

Preface

The journey of this book started nearly 10 years ago when I prepared my first textbook for students on Ethical IT Innovation in 2014. At the time few people realized how urgently we need to develop a more thoughtful and ethically sensitive approach to IT system design. Of course, many critical minds at universities and tech-focused NGOs had realized that IT systems would need to be more sensitive to privacy and security. The EU's General Data Protection Regulation (GDPR) was under way. But beyond the GDPR effort hardly any initiatives were able to successfully call for a more thorough protection of human values. This has changed dramatically since 2014. Since Edward Snowden's revelations on the scale of global digital surveillance (late 2013), Cambridge Analytica's exposure for manipulating the Trump election (2018) and the influential documentary film *The Social Dilemma* (2020), many noteworthy activists, whistleblowers, individual tech pioneers, corporate leaders and politicians have been critical of the sheer scale at which our IT innovations threaten democracy, undermine political and economic control, change the nature of warfare, and influence human lifestyle, personality and health. A political debate has emerged around untransparent, unaccountable and uncontrollable AI systems, which might compromise the values that we as a society hold dear. Furthermore, the looming specter of climate change has caused financial investors, insurance companies and other economic forces to recognize the need for a more environmentally friendly and social governance of IT operations.

But how are IT companies to build systems that live up to this new call for more value respect? In 2014, I realized that it is not easy to build IT systems in a value-based manner. At the time I had already been teaching IT system design and analysis at the University for 10 years. I had also been teaching about human-computer interaction, user-centric system design and user experience. But neither standardized system development lifecycle methods, nor more agile approaches to system design, nor HCI or UX coursework seemed to integrate any sensible methodological approach to address or even think about value issues. Standard textbooks on software engineering mostly ignore the social challenges of IT systems to this day. The only straw I could clutch at back then was the work that Batya Friedman as well as others had been doing on Value Sensitive Design (VSD) since the late 1990s. The case studies accumulated by the VSD community over the years as well as her conceptual work on values in system design inspired my own work on Value-Based Engineering as it is summarized in this book. In fact, I tried in my 2014 textbook on Ethical IT Innovation to come up with an "Ethical System Development Life Cycle," in which I combined the Value Sensitive Design approach with participatory design and risk management. But at this first attempt, I was not able to solve the core conundrum of value-based design; namely, how to go systematically from value principles to practice. Or in other words: how to practically, systematically and traceably derive concrete IT system features and requirements from lofty value principles such as love, dignity, freedom or fairness. It

took the intervening 10 years for me to find a reliable answer for this problem, which you now find published in this volume.

How did I find the answer and develop Value-Based Engineering since 2014? I certainly did not do it alone. Together with a workgroup of IEEE volunteers, I engaged in a five-year-long standardization project called “P7000.” I co-initiated, served as vice chair, and intellectually guided this project, which aimed to develop the world’s first model process for tackling ethical concerns in system design, between 2016 and 2021. Given the enormous amount of ambition at the heart of this goal, it should not come as a surprise that endless and fierce discussions emerged. I can now with certainty attest to the fact that discussions around ethics and values and how to best live up to them are among the most emotionally charged. They touch upon very personal human convictions and philosophies on how to see the world and do things. Workgroup members engaged, for example, in discussions about what the word “design” actually means, whether ethical systems should have “zero fault tolerance,” what “values” are and how they can end up in a system since they are actually invisible phenomena. They debated how companies’ pursuit of profit could at all be aligned with ethics, how top management could be involved more in a value mission, how work conditions for engineers could allow for more room to tackle ethical issues, how to bridge the gap between engineers and managers (“the two-domain problem”), how to respect human rights in a global standard if the whole world does not equally share in them, how to avoid ethical relativism, etc. I think that leading and completing the IEEE Std 7000TM project has been one of the most painful and difficult endeavors of my life. As in any truly democratic and bottom-up decision process, the challenge is how to ensure the highest quality standards while listening to and including everyone. Is this at all possible? IEEE Std 7000TM has been adopted by ISO/IEC JTC 1 and is also available as ISO/IEC/IEEE 24748-7000, *Systems and Software Engineering – Life Cycle Management – Part 7000: Standard Model Process for Addressing Ethical Concerns during System Design*. Personally, I am proud of the result. I believe that companies are well advised to use IEEE Std 7000TM – or parts of it. Their IT systems will be much better than what they are currently building.

Why will IT systems be much better with the Value-Based Engineering approach? I think an eye-opener for me was the case studies we conducted at my university, while developing the standard. In particular, we explored how the use of the three grand philosophical streams of utilitarianism, virtue ethics and duty ethics would sensitize IT innovation projects for the likely value harms inherent in their projects as well as for the potentially positive value propositions dormant in the technology. What struck me most was how one and the same technology can actually be put to bad as well as good use depending on how it is built. We can build an IT system for food delivery that treats bike couriers like isolated, surveilled and controlled pinballs in a flipper machine. Or we can design the very same technical system in such a way that bike couriers benefit from a free and sporty job in which they have control over their own time, find a sense of community and remain

private. We can build telemedicine platforms in such a way that they are no more than a cheap dial-in service – dehumanized, convenient and supported by a half-baked self-service AI. Or we can build such a platform as a cooperative online environment, helping remote patients and practitioners to benefit from global health knowledge expertise.

When values are explored with the Value-Based Engineering approach early on in an innovation project, what emerges is a whole new value story or likely value proposition reachable with the technology. The dormant value potential inherent in the technical capabilities becomes apparent – value potentials that are social and human-friendly and go beyond the narrow, profit-driven goals of efficiency, productivity or monetary gain pursued by the majority of IT innovation efforts these days. The case studies conducted in cooperation with my Ph.D. student Kathrin Bednar have opened my eyes to the fact that we could indeed create a better world with technology if we wished to. But will we do that? Will anyone care to use Value-Based Engineering, seeing that it takes more time and effort to develop a system with care? At least this book, together with IEEE Std 7000TM, gives tech companies a clear, transparent, reviewed and sound methodology on how to go about such an undertaking.

This book guides its users on how to use IEEE Std 7000TM and how to interpret many of the standard's terms. It provides project teams with practical forms to fill when traceably deriving system features and requirements from value principles. But it also encourages the philosophically interested reader to understand what values are. Unlike any other contemporary work on values in technology (that I know of), it contains what philosophers call an “ontology” – an ontology of the value construct that allows the users of this book to discern value dispositions in systems, from value qualities perceived by humans to those absolute values we find, for instance, in laws or human rights agreements (see Chapter 3). This ontology was derived from Max Scheler's *Material Value Ethics*, a philosophical masterpiece ignored and underestimated by many contemporary value ethicists. I am particularly proud of this value ontology and the corresponding value vocabulary, which I could then apply in large part to bolster the standardization effort. I believe that without a sound definition and understanding of the value phenomenon itself, the IEEE Std 7000TM effort would have remained unsatisfactory.

I am especially grateful to my husband Johannes Hoff, who as a professor of philosophy and theology accompanied and continuously challenged me in this endeavor to develop a proper value ontology for technology design – an endeavor that would ordinarily be pursued only by “hardcore” philosophers and hardly IS scholars like myself! I am also grateful to Lee Barford, a mathematician, theologian and quantum physics veteran in Silicon Valley who is co-responsible for much of the ethical details in IEEE Std 7000TM that are now also embedded in this book. I want to thank Lewis Gray, a consultant on software system development based in Washington, who taught me more about IT requirements engineering than I could

learn from any textbook in the world. Equally, I want to thank Ruth Lewis, an Australian IT consultant with whom I co-developed the EVR-construct (Ethical Value Requirements) key to Value-Based Engineering, and who opened my eyes to the interfaces between Value-Based Engineering and design thinking. I am indebted to Annette Reilly, Technical Editor of IEEE Std 7000; Ali Hessami, Chair of the P7000 Working Group (WG); and Gisele Waters and Alexander Novotny who patiently endured the IEEE Std 7000TM project for as long as it lasted, leading it, improving it and making it what it has become. Without them, this book would not exist either. Ruth Lewis, Lewis Gray, Ali Hessami and Alexander Novotny have also reviewed various sections of this book and gave me invaluable feedback. I want to thank my Ph.D. students Till Winkler and Kathrin Bednar. Both helped me grasp the complexity of value clusters. Till also participated in the P7000 project and was a wonderful peer, equally reviewing parts of this book. Ross Ludlam continuously helped to polish and improve the language as my editor. Finally, I am deeply indebted to Marie Therese Sekwenz, who illustrated this book for me. My idea to develop drawings for almost every page was inspired by old biblical scriptures like the Lutheran Psalter, which always had drawings accompanying the text. The drawings served the purpose of helping to better commit the content to memory. Indeed, it would be possible for a skilled Value-Based Engineering lecturer to teach the material of this book based on the figures alone. Without Marie Therese's talent, this wonderful visual format would not have been possible.

Last but not least, I want to thank the 154 experts who were at some point involved in the making of IEEE Std 7000TM, 34 of which were WG members. Seventy-seven experts balloted for the standard's publication in 2021 (93% acceptance rate). Without them and the generous support of the IEEE Standards Association (in particular, Konstantinos Karachalios) and IEEE Computer Society, neither the standard nor this book would exist.

At this point I must also make a disclaimer: In this book I cite extensively from IEEE Std 7000TM, but I want it to be noted that IEEE standards like IEEE Std 7000TM are the result of group work and consensus-building. Many small decisions are taken, and not all of these are shared unanimously. The interpretations and views expressed in this book solely represent my view on how IEEE Std 7000TM can be put into practice. I do not necessarily represent the position of either the IEEE Std 7000TM Working Group, IEEE or the IEEE Standards Association. Where I have felt that IEEE Std 7000TM does not recommend what I think should be done or has a suboptimal definition of terms, I have made this explicit here, and also give reasons for my own divergent view where required.