of these paradigms face similar limitations. In many interfaces (particularly generative AI systems), human involvement is reduced to selecting or modifying system-generated suggestions. In such cases, the system dynamically controls which options are presented, thereby pre-structuring the decision space and potentially excluding alternatives from consideration. As a result, even in supervisory settings, human control may remain limited, providing only the appearance of oversight while the system retains substantial influence over the effective decision process [11].

2.2. Software Quality Models

Traditional software quality attributes (also referred to as non-functional or quality requirements [12]) described in the *ISO/IEC 25010 — Product Quality Model* [8], such as safety and security, are not values themselves but a means through which stakeholder values can be supported or realized.

For example, security supports the value privacy, and safety supports the value health. On the other hand, values motivate the selection and prioritization of quality attributes. Quality attributes define how a system should behave, but they do not always make explicit why these properties are important, i.e., which stakeholder values they are intended to support (cf. Figure 1).

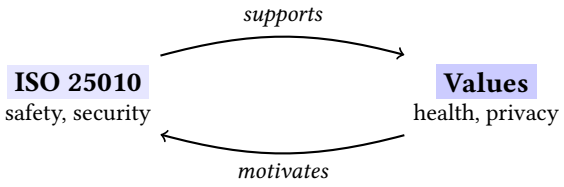

Figure 1: Relationship of ISO 25010 software quality attributes and stakeholder values.

While systems may satisfy specified quality attributes, this does not guarantee that the underlying stakeholder values are adequately addressed [13]. For example, a system may improve usability through simplified interfaces intended to support accessibility, yet still exclude certain users if this simplification removes information that users with specific needs depend on [14]. In such cases, quality attributes capture specific system properties but fail to represent the broader value context and potential trade-offs.

2.3. Design Approaches

Design approaches aim to incorporate stakeholder values into system development through participatory and value-oriented methodologies, but often struggle to capture their full complexity in practice.

Inclusive Design aims to ensure that systems are accessible and non-discriminatory across diverse user groups. However, these approaches focus on optimizing specific outcomes, such as fairness along a single dimension. This narrow focus neglects the broader value landscape and can introduce new trade-offs or unintended forms of discrimination. For example, recent studies on large language models reveal that attempts to mitigate gender bias can lead to asymmetric outcomes. In one set of experiments, stereotypically masculine statements were more frequently attributed to female authors than vice versa, suggesting an overcorrection effect [15].

Value-Sensitive Design (VSD) provides a well-established framework for identifying and incorporating the values of diverse stakeholder groups through conceptual, empirical, and technical investigations [16]. However, despite its strengths in value elicitation, VSD exhibits several limitations in the context of VCSs. First, due to the context-sensitive nature of values, VSD resists strict formalization. This leads to a practical limitation: it lacks concrete mechanisms to operationalize values as system properties, leaving them largely qualitative.

Second, while methods such as *Value Dams and Flows* [17] are proposed to handle value tensions, they rely on aggregating stakeholder responses and do not account for power asymmetries or ethically relevant differences between stakeholder groups. As a result, they offer limited support to reason about trade-offs for competing values across stakeholder groups.

Finally, VSD is primarily focused on design-time activities and offers limited support for integrating values throughout the lifecycle of VCSs. In particular, it does not provide mechanisms for stakeholders to continuously redefine and negotiate values as their importance evolves after deployment.

3. Design Requirements

To address the limitations of existing approaches, we derive a set of design requirements (DR1–DR5) for treating values as explicit, operational, and evolving elements when designing VCSs.

DR1: Systems should capture the diversity of stakeholder values across abstraction levels.

Existing approaches often rely on a small and standardized set of values, which fails to reflect the diversity of stakeholder perspectives.

Values manifest across the abstraction levels outlined in Section 1, encompassing cultural values, organizational goals, and individual stakeholder preferences. For example, in high power distance cultures, stakeholder values are often filtered through authority figures when eliciting requirements, so the needs of lower-level users remain underrepresented or unspoken [18]. On the other hand, eliciting values for each individual stakeholder is often infeasible. To address this, stakeholders can be abstracted into groups that share similar values. However, determining an appropriate level of granularity is non-trivial: groups can always be further subdivided, while overly coarse groupings risk overlooking important differences (cf. Figure 2).

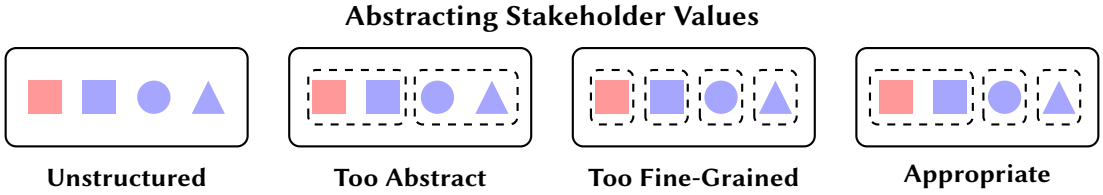

Figure 2: Schematic variants of abstracting stakeholder values based on different perspectives.

Capturing value diversity therefore requires balancing expressiveness and tractability by defining stakeholder groups based on systematic, revisable value profiles that reflect meaningful yet manageable differences rather than fixed or ad hoc categorizations.

DR2: Systems should represent stakeholder values as explicit and traceable elements.

Values are often treated implicitly, leaving them underspecified and difficult to trace, interpret, or operationalize within the system.

To address this, values must be represented explicitly as first-class elements within the design process [2]. These value representations must be semantically precise. High-level concepts such as fairness require explicit, context-dependent interpretation. Otherwise, they remain ambiguous and cannot guide system behavior. While VSD emphasizes stakeholder involvement in interpreting values [16], this interpretation is typically mediated by system designers. Following participatory design approaches [19], systems should instead enable stakeholders to actively define value semantics in context rather than relying on predefined interpretations.

Further, value representations must be traceable across the system lifecycle [2]. It should be possible to link abstract values to design decisions, system components, and observable outcomes, providing a basis for revising value interpretations as contexts evolve.

DR3: System designs should support the negotiation of value trade-offs.

Stakeholder values often conflict (e.g., individual privacy vs. national security), yet existing approaches lack mechanisms to systematically reason about such trade-offs or make prioritization decisions explicit.

Value representations (cf. DR2) must therefore be comparable, enabling the identification of tensions both within and across stakeholder groups and abstraction levels.

Trade-offs must be explored and negotiated collaboratively with stakeholders through structured processes such as workshops [20], co-design sessions, or scenario-based discussions, where alternative

system behaviors and their value implications are made explicit. Engaging stakeholders in comparing these alternatives helps surface assumptions, reveal conflicts, and shape how trade-offs are defined and resolved. Resulting decisions must remain transparent and justifiable by documenting prioritized values, compromises, and their relation to stakeholder perspectives.

DR4: Systems should be evaluated based on how stakeholder values are realized in behavior.

Beyond traceability (cf. DR2), assessing whether values are effectively implemented requires metrics that capture how values manifest in system behavior, such as statistical measures of fairness. However, measurement must account for multiple, potentially conflicting interpretations and make explicit which interpretation is being operationalized. Causal inference can identify true causes of outcomes, while robustness techniques (e.g., perturbations) can be used to assess outcome sensitivity to design choices.

In practice, this should be complemented by stakeholder-centered evaluation, engaging affected users through surveys or interviews to capture how values are perceived and experienced in context. Stakeholders themselves are best suited to validate whether their values are addressed [21].

DR5: Systems should be continuously adapted to reflect evolving stakeholder values.

Values cannot be fixed at design time alone. Instead, value semantics must be defined and continuously refined by stakeholders in context. This requires mechanisms that support ongoing negotiation, revision, and reconfiguration of value representations and their operationalizations after deployment.

This perspective aligns with the *Seeding, Evolutionary Growth, and Reseeding* model [22], in which systems are initially seeded through participatory design, evolve through use as new requirements and knowledge emerge, and are periodically restructured to integrate and formalize accumulated changes. Applying this concept to VCS design implies that value definitions, prioritizations, and trade-offs should be continuously shaped, revised, and realigned through ongoing stakeholder interaction.

4. Conclusion

This paper identified limitations of existing approaches to addressing stakeholder values in VCSs and derived a set of design requirements for treating values as explicit, operational, and evolving elements of system design. By emphasizing value representation, trade-off reasoning, evaluation, and adaptation, the requirements shift from static principles toward dynamic, context-sensitive value operationalization, while defining necessary properties for value-aware systems at an intermediate level of abstraction.

These requirements highlight the need to move beyond implicit and design-time assumptions about values, enabling stakeholders to actively shape and revise value interpretations over time. Realizing this vision requires new forms of tool support, integration into development processes, and mechanisms for stakeholder participation throughout the lifecycle of VCSs.

Future work should investigate how these requirements can be instantiated in concrete VCSs, how value trade-offs can be systematically managed in practice, and how continuous value adaptation can be supported in real-world deployment contexts.

Declaration on Generative AI

During the preparation of this work, the author(s) used ChatGPT, LanguageTool in order to: Grammar and spelling check, Paraphrase and reword. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the publication's content.

References

[1] S. H. Schwartz, Universals in the content and structure of values: Theoretical advances and empirical tests in 20 countries, in: M. P. Zanna (Ed.), Advances in Experimental Social Psychology, volume 25, Academic Press, 1992, pp. 1–65. doi:10.1016/S0065-2601(08)60281-6.

[2] R. Wohlrab, et al., Supporting value-aware software engineering through traceability and value tactics, in: D. Pfahl, J. Gonzalez Huerta, J. Klünder, H. Anwar (Eds.), PROFES, Springer Nature Switzerland, Cham, 2025, pp. 368–376. doi:10.1007/978-3-031-78386-9_27.

[3] G. Hofstede, National cultures in four dimensions: A research-based theory of cultural differences among nations, International Studies of Management & Organization 13 (1983) 46–74. doi:10.1080/00208825.1983.11656358.

[4] M. A. Maher, Diagnosing and changing organizational culture: Based on the competing values framework, Journal of Organizational Change Management 13 (2000) 300–303. doi:10.1108/jocm.2000.13.3.300.1.

[5] H. W. J. Rittel, M. M. Webber, Dilemmas in a general theory of planning, Policy Sciences 4 (1973) 155–169. doi:10.1007/BF01405730.

[6] J. C. Jung, E. Sharon, The volkswagen emissions scandal and its aftermath, Global Business and Organizational Excellence 38 (2019) 6–15. doi:10.1002/joe.21930.

[7] A. Agnihotri, S. Bhattacharya, Robinhood: Empowering or controlling retail investor behavior, in: Business Cases Originals, SAGE Publications, London, 2023. doi:10.4135/9781529611694.

[8] ISO/IEC, Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Product quality model, 2011. URL: https://iso.org/standard/78176.html.

[9] D. Schiff, B. Rakova, A. Ayesh, A. Fanti, M. Lennon, Principles to practices for responsible ai: Closing the gap, 2020. arXiv:2006.04707.

[10] E. Ferrara, Fairness and bias in artificial intelligence: A brief survey of sources, impacts, and mitigation strategies, Sci 6 (2024). doi:10.3390/sci6010003.

[11] S. Tschiatschek, E. Stamboliev, T. Schmude, M. Coeckelbergh, L. Koesten, Challenging the human-in-the-loop in algorithmic decision-making, 2024. arXiv:2405.10706.

[12] M. Glinz, On non-functional requirements, in: 15th IEEE International Requirements Engineering Conference (RE 2007), 2007, pp. 21–26. doi:10.1109/RE.2007.45.

[13] L. Bass, P. Clements, R. Kazman, Software Architecture in Practice, 3rd ed., Addison-Wesley, 2012.

[14] A. F. Newell, P. Gregor, "User sensitive inclusive design"— in search of a new paradigm, CUU '00, ACM, NY, USA, 2000, p. 39–44. doi:10.1145/355460.355470.

[15] R. A. Fulgu, V. Capraro, Surprising gender biases in gpt, Computers in Human Behavior Reports 16 (2024) 100533. doi:10.1016/j.chbr.2024.100533.

[16] B. Friedman, Value-sensitive design, Interactions 3 (1996) 16–23. doi:10.1145/242485.242493.

[17] J. K. Miller, B. Friedman, G. Jancke, B. Gill, Value tensions in design: the value sensitive design, development, and appropriation of a corporation's groupware system, in: Proceedings of the 2007 ACM International Conference on Supporting Group Work, GROUP '07, ACM, NY, USA, 2007, p. 281–290. doi:10.1145/1316624.1316668.

[18] C. S. Muzammel, M. Spichkova, J. Harland, Cultural influence on re activities: An extended analysis of state of the art, MobileHCI '24, ACM, NY, USA, 2024, pp. 1–8. doi:10.1145/3640471.3680236.

[19] D. Schuler, A. Namioka, Participatory design: Principles and practices, CRC press, 1993.

[20] M. Herrmann, M. Mircea, K. Schneider, Aligning AI systems with human values, in: International Requirements Engineering Conference Workshops (REW), IEEE Computer Society, Los Alamitos, CA, USA, 2025, pp. 442–444. doi:10.1109/REW66121.2025.00065.

[21] M. Mircea, E. Gevrek, E. Schmid, K. Schneider, Supporting stakeholder requirements expression with llm revisions: An empirical evaluation, in: R. Guizzardi, J. Araújo (Eds.), REFSQ, Springer Nature Switzerland, Cham, 2026, pp. 303–319. doi:10.1007/978-3-032-21423-2_21.

[22] G. Fischer, R. McCall, J. Ostwald, B. Reeves, F. Shipman, Seeding, evolutionary growth and reseeding: supporting the incremental development of design environments, CHI '94, ACM, NY, USA, 1994, p. 292–298. doi:10.1145/191666.191770.