Sustainability Design and Software: The Karlskrona Manifesto

Christoph Becker
Faculty of Information
University of Toronto
Toronto, ON, Canada
christoph.becker@utoronto.ca

Ruzanna Chitchyan
Dept of Computer Science
University of Leicester
Leicester, UK
rc256@leicester.ac.uk

Leticia Duboc
Dept of Inf. & Computer Science
State Univ. of Rio de Janeiro
Rio de Janeiro, Brazil
leticia@ime.uerj.br

Steve Easterbrook
Dept of Computer Science
University of Toronto
Toronto, ON, Canada
sme@cs.utoronto.ca

Birgit Penzenstadler
Institute for Software Research
University of California, Irvine
Irvine, California, US
bpenzens@uci.edu

Norbert Seyff
Dept of Informatics
University of Zurich
Zurich, Switzerland
seyff@ifi.uzh.ch

Colin C. Venters
School of Computing & Engineering
University of Huddersfield
Huddersfield, UK
c.venters@hud.ac.uk

***Abstract*—Sustainability has emerged as a broad concern for society. Many engineering disciplines have been grappling with challenges in how we sustain technical, social and ecological systems. In the software engineering community, for example, maintainability has been a concern for a long time. But too often, these issues are treated in isolation from one another. Misperceptions among practitioners and research communities persist, rooted in a lack of coherent understanding of sustainability, and how it relates to software systems research and practice. This article presents a cross-disciplinary initiative to create a common ground and a point of reference for the global community of research and practice in software and sustainability, to be used for effectively communicating key issues, goals, values and principles of sustainability design for software-intensive systems. The centrepiece of this effort is the *Karlskrona Manifesto for Sustainability Design*, a vehicle for a much needed conversation about sustainability within and beyond the software community, and an articulation of the fundamental principles underpinning design choices that affect sustainability. We describe the motivation for developing this manifesto, including some considerations of the genre of the manifesto as well as the dynamics of its creation. We illustrate the collaborative reflective writing process and present the current edition of the manifesto itself. We assess immediate implications and applications of the articulated principles, compare these to current practice, and suggest future steps.**

I. Introduction

It is clear that society is facing major sustainability challenges that require long-term, joined-up thinking. How do we sustain our technical infrastructures, given how much we rely on them for everything from communication and navigation through to storing health records, identifying security threats, and keeping the lights on? How do we sustain prosperity in society, given the erosion of trust in our political institutions and a growing inequality in ownership of resources? And, above all, how do we sustain the planetary systems that support life on earth, in the face of accumulation of pollutants, species loss, and accelerating climate change?

The discipline of Software Engineering (SE) has a major role to play in sustainability, because of the extent to which software systems mediate so many aspects of our lives. However, software practice has a tendency to focus only on the immediate effects and tangible benefits of software products and platforms. SE research has, for the most part, focused on increasing the reliability, efficiency and cost-benefit relation of software products for their owners, through a focus on processes, methods, models and techniques to create, verify and validate software systems and keep them operational.

The lack of long-term thinking in software engineering research and practice has been critiqued throughout the history of the discipline. For example, software maintenance and evolution were raised as concerns even at the very first software engineering conference [1]. Since then, efforts to increase the maintainability of software products and facilitate their evolution have often focused on improving architecture, decreasing lifecycle costs and managing technical debt [2]. Neumann has criticized the lack of long-term thinking over security considerations in SE [3]. For our digital information assets, some now speak of a 'digital dark age' [4], where, having discarded analog media in preference for digital, we now find that many of these assets become unreadable, due, in part, to the rapid lifecycles of software technology.

While progress has been made on design for maintainability of software *per se*, considerations that extend beyond immediate software product qualities and user benefits are generally treated as secondary concerns, optional qualities to be addressed only after the system under design has been shown to be a success in terms of technical and/or marketing criteria. The larger impact of software artefacts on society and the natural environment is not routinely analyzed. But by trading off longer-term sustainability questions for shorter-term success criteria, we accumulate threats to sustainability. We argue that this trade-off itself is unnecessary. As Neumann

points out, "there is much to be gained from farsighted thinking that also enables short-term achievements." [3].

The benefits of longer-term thinking in software design are broad and far-reaching. In almost any domain, software is a key driver of continued automation and dematerialization [5]. Increasingly, our consumption of resources and access to information are shaped by the design choices embedded in our software systems, rather than the conscious choices we make as individuals. Hence, the design of software systems comes with a special set of responsibilities to society that are much broader than those described in existing codes of ethics for computing professionals. While increasing attention on the broader effects of software on society has helped develop our understanding of these issues, a common perspective on how to incorporate sustainability thinking into the design of software systems is missing.

This article describes a cross-disciplinary initiative to create a common ground and develop a focal point of reference for the global community of research and practice in software and sustainability. The process was initiated at a working session at the Third Int. Workshop on RE for Sustainable Systems (RE4SuSy), held at RE'14, Karlskrona, Sweden. The session was was included in the program to take up a proposal in one of the workshop contributions which suggested that "[a]n open manifesto for forward-thinking sustainable software design, drafted collaboratively in an open and sustainable process, could set a milestone and provide the necessary focal point for joint future efforts" [6]. The central result of our work is the Karlskrona manifesto [7], a document to be used for effectively communicating key issues, goals, values and principles of sustainability design.

The next section provides a basis for the discussion by reviewing the history of ideas in the area of sustainability in software. Section III traces the history of the manifesto as a genre and draws observations from a study of manifestos. It summarizes the lessons learned and the principles guiding the collaborative writing process that we have initiated. The current version of the manifesto is reproduced in Section IV. Section V discusses the implications of these principles on SE research and practice, and some remaining open questions.

II. Software systems and sustainability

The concept of sustainability is used by many different communities, often in ambiguous ways. Its Latin origin *sustinere* was used as both *endure* and as *uphold, furnish [something] with means of support*.[1] In modern English, sustainability refers to the 'capacity' of a system 'to endure' [8]. But these definitions merely raise further questions [9]. Tainter points out we need to ask: (i) Sustain what? (ii) For whom? (iii) How long? (iv) At what cost? [10]

In Software Engineering, implicit discussions about sustainability can be traced back as early as 1968, when software maintenance and evolution were brought up at the NATO SE conference [1]. Lehman proposed his laws of software evolution shortly thereafter [11], [12], pointing out that most real-world software is embedded in a social context, and that changes in the software and the world affect each other. His first law of software evolution encapsulated this: "Any program that ... reflects some external reality undergoes continual change or becomes progressively less useful" [11]. Over the following decades, the software evolution community made significant advances in the areas of software maintenance, program understanding, reverse engineering, reengineering, mining software repositories, software migration, and software process improvement. Each of these areas has provided insights on how to improve SE practice and how to improve the quality of the systems we build [13].

Despite these advances in understanding software maintainability as a technical concern, in practice it is interdependent with its organizational and business context. Thus, technical sustainability of a software system cannot be separated from social and financial sustainability of the organization that created it, a challenge already acknowledged at the first Software Maintenance Workshop in 1983 [14]. Durdik *et. al.* point out that "in many software development projects, sustainability is treated as an afterthought, as developers are driven by time-to-market pressure and are often not educated to apply sustainability-improving techniques" [15], and they call for better guidance for software practice.

This concern for software sustainability parallels a much broader concern about sustainability in human society. Here sustainability is often defined using some variant of "the ability of the current generation to meet its needs without compromising the needs of future generations" [16]. While it can be hard to say what would constitute sustainability at the societal level, it is often easier to recognize which systems are not sustainable. Any system that consistently consumes more value (e.g., money, energy, effort) than it produces cannot be sustained indefinitely if its environment, from which resources are drawn, is finite. By all accounts, human society has been in such a state since the 1970s, consistently consuming more ecological resources than the planet can produce [17].

In developing a manifesto, the question of how we define sustainability has proved challenging. Our intent has been to develop a broad set of principles that would motivate deeper thinking on sustainability and software, while avoiding terminological disputes. In this, we have struggled to find a balance between abstraction and precision. We collected a number of definitions of sustainability from the literature (see Table I) and conducted a straw poll of workshop participants. The results favoured fundamental definitions such as 'the capacity to endure' [18], but emphasized the importance of more specific expressions and showed considerable support for most of the entries in Table I.

Our approach has therefore been to select as simple a definition as possible ('the capacity to endure'), and focus instead on a conceptual framework for thinking about sustainability and a set of dimensions by which to approach it. Even this approach has proved difficult. Early feedback on drafts of the manifesto questioned whether it is even appropriate to discuss

[1] http://en.wiktionary.org/wiki/sustineo#Latin

Table I: Sustainability-related definitions

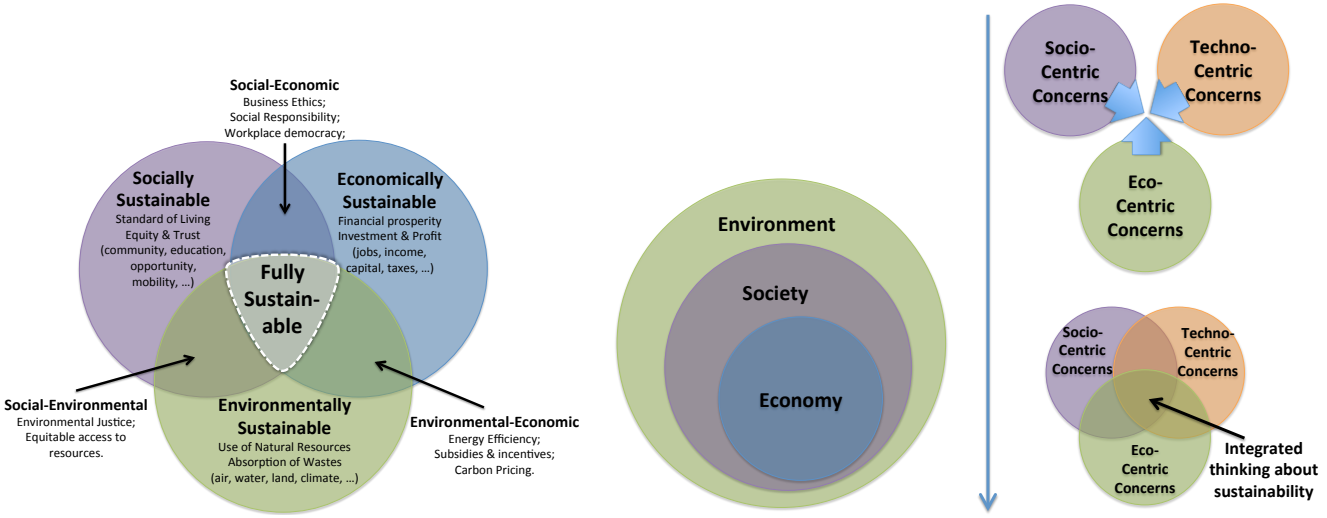
Source	Definition
SustainAbility [18]	Sustainability as 'capacity to endure'
Carlowitz, H. C. [19]	'Nachhaltigkeit' (Sustainability) as sustained-yield forestry
The Oxford Dictionary of English [8]	'To keep in being; to continue in a certain state; to keep or maintain at the proper level or standard; to preserve the status of. 'To support life in; to provide for the life or bodily needs of; to furnish with the necessities of life; to keep'
UN WC on Envir. & Development [16]	Sustainable development as a development that 'meets the needs of the present generation without compromising the ability of future generations to meet their own needs'
The Natural Step [20]	In a sustainable society, nature is not subject to systematically increasing: 1. concentrations of substances extracted from the earth's crust. 2. concentrations of substances produced by society. 3. degradation by physical means. And, in that society: 4. people are not subject to conditions that systematically undermine their capacity to meet their needs.
Rees W. and Wackernagel M. [17]	Ecological footprint as the amount of land and water area a human population would hypothetically need to provide the resources required to support itself and to absorb its wastes, given prevailing technology
Heinberg R. [21]	Five axioms to define sustainability: 1) Any society that continues to use critical resources unsustainably will collapse. 2) Population growth and/or growth in rates of consumption of resources cannot be sustained. 3) The use of renewable resources must proceed at a rate that is less than or equal to the rate of natural replenishment. 4) The use of nonrenewable resources must proceed at a rate that is declining, and the rate of decline must be greater than or equal to the rate of depletion. 5) Substances introduced into the environment from human activities must be minimized and rendered harmless to biosphere functions. Where pollution from extraction and consumption of nonrenewable resources has proceeded at expanding rates for some time and threatens the viability of ecosystems, reduction in the rates of extraction and consumption of those resources may need to occur at a rate greater than the rate of depletion.
Tainter J. [10]	'1) Sustainability is an active condition of problem solving, not a passive consequence of consuming less. 2) Complexity is a primary problem-solving tool, including problems of sustainability. 3) Complexity in problem solving is an economic function, and can reach diminishing returns and become ineffective. 4) Complexity in problem solving does its damage subtly, unpredictably, and cumulatively over the long term. Sustainability must therefore be a historical science. 5) Sustainability may require greater consumption of resources rather than less. One must be able to afford sustainability. 6) The members of an institution may resort to resiliency as a strategy of continuity only when the option of sustainability is foreclosed. 7) A society or other institution can be destroyed by the cost of sustaining itself. To define sustainability in a specific context, the questions should be (i) Sustain what? (ii) For whom? (iii) How long? (iv) At what cost?'
Dillard J., Dujon V. and King M. [22]	Sustainability is often thought of as composed of three overlapping, mutually dependent goals: a) to live in a way that is environmentally sustainable, or viable over the very long-term, b) to live in a way that is economically sustainable, maintaining living standards over the long-term, and c) to live in a way that is socially sustainable, now and in the future. The social dimension of sustainability should be understood as both 1) the processes that generate social health and well-being now and in the future, and 2) those social institutions that facilitate environmental and economic sustainability now and for the future.
Polese M. and Stren R. [23]	Social sustainability as 'policies and institutions that have the overall effect of integrating diverse groups and cultural practices in a just and equitable fashion.'
Harris J. M. and Goodwin N. R. [24]	'A socially sustainable system must achieve fairness in distribution and opportunity, adequate provision of social services, including health and education, gender equity, and political accountability and participation.'
Hilty L. M. et al. [25]	For evaluating sustainability of ICT systems, three orders of effect need to be considered. 'First-order' or 'primary' effects: effects of the physical existence of ICT (environmental impacts of the production, use, recycling and disposal of ICT hardware). 'Second order' or 'secondary' effects: indirect environmental effects of ICT due to its power to change processes (such as production or transport processes), resulting in a modification (decrease or increase) of their environmental impacts. 'Third order' or 'tertiary' effects: environmental effects of the medium- or long-term adaptation of behaviour (e.g., consumption patterns) or economic structures due to the stable availability of ICT and the services it provides.
Mahaux M. (unpublished)	'For me, what matters is naively to make the world a better place, for this and all coming generations, and for all populations. That means keeping an environment in which it is great to live in and that can provide the resources to live well. That means people who live in harmony with each other, with their environment, who are free to think and able to do what will make them happy. Beyond fulfilling their basic needs, it's about realizing the human potential on earth.'

sustainability in terms of a set of dimensions. Discussion of this feedback revealed some sharply different interpretations of sustainability.

The core issue can be illustrated in terms of a preference over one or other of the diagrams shown in Figure 1. Fig. 1(a) shows a common visualization of sustainability in terms of three separate concerns. The key idea is that human society is only sustainable if it can be sustained in all three dimensions: social, economic and environmental. This view is incorporated into the *triple bottom line* approach, where companies account not just for financial returns, but also for benefits and impacts in the social and environmental spheres [27].

This approach is rejected as *weak* sustainability by those who argue that it's an error to seek to balance the three concerns. According to this view, Figure 1(a) is misleading, as the economy is really only a subsystem of society, which in turn is a subsystem of the environment (Figure 1(b)). To achieve *strong* sustainability, we have to acknowledge that there are fundamental biophysical limits that constrain the flows of natural resources on planet earth, and no arrangement of society can be considered sustainable unless it lives within these limits [28]. In this view, it is wrong to talk about sustainability in terms of a set of 'dimensions', as the concerns are strictly hierarchical.

The conflict between these two views plays out differently in different disciplines. In economics, it rests on the question of substitutability. Many economists assume that natural capital (the stock of natural resources) are infinitely substitutable with human capital (e.g., human ingenuity). If they are, then economic growth need not be constrained by biophysical limits. However, ecological economists dispute this, and argue that there are firm limits to substitutability, which implies there are limits on economic growth [29]. For social issues, the dispute centres on whether all aspects of social sustain-



(a) Weak Sustainability (adapted from [18])

(b) Strong Sustainability

(c) A more pragmatic view for software research (adapted from [26])

Figure 1: Competing Visualizations of Sustainability.

ability eventually lead to questions of distributional justice over access to (natural) resources, or whether there are other aspects of social sustainability (e.g., human rights) that arise independently from the question of how we allocate resources.

While we believe these questions are important, we do not believe they offer a useful starting point for software practitioners and researchers struggling with the question of what sustainability means for *them*. A more pragmatic view is shown in Figure 1(c), where sustainability is depicted as a learning process by which we move towards integrated thinking. Software practitioners tend to treat techno-centric concerns (e.g., software qualities and the economic value they create) separately from socio-centric concerns (how software can make people's lives better) and eco-centric concerns (protecting the environment). Rather than asking whether it is appropriate to balance these concerns, we should instead be asking *What methods and tools are needed to explore inter-dependencies between these concerns, and to foster more integrated and long-term thinking?*

In the past few decades, production and use of information technologies (IT) have had a dramatic effect on society, giving us new tools and new capabilities, but also generating a massive growth in demand for energy and other resources. Software systems, in particular, play a transformative role, as they enable dematerialization [30], drive consumption patterns for products, services, materials, and energy, and facilitate structural changes from consuming material goods towards consuming immaterial services, such as the shift to listening to music online instead of purchasing (and discarding) physical records and CDs. They also collect, manage and distribute information needed to understand long-running complex phenomena ranging from climate data to personal health records, and statistics on global equity and capital. As such, the software industry increasingly represents a central driver for innovation and economic prosperity, but simultaneously increases social inequity, as people without access and technical skills are left behind [31], and causes environmental damage, as consumption of technology grows [32].

The approach we have adopted is to focus on how we understand and take responsibility for the multiple interacting opportunities and impacts of software technology, including first, second and third order effects [33]. *First order effects* are impacts and opportunities created by the immediate existence of a software system, arising from its design features and flaws. *Second order effects* are those created by the ongoing use and application of the software, such as how it changes what we do and what we're capable of. *Third order effects* are the changes that occur through the aggregated behaviours of very large numbers of people using the technology over the medium to long term (e.g., energy demand, mass surveillance, etc). These effects play out across many domains.

Following Goodland [34] and Penzenstadler & Femmer [35], we identify five sustainability dimensions:

- **Environmental**: concerned with the long term effects of human activities on natural systems. This dimension includes ecosystems, raw resources, climate change, food production, water, pollution, waste, etc.
- **Social**: concerned with societal communities (groups of people, organizations) and the factors that erode trust in society. This dimension includes social equity, justice, employment, democracy, etc.
- **Economic**: focused on assets, capital and added value. This includes wealth creation, prosperity, profitability, capital investment, income, etc.
- **Technical**: refers to longevity of information, systems, and infrastructure and their adequate evolution with changing surrounding conditions. It includes maintenance, innovation, obsolescence, data integrity, etc.

- **Individual**: refers to the well-being of humans as individuals. This includes mental and physical well-being, education, self-respect, skills, mobility, etc.

These dimensions are interdependent. Cumulative effects from the individual dimension will bleed into the social one; effects from the environmental dimension into the individual, social, and economic, and so on. Yet, these dimensions provide a useful tool for dis-aggregating and analyzing relevant issues.

An understanding of these short and long term effects of software technology and how they play out over multiple dimensions can then lead to a consideration of leverage points [36]: where are the most effective places to intervene to achieve sustainability? Such interventions might be changes in the way we analyze and design software systems, or they might be changes in how we seek to apply software solutions to societal problems.

A growing concern among software researchers about these impacts, and a desire to find good leverage points, has inspired a number of workshops at software-related conferences dedicated to software and sustainability. For example, at ICSE'09, a special conference session explored the relationship between Software Engineering and climate change, which led to a series of workshops on software research and climate change (WSRCC), at Oopsla/Onward! in 2009, at ICSE in 2010, and at ECOOP in 2011. These led to a special issue of IEEE Software [37]. As interest grew and broadened, the community was brought together under the umbrella of the GREENS (Green & Sustainable Software) workshop series at ICSE through a workshop merger, held in 2012, 2013, and 2014, and another series of workshops on RE for Sustainable Systems (RE4SuSy) at REFSQ in 2012 and at the RE Conference in 2013 and 2014.

Other research communities have followed a similar path [5]. The HCI community has held an annual series of workshops on HCI and sustainability at CHI since 2009. The AI community runs a series of workshops on 'Computational Sustainability' which began with separate workshops at Cornell in 2009 and MIT in 2010, and a conference track at AAAI since 2011. The scientific computation community ran workshops on 'Sustainable Software for Science: Practice and Experiences (WSSSPE)' at SC'13 and SC'14, focusing more specifically on issues of technical sustainability. Finally, a new annual conference series on ICT for Sustainability was launched in 2013. At the same time, other communities with a long-term view on socio-technical systems, such as digital curation and preservation, have attempted to identify what sustainability concerns in software technology mean for them [6].

These communities share a sense that the design of software is critically important for sustainability. We take the view that all design has an impact on sustainability and all software has an impact on the world. Therefore, it is the responsibility of those who are involved in the creation of software to consider this impact carefully.

The various efforts described above tackle a wide range of different research questions, often with very little overlap. They are developing conceptual frameworks, techniques, and systems to understand different aspects of the problem [5]. Some seek to encourage reductions in consumption of energy and material goods, or to support changes in purchasing behavior. Others seek to use software capabilities to build smarter (lower impact) infrastructure. However, there is a lack of common understanding of the fundamental concepts of sustainability and how they apply, and a need for a common ground and consistent terminology. As such, persistent misperceptions occur, as researchers and practitioners disagree over whether we're even asking the right questions (see, for example, Strenger's essay on 'Designing for Resource Man' [38]). We lack a coherent framework with sound theoretical basis that can provide a well-understood trans-disciplinary basis for sustainability design [6], [9].

III. Towards a manifesto

Historically, a common vehicle for catalyzing communities and providing such a focus in comparable situations has been the genre of the manifesto. Communities have relied upon it to articulate their viewpoint, often to oppose a prevailing paradigm. Some manifestos very effectively captured key messages with an appeal beyond the originating community and hence provided a platform for subsequent thoughts and initiatives to develop. Examples in the software world include the GNU Manifesto[2] and the Agile Manifesto[3]. However, these have little in common in terms of their structure and content, and the manifesto is a delicate genre. What makes a manifesto a successful 'point of reference'?

As a preparation for RE4SuSy, the first author conducted an informal study to reflect on the nature and history of this genre, based on a review of about two dozen manifestos in the areas of SE, computer science and broader fields of product design, and secondary sources discussing the genre and its practical aspects. The purpose of this study was to understand the nature of the writing process and the possible implications of its product; enable a conscious choice as to whether the creation of a manifesto is a desirable mechanism with a positive impact; identify key elements to address and possible pitfalls to avoid; and derive a set of principles to guide the process.

The origin of the manifesto as a distinct genre can be traced to documents such as Luther's theses and the Communist manifesto. A fascinating account of the early history of manifestos in politics and art is offered by Puchner [39] who traces the distinct nature, rhetorics and effects of manifestos across the evolution of the genre up to the art and politics manifestos of the 20^{th} century. Fundamentally, the manifesto is a *speech act* [39]. While originally, it manifested the will of an authority, Luther and Marx morphed this act into one that assumes such authority. As such, the act becomes inherently one in future perfect tense [39]. A successful manifesto will have been effective in capturing a more commonly understood message and articulating it clearly enough to enable others to self-identify with the message. Here, the distinction between

[2]https://www.gnu.org/gnu/manifesto.html
[3]http://agilemanifesto.org/

declaring 'manifesto' in a title line and the rhetorical nature of the manifesto becomes visible. However, we also need to distinguish between the intentionally polarizing nature of earlier manifestos in politics, art, and design, and recent interpretations of manifestos which implicitly assume a much more conversational standpoint, aiming to initiate broader reconsideration rather than aiming to create a revolution.

What is a successful manifesto? An ideal manifesto can provide a focal point of reference and catalyze communities by phrasing key questions in accessible language appealing to a broad audience. It enables others to see connections and synergies and self-identify with the concerns articulated in the manifesto. It contributes to unifying the language about a subject and facilitates visible community building. It also facilitates the action of reaching out to related communities with a clear value proposition and provides arguments for the relevance of the topic, especially encouraging new community members to engage. As such, it can enable a clear communication of the benefits of engaging in the subject.

However, the manifesto has also been called a 'defunct format' that 'belongs to the early twentieth century' [40]. It arises from the nature of the genre that the creation of a manifesto brings with it the potential for pitfalls and negative consequences. The compact, shortened form of communication often assumed in a manifesto can appear dogmatic, and catchy language designed to be broadly appealing can ultimately hide the real complexity of underlying issues. A polarizing perspective can result in splinter groups rather than a unification, and the questioning of commonly accepted assumptions might alieniate rather than unite the audience. Hence, an intended focal point of reference can make others feel excluded rather than invited.

The acknowledgment of these risks was discussed in the early stages when initiating the collaborative process and has led us to articulate a set of principles, meant to guard the emerging group from the above mistakes, avoid groupthink, and foster a sustainable process. These have guided our work:

Principles, not techniques. The manifesto should focus on principles and values of sustainability, not on current techniques, specific models, and suggested approaches.

Scope. The intended scope is broad and inclusive, but clearly delimited. This is inherently difficult to define and achieve, but the principle has repeatedly been brought in when shaping the manifesto to provide a balance and a focus consistent with what the authors are confident to adddress.

Emerging structure. We believe that the content and the structure of the manifesto needed to emerge from a common set of elements arising from the discussion, initially at the RE4SuSy workshop. This has prompted the process to be very bottom-up and the structure strongly grounded in what emerged from the contributions of the initial group of workshop participants. Facilitation within and beyond the workshop was strongly focused on providing a stable structure, documenting outcomes, and facilitating the discussion.

Participation and transparency. The discussion was initiated within the workshop, and all participants of the workshop were invited to the subsequent process. No conditions are set for entering the discussion process. The initial document was released publicly for comments[4] and presented at the RE closing session, and direct discussions with a number of experts in the fields of sustainability have been initiated. Broader engagement with the community includes a discussion panel held at the SPLC'14 conference[5], a workshop at the iConference 2015[6], a discussion at WSSSPE2[7], and a special discussion session planned for RE4SuSy'2015.

Conversation over consensus. This acknowledges that while internal consensus is critical, universal consensus is an elusive, and perhaps undesirable, goal. The intention for external engagement is to initiate a dialogue rather than aim for full consensus within an extremely broad community.

Minimal and adaptive process. In line with the focus on emergent content and structure, we designed only a minimal process and created the required support structure only as necessary. This eventually included an email list, regular Google hangouts, and a shared folder.

Synchronous collaboration. The elements of the manifesto, at all times, were written and edited in fully synchronous collaboration, first in person during the RE conference, then virtually on Google Drive[8].

Iterative evolution. A vision was formulated early on, but no specific milestones or objectives were set and the process was intended to be incremental, iterative, and open-ended.

As preparation for the workshop discussions, we elicited initial responses, gathered a sense of the common ground through a vote on sustainability definitions, and collected initial thoughts and principles of sustainability for a possible manifesto.

At the workshop, all participants supported the collaborative writing of a manifesto and worked through a number of brainstorming sessions to collect starting points for central statements in the categories *context*, *purpose*, *scope*, *principles and values*, *best practices*, and *prescriptions*. After the workshop, the core set of interested collaborators continued to work on the manifesto throughout the conference via a number of intense face-to-face writing sessions, each between two and five hours. As a result, by the end of the third day of the conference, an initial version of the manifesto was released publicly and presented at the Workshop Highlights Session of RE'14. A combination of weekly synchronous collaborative writing sessions and individual contributions and email discussions continued over several months.

IV. The Karlskrona Manifesto

The latest release of the manifesto is included here. It is becoming a living document at http://sustainabilitydesign.org.

[4]http://bit.ly/RE14Manifesto
[5]http://www.splc2014.net/panels.html
[6]http://ischools.org/the-iconference/
[7]http://wssspe.researchcomputing.org.uk/wssspe2
[8]This paper was developed in similar fashion on overleaf.com.

THE KARLSKRONA MANIFESTO FOR SUSTAINABILITY DESIGN

Version 0.5, January 2015

Introduction
As software practitioners and researchers, we are part of the group of people who design the software systems that run our world. Our work has made us increasingly aware of the impact of these systems and the responsibility that comes with our role, at a time when information and communication technologies are shaping the future. We struggle to reconcile our concern for planet Earth and society with the work that we do. Through this work we have come to understand that we need to redefine the narrative on sustainability and the role it plays in our profession. What is sustainability, really? We often define it too narrowly. Sustainability is at its heart a systemic concept and has to be understood on a set of dimensions, including social, environmental, economic, individual, and technical. Sustainability is fundamental to our society. The current state of our world is unsustainable in more ways that we often recognize. Technology is part of the dilemma and part of possible responses. We often talk about the immediate impact of technology, but rarely acknowledge its indirect and systemic effects. These effects play out across all dimensions of sustainability over the short, medium and long term. Software in particular plays a central role in sustainability. It can push us towards growing consumption of resources, growing inequality in society, and lack of individual self-worth. But it can also create communities and enable thriving of individual freedom, democratic processes, and resource conservation. As designers of software technology, we are responsible for the long-term consequences of our designs. Design is the process of understanding the world and articulating an alternative conception on how it should be shaped, according to the designer's intentions. Through design, we cause change and shape our environment. If we don't take sustainability into account when designing, no matter in which domain and for what purpose, we miss the opportunity to cause positive change.

We recognize that
there is a rapidly increasing awareness of the fundamental need and desire for a more sustainable world, and there is a lot of genuine desire and goodwill - but this alone can be ineffective unless we come to understand that…

There is a narrow perception of sustainability that frames it as protecting the environment or being able to maintain a business activity. **Whereas** as a systemic property, sustainability does not apply simply to the system we are designing, but most importantly to the environmental, economic, individual, technical and social contexts of that system, and the relationships between them.
There is a perception that sustainability is a distinct discipline of research and practice with a few defined connections to software. **Whereas** sustainability is a pervasive concern that translates into discipline-specific questions in each area it applies.
There is a perception that sustainability is a problem that can be solved, and that our aim is to find the 'one thing' that will save the world. **Whereas** it is a 'wicked problem' - a dilemma to respond to intelligently and learn in the process of doing so; a challenge to be addressed, not a problem to be solved.
There is a perception that there is a tradeoff to be made between present needs and future needs, reinforced by a common definition of sustainable development, and hence that sustainability requires sacrifices in the present for the sake of future generations. **Whereas** it is possible to prosper on this planet while simultaneously improving the prospects for prosperity of future generations.
There is a tendency to focus on the immediate impacts of any new technology, in terms of its functionality and how it is used. **Whereas** following orders of effects have to be distinguished: *Direct, first order effects* are the immediate opportunities and effects created by the physical existence of software technology and the processes involved in its design and production. *Indirect, second order effects* are the opportunities and effects arising from the application and usage of software. *Systemic, third order effects*, finally, are the effects and opportunities that are caused by large numbers of people using software over time.
There is a tendency to overly discount the future - in fact, the far future is discounted so much that it is considered for free (or worthless). Discount rates mean that long-term impacts matter far less than current costs and benefits. **Whereas** the consequences of our actions play out over multiple timescales, and the cumulative impacts may be irreversible.
There is a tendency to think that taking small steps towards sustainability is sufficient, appropriate, and acceptable. **Whereas** incremental approaches can end up reinforcing existing behaviours and lure us into a false sense of security. However, current society is on a path that is so far from sustainability that deeper transformative changes are needed.
There is a tendency to treat sustainability as a desirable quality of the system that should be considered once other priorities have been established. **Whereas** sustainability is not in competition with a specific set of quality attributes against which it has to be balanced - it is a fundamental precondition for the continued existence of the system and influences many of the goals to be considered in systems design.
There is a desire to identify a distinct completion point to a given project, so that success can be measured at that point, with respect to a pre-ordained set of criteria. **Whereas** measuring success at one point in time fails to capture the effects that play out over multiple timescales, and so tells us nothing about long-term success. Criteria for success change over time as we experience those impacts.
There is a narrow conception of the roles of system designers, developers, users, owners, and regulators and their responsibilities, and there is a lack of agency of these actors in how they can fulfill these responsibilities. **Whereas** sustainability imposes a distinct responsibility on each one of us, and that responsibility comes with a right to know the system design and its status, so that each participant is able to influence the outcome of the technology application in both design and use.
There is a tendency to interpret the codes of ethics for software professionals narrowly to refer to avoiding immediate harm to individuals and property. **Whereas** it is our responsibility to address the potential harm from the 2nd and 3rd-order effects of the systems we design as part of our design process, even if these are not readily quantifiable.
As a result, even though the importance of sustainability is increasingly understood, the majority of software systems are created unsustainably and often decrease sustainability instead of increasing it.

Thus, we propose the following initial set of principles and commitments:

Sustainability is systemic. Sustainability is never an isolated property. Systems thinking has to be the starting point for the transdisciplinary common ground of sustainability.
Sustainability has multiple dimensions. We have to include those dimensions into our analysis if we are to understand the nature of sustainability in any given situation.
Sustainability transcends multiple disciplines. Working in sustainability means working with people from across many disciplines, addressing the challenges from multiple perspectives.
Sustainability is a concern independent of the purpose of the system. Sustainability has to be considered even if the primary focus of the system under design is not sustainability.
Sustainability applies to both a system and its wider contexts. There are at least two spheres to consider in system design: the sustainability of the system itself and how it affects the sustainability of the wider system of which it will be part of.
System visibility is a necessary precondition and enabler for sustainability design. Strive to make the status of the system and its context visible at different levels of abstraction and perspectives to enable participation and informed responsible choice.
Sustainability requires action on multiple levels. Seek interventions that have the most leverage on a system and consider the opportunity costs: Whenever you are taking action towards sustainability, consider whether this is the most effective way of intervening in comparison to alternative actions (leverage points).
It is possible to meet the needs of future generations without sacrificing the prosperity of the current generation. Innovation in sustainability can play out as decoupling present and future needs. By moving away from the language of conflict and the trade-off mindset, we can identify and enact choices that benefit both present and future.
Sustainability requires long-term thinking. Consider multiple timescales, including longer-term indicators in assessment and decisions.

Signed,
Christoph Becker, Ruzanna Chitchyan, Leticia Duboc, Steve Easterbrook, Martin Mahaux, Birgit Penzenstadler, Guillermo Rodriguez-Navas, Camille Salinesi, Norbert Seyff, Colin Venters, Coral Calero, Sedef Akinli Kocak and Stefanie Betz.

V. Implications for Software Engineering

What implications do these principles and commitments have on Software Engineering? The present section focuses on the implications and questions that the principles advocated in the manifesto raise for SE research and practice.

While some software systems have very explicit sustainability goals, for other cases the role of sustainability is more subtle. In practice, the opportunities and risks raised through such interventions have to be understood from multiple perspectives. This requires conceptual frameworks, but also a culture that welcomes, encourages and rewards this understanding and enables these perspectives to be adopted in the professional practice of system analysts and designers.

In this practice, sustainability cannot simply be seen as *a quality* of the systems we design, Crucially, we must distinguish between a [solution-oriented] system quality and a [problem-oriented] concern i.e. an 'interest in a system relevant to one or more of its stakeholders' [41]. Considering only the system under design from a technical and economic perspective, the [technical] sustainability of a system architecture, as defined in [42], is clearly a system quality and can be measured and improved by techniques such as evolution scenario analysis, architecture compliance checks, and tracking of architecture-level code metrics. However, in the overall design of the complex socio-technical system that contains this system architecture, sustainability needs to be treated as a design concern of interest to multiple stakeholders that will drive specific capabilities and qualities in the system [6]. As such, it will interact in different ways with technical features and system qualities. Understanding these interactions and designing the system accordingly is a challenge that current methods, techniques and tools do not fully address, and needs a broader, more holistic perspective than product quality models and architectural metrics can capture.

Consider an imaginary software company called *CodeIT*. Their next project is developing a community car sharing application that specifically intends to satisfy the needs of a suburban community in a western country such as the US. Kodi, the project manager, is sensitive to the impact of car traffic on the environment and painfully aware of the technical debt carried by the project she just completed. As such, she is determined to make her best effort to design responsibly and effectively this time. Is she also aware of the complex dependencies that will surface in this system, and will she be able to contribute both to the sustainability of this system and to the envisioned positive impact it should have?

Kodi could start an investigation into the concerns for economic, environmental, individual, social, and technical sustainability by asking initial guiding questions similar to those suggested in [43]: (1) Does the system have an explicit sustainability purpose? Can we analyze it in depth using sustainable development scenario techniques [44]? (2) Which direct impact does the system have on its operational environment? Which indirect effects can we identify? (3) Who are the stakeholders for sustainability? Who are the domain sustainability experts, policy makers, and legal representatives?

These questions highlight that Requirements Engineering is a key area where systems level thinking can be applied to identify sustainability concerns, as it translates the domain-dependent goals and concerns into technical requirements that can be realized in the implementation of a software system. Requirements engineers can question user needs, shape people's expectations and use sustainability concerns and apparent conflicts creatively as drivers for innovation.

Currently, however, Kodi is unlikely to even begin asking these questions. Corporate culture will often not encourage and reward her, and the company's incentive structure may instead favour short-term thinking based on the economic paradigms that corporations are operating in. Kodi's SE education will not have prepared her for this responsibility nor equipped her with the mindset to raise these concerns.

Using the manifesto as a guide, how does a commitment to the principles affect the car sharing scenario?

Sustainability is systemic. This raises fundamental questions about the relationship between the proposed software and the problem it is attempting to solve. Often, software engineers fall into the trap of solutionism [45]. Kodi's questions are based on the assumption that the car sharing app will solve a real problem. However, sometimes, the system the customer wants and the system that should be built are quite different. Choosing appropriate system boundaries and actively critiquing assumptions about these boundaries is important, as any given choice will privilege some stakeholders and their concerns, and marginalize others [46].

Sustainability has multiple dimensions. Sustainability transcends multiple disciplines. Sustainability design in this scenario will require cross-disciplinary expertise covering transportation systems, carbon emissions, social network effects, effects on family structures, but most importantly, the interaction between these and additional aspects. The problem of transportation in suburban communities is often a dilemma to be addressed rather than a problem to be solved, and success may be a moving target.

Conceptual models, techniques and tools are needed to communicate, represent, and visualize relationships between software, systems, and particular aspects of sustainability in their social, economic, technical, and natural environment.

Sustainability applies to both a system and its wider contexts. Sustainability requires action on multiple levels.

The car sharing application could focus on action on multiple levels. For example, the design of the system could focus on making sharing effortless - an obvious choice. If car booking becomes as efficient and effortless as using one's own car, the volume of traffic could potentially increase, countering a key argument originally brought forward in support of the car sharing project. The application design, however, could also attempt to facilitate joint usage to support a reduction in total traffic.

System visibility is a necessary precondition and enabler for sustainability design. Each participant carries a distinct responsibility, and this comes with the need to be informed about the status and structure of the socio-technical system

under design and the right to influence the outcomes.

It is possible to meet the needs of future generations without sacrificing the prosperity of the current generation. Sustainability requires long term thinking. The project's effects will need to be studied over time rather than at the initial product release date. A long-term requirement that may surface is the availability of authentic, reliable records about the system usage and its trends beyond the system life span so that the phenomena associated with such an initiative can be analyzed. As a typical case involving the interest of stakeholders that are not commonly involved in such scenarios, this example also illustrates that often, there does not need to be a trade-off between these future needs and current needs. Well-defined data models and records management principles benefit current stakeholders as well, and ignoring them increases the technical debt of the system design.

If she would ask questions like these, could Kodi convince the stakeholders to see sustainability as the first and foremost goal, to formulate a project vision with sustainability as a precondition rather than an additional requirement?

Kodi will need clear guidelines for assessing sustainability on multiple dimensions and multiple timescales. If she is to convince the stakeholders with robust arguments and evidence, she will demand empirically evaluated methods and tools, and metrics and measures for evaluating effects and their interactions. Showcases that demonstrate how sustainability concerns can be integrated and balanced with existing quality attributes and business constraints can support her in understanding and communicating the benefits and opportunities.

Incentive systems are a way to enable Kodi to ask these questions. Can we build reward structures that foster the production of sustainable systems? Is Kodi encouraged to pursue sustainable practice? Perhaps, one of the best leverage points would be to redesign the reward structure in the company to encourage those who attempt to apply sustainability design principles. For example, this could involve focusing interventions on strong leverage points rather than weak ones through fostering responsible consumption instead of improving energy efficiency, or by giving people information that empowers them to take action themselves (system visibility).

Universities have started to address sustainability concerns (beyond technical sustainability) in SE education [47]. The SE curricula are an opportunity to promote awareness and provide software engineers with the skills to take into account other disciplines. Software engineers need systems thinking skills [45] and sufficient understanding of sustainability to understand the implications of software systems on different dimensions of sustainability, ask relevant questions, involve the right stakeholders, and help to perform the required analysis. However, current standard references and textbooks for SE do not yet address such topics [48]. For instance, the term 'sustainability' features only once in the latest edition of SWEBOK [49], under the section finance.

Finally, the codes of ethics of professional associations such as ACM and IEEE may need to be revisited. For example, while the current ACM code of ethics [50] acknowledges that actions with good intentions 'may lead to harm unexpectedly', it defines harm as 'injury or negative consequences, such as undesirable loss of information, loss of property, property damage, or unwanted environmental impacts'. This is not sufficient to cover the potential harmful impacts of technology over multiple timescales and across all sustainability dimensions. The code does not consider second and third order effects adequately in stating, 'to minimize the possibility of indirectly harming others, computing professionals must minimize malfunctions by following generally accepted standards for system design and testing'.

In contrast, the UK Standard for Professional Engineering Competence (UK-SPEC) demonstrates explicit awareness and commitment to sustainability. It defines specific competencies and commitments for engineering roles with different levels of responsibility. For each, responsibility for sustainability plays out in specific ways. For example, engineers need to demonstrate their commitment and competencies to 'undertake engineering activities in a way that contributes to sustainable development', including the 'ability to ... progress environmental, social and economic outcomes simultaneously' [51]. They are encouraged to 'do more than just comply with legislation and codes' and made aware that they need to 'seek multiple views to solve sustainability challenges' [52]. UK-SPEC also covers ethical principles and states that '[c]odes of Conduct should oblige members to ... Act in accordance with the principles of sustainability, and prevent avoidable adverse impact on the environment and society.' [51]

It is questionable whether the current codes of the SE profession meet these expectations.

VI. Conclusions and Outlook

Increasing attention is being paid to the broad effects of software on society and the need to embody longer-term thinking, ethical responsibility, and an understanding of sustainability into the design of software systems. However, the software profession lacks a common ground that articulates its role in sustainability design, and a long rooted set of misperceptions persist in research, theory, and practice. To truly make progress in understanding the role software plays in the choices we make as designers of the systems at the backbone of our society, we need to understand the nature of sustainability and find a common ground for a conceptual framework.

This article presented a collaborative cross-disciplinary effort to foster the establishment of such common ground. The paper consolidates and expands the current understanding of sustainability concerns within the SE community. The manifesto unveils and straightens out a number of common misunderstandings with respect to sustainability and SE. The principles of sustainability design pose new challenges to research on sustainability in and through software engineering.

The manifesto will undergo future iterations and stay a living, publicly accessible document. We envision specific extensions articulating the concrete impact that the core principles should have in particular areas such as RE or software architecture. The set of fundamental principles stated in the

manifesto is a contribution to the conversation on the role of the SE profession both in undermining and in enabling a sustainable future for our planet.

VII. Acknowledgments

The authors thank all signatories of the manifesto; Sedef Akinli Kocak and Coral Calero, for comments and contributions; Emily Maemura for contributions to the study of manifestos; and the reviewers for valuable comments and suggestions. Part of this work was supported by the Vienna Science and Technology Fund (WWTF) through the project *BenchmarkDP* (ICT2012-046), by the Deutsche Forschungsgemeinschaft under project EnviroSiSE (grant PE 2044/1-1), by FAPERJ (No 41/2013), by CNPQ (No 14/2014) and by NSERC (RGPIN-2014-06638).

References

[1] *Software Engineering: Report of a conference sponsored by the NATO Science Committee, Garmisch, Germany*. Brussels, Scientific Affairs Division, NATO (1969), 7-11 Oct. 1968.

[2] P. Kruchten, "Technical Debt: From Metaphor to Theory and Practice," *IEEE Software*, vol. 29, no. 6, pp. 18–21, Nov. 2012.

[3] P. G. Neumann, "The Foresight Saga, Redux," *Commun. ACM*, vol. 55, no. 10, pp. 26–29, Oct. 2012.

[4] T. Kuny, "The digital dark ages? Challenges in the preservation of electronic information," *International preservation news*, no. 17, 1998.

[5] L. M. Hilty and B. Aebischer, "ICT for Sustainability: An Emerging Research Field," in *ICT Innovations for Sustainability*, L. M. Hilty and B. Aebischer, Eds. Springer Int. Publishing, 2015, pp. 3–36.

[6] C. Becker, "Sustainability and Longevity: Two Sides of the Same Quality?" in *RE4SuSy: Proceedings of the Third Int. Workshop on RE for Sustainable Systems*. Karlskrona, Sweden: CEUR-WS 1216, 2014.

[7] C. Becker, R. Chitchyan, L. Duboc, S. Easterbrook, M. Mahaux, B. Penzenstadler, G. Rodriguez-Navas, C. Salinesi, N. Seyff, C. Venters, C. Calero, S. A. Kocak, and S. Betz, "The Karlskrona manifesto for sustainability design," 2014, http://arxiv.org/abs/1410.6968.

[8] *The Oxford Dictionary of English*. Oxford University Press, 2010, sustainability.

[9] C. C. Venters, "Software sustainability: The modern tower of Babel," in *RE4SuSy: Proceedings of the Third Int. Workshop on RE for Sustainable Systems*. Karlskrona, Sweden: CEUR-WS 1216, 2014.

[10] J. A. Tainter, "Social complexity and sustainability," *Journal of Ecological Complexity*, no. 3, pp. 91–103, 2006.

[11] M. Lehman, "Programs, life cycles, and laws of software evolution," *Proceedings of the IEEE*, vol. 68, no. 9, pp. 1060–1076, Sept 1980.

[12] M. M. Lehman, "Laws of software evolution revisited," in *Proceedings of the 5th European Workshop on Software Process Technology*, ser. EWSPT '96. London, UK, UK: Springer-Verlag, 1996, pp. 108–124.

[13] K. H. Bennett and V. T. Rajlich, "Software maintenance and evolution: A roadmap," in *Proc. of the Conference on The Future of Software Engineering (ICSE '00)*. ACM, 2000, pp. 73–87.

[14] R. S. Arnold, *Software Maintenance Workshop, Moneterey, California, December 6-8, 1983: Record*. IEEE Computer Society Press, 1984.

[15] Z. Durdik, B. Klatt, H. Koziolek, K. Krogmann, J. Stammel, and R. Weiss, "Sustainability guidelines for long-living software systems," in *28th Intl. Conf. on Software Maintenance*, Sept 2012, pp. 517–526.

[16] G. H. Brundtland and World Commission on Environment and Development, *Our common future*. Oxford University Press Oxford, 1987.

[17] M. Wackernagel and W. E. Rees, *Our Ecological Footprint: Reducing Human Impact on the Earth.* New Society Publishers, 1996.

[18] SustainAbility. (2010) Sustainability: Can our society endure? [Online]. Available: http://www.sustainability.com/sustainability

[19] C. Hans Carl von, *Sylvicultura Oeconomica, oder haußwirthliche Nachricht und Naturmäßige Anweisung zur wilden Baum-Zucht*, 1713.

[20] The Natural Step. (2014) The Four System Conditions of a Sustainable Society. [Online]. Available: http://www.naturalstep.org/en/the-system-conditions

[21] R. Heinberg, *Peak everything: Waking up to the century of declines*. New Society Publishers, 2007.

[22] J. Dillard, V. Dujon, and M. King, *Understanding the Social Dimension of Sustainability*. Taylor & Francis, 2008.

[23] M. Polèse and R. E. Stren, *The social sustainability of cities: diversity and the management of change*. University of Toronto Press, 2000.

[24] J. M. Harris and N. R. Goodwin, *"Volume Introduction" in A Survey of Sustainable Development: Social And Economic Dimensions.*, ser. Frontier Issues in Economic Thought. Island Press, 2001.

[25] L. M. Hilty, P. Arnfalk, L. Erdmann, J. G. 0002, M. Lehmann, and P. A. Wäger, "The relevance of information and communication technologies for environmental sustainability - A prospective simulation study." *Environmental Modelling and Software*, no. 11, pp. 1618–1629.

[26] R. Dodds and R. Venables, *Engineering for sustainable development: Guiding principles*. The Royal Society of Engineering, 2005.

[27] J. Elkington, "Enter the triple bottom line," in *The triple bottom line: Does it all add up?*, A. Henriques and J. Richardson, Eds. Routledge, 2004, pp. 1–16.

[28] E. Neumayer, *Weak Versus Strong Sustainability: Exploring the Limits of Two Opposing Paradigms*. Edward Elgar, 2003.

[29] H. Daly, *Steady-State Economics: Second Edition*. Island Press, 1991.

[30] L. M. Hilty et al., "The relevance of information and communication technologies for environmental sustainability — A prospective simulation study," *Environmental Modelling & Software*, 2006.

[31] M. Graham, S. A. Hale, and M. Stephens, *Geographies of the World's Knowledge*. Convoco! Edition, 2011.

[32] E. Williams, "Environmental effects of information and communications technologies." *Nature*, vol. 479, no. 7373, pp. 354–8, Dec. 2011.

[33] Forum for the Future, "The impact of ICT on sustainable development," in *EITO - European Information Technology Observatory*, A. M. Franfurt, Ed. European Observatory Interest Group, 2002, pp. 250–283.

[34] R. Goodland, *Sustainability: Human, social, economic and environmental*. John Wiley & Sons, 2002.

[35] B. Penzenstadler and H. Femmer, "A generic model for sustainability with process-and product-specific instances," in *Proceedings of the 2013 workshop on Green in/by software engineering*. ACM, 2013, pp. 3–8.

[36] D. H. Meadows, "Leverage Points: Places to Intervene in a System," The Sustainability Institute, Tech. Rep., 1999.

[37] S. M. Easterbrook, P. N. Edwards, V. Balaji, and R. Budich, "Guest Editors' Introduction: Climate Change - Science and Software," *IEEE Software*, vol. 28, no. 6, pp. 32–35, Nov. 2011.

[38] Y. Strengers, "Smart energy in everyday life: Are you designing for resource man?" *interactions*, vol. 21, no. 4, pp. 24–31, Jul. 2014.

[39] M. Puchner, *Poetry of the revolution : Marx, manifestos, and the avant-gardes*. Princeton: Princeton University Press, 2006.

[40] H. U. Obst, "Manifestos for the Future," 2010. [Online]. Available: http://www.e-flux.com/journal/manifestos-for-the-future/#_ftn12

[41] ISO/IEC/IEEE, "ISO/IEC/IEEE 42010:2011 - Systems and software engineering – Architecture description," online, 2011.

[42] H. Koziolek, D. Domis, T. Goldschmidt, and P. Vorst, "Measuring architecture sustainability," *Software, IEEE*, vol. 30, no. 6, Nov 2013.

[43] B. Penzenstadler, "Infusing Green: Requirements Engineering for Green in and through software systems," in *RE4SuSy: Proceedings of the Third Int. Workshop on RE for Sustainable Systems*, 2014.

[44] S. Bell and S. Morse, *Sustainability Indicators — Measuring the Immeasurable?* Earthscan, 2008, 2nd Ed.

[45] S. Easterbrook, "From Computational Thinking to Systems Thinking," in *Proc. ICT4S*. Atlantis Press, 2014.

[46] G. Midgley, I. Munlo, and M. Brown, "The theory and practice of boundary critique: developing housing services for older people," *Journal of the Operational Research Society*, pp. 467–478, 1998.

[47] E. Eriksson and D. Pargman, "ICT4S reaching out: Making sustainability relevant in higher education," in *Proc. ICT4S*. Atlantis Press, 2014.

[48] B. Penzenstadler and A. Fleischmann, "Teach sustainability in software engineering?" in *24th IEEE-CS Conference on Software Engineering Education and Training*, 2011.

[49] A. Abran and P. Bourque, *SWEBOK: Guide to the software engineering Body of Knowledge*. IEEE Computer Society, 2004.

[50] (1992) ACM Code of Ethics. [Online]. Available: http://www.acm.org/about/code-of-ethics

[51] *UK Standard for Professional Engineering Competence (UK-SPEC)*. The Engineering Council, 2014.

[52] *Guidance on Sustainability for the Engineering Profession*. The Engineering Council, 2009. [Online]. Available: http://www.engc.org.uk/about-us/sustainability