The Elephant in the Room – Educating Practitioners on Software Development for Sustainability

The Karlskrona Alliance
FHNW & University of Zurich
Switzerland
norbert.seyff@fhnw.ch

***Abstract*—Software engineering as a discipline has recognized that software systems can have an impact on sustainability and researchers have started to investigate this topic. New knowledge is being created and methods and tools that support software engineering for sustainability are emerging. Several universities have started to train their students in software engineering for sustainability. But what about the millions of SE practitioners who are already designing and developing future software systems – how would they know about making systems sustainable? Currently, there is a lack of consolidated guidance. In this paper, we promote the idea of developing a standardized curriculum and handbook as a basis for training and certification of practitioners in sustainable software development. This would not only increase their market value, but would also have a positive impact on their companies and could help to satisfy customers who demand sustainable software systems.**

I. Context

The high dependency of modern society on software systems has led to the increasing importance of sustainability in the field of software engineering, with at least two different perspectives: sustainable software and software engineering for sustainability (SE4S). The former addresses the principles, practices and processes that contribute to the longevity of software, i.e. technical sustainability, while the latter focuses on software systems to support one or more dimensions of sustainability that address issues outside the software systems themselves. This is a view supported by Hilty and Aebischer [1], who postulate that two things are essential to make ICT work in favour of sustainability: a) To stop the growth of software systems' own footprint and b) find ways to use ICT as an enabler to reduce the footprint of society's production and consumption. When designing software for sustainability the requirements are the key as they build the basis of the system under design [2]. The overlapping fields of software engineering and systems engineering both depend heavily on the quality of the requirements elicited and specified. Both fields have existing published bodies of knowledge SWEBOK [3] and SEBoK [4] but their treatment of engineering for sustainability is minimal, with SEBoK having a short section on Environmental Engineering.

II. The Challenge

While traditional SE methods and tools within various SE disciplines such as requirements engineering (RE) do not explicitly facilitate discussion of sustainability-related concerns, research suggests that existing SE techniques, approaches and methods can serve as a starting point for practitioners to integrate sustainability into their practice [5]. Chitchyan et al. [6] identified several techniques that help support sustainability in RE and demonstrated the application of some of these techniques through two case studies. Although several novel approaches are emerging, software professionals lack, for example, tools and methodological support to integrate the analysis of sustainability impacts of ICT products and services into software development practices [7]. Furthermore, evidence suggests that software engineers still generally lack the awareness that ICT products and services have an impact on sustainability [6]. Realising this knowledge gap, the central challenge addressed in this paper is therefore how to train practitioners in software engineering for sustainability.

III. The Anticipated Solution

The anticipated solution is a syllabus and handbook that enables training providers to come up with 2 to 3 day training course for practitioners to make them familiar with essential and industry-relevant content on software engineering for sustainability. Identifying this content is challenging. The main contribution of this Extended Abstract, besides recognizing this particular challenge, is to identify potential content for such a training course. A draft outline of its content consisting of several modules is discussed below and was identified through brainstorming and discussions by the authors of this paper:

Definition of Sustainability and Scoping of System's impact: We can only sustain something if we clearly define what it is, for whom we sustain it, for how long, and at what cost [8], and therefore we need a clear definition and a template for scoping of system's impact. In addition, we define the dimensions of sustainability across time [2], [9]. This is the foundation for the following parts of the course.

Business Cases and Sustainability: To strengthen the argumentation for integrating sustainability into software development, we provide a review of success case examples for sustainable businesses and point towards methods that help phrase and specify a business case for sustainability, e.g. [10].

Stakeholders for Sustainability: We give a wider view of stakeholders, of the advocates and opponents for sustainability, affected and decision makers, that all need to be included for maximizing the understanding around the sustainability impacts of a system [11].

Information sources: A wide range of sources of information is available and requires consideration for the dimensions of sustainability, e.g. legal frameworks, constraints, etc. This part encompasses an overview of types of standards and reference frameworks for sustainability, e.g. ISO standards [12], [13], Doughnut Model [14].

Ethical Considerations for Sustainability: This part introduces value sensitive design [15] from a perspective of sustainability and suggests concrete tools to work with, e.g. ethicalexplorer [16]. In a broader context, the EU Responsible Research and Innovation (RRI) toolkit addresses similar issues [17].

Sustainability Analysis: To get a preliminary assessment of potential positive and negative impacts of a system, we propose to use an analysis framework that systematically breaks down the sustainability concerns that could arise for a small, growing, and extrapolated large user base. We illustrate such an analysis using SusAF [9] and the SDG impact assessment tool [18].

Sustainability Visioning: From a preliminary analysis, we can develop narratives that help in illustrating benefits and risks of a system, such that they can be explained adequately to diverse sets of stakeholders. This may include the use of design fiction [19].

RE Methods: Concrete requirements engineering methods can easily be adapted to explicitly include sustainability. We are demonstrating this with Easy Win Win for Sustainability [20].

Architecture: For evaluating design on the architecture level, we refer to the use of models like SAF [21] as well as the works around technical debt [22].

Quality models: This module presents the use of quality models that focus on sustainability and adequate KPIs for assessing the impacts of taken measures, for example [23].

Continuous evolution and iterative assessment: Finally, sustainability is like balance - it is not a static state, it is dynamic and always evolving, therefore subject to continuous adaptation and iterative assessment [24].

IV. Discussion & Conclusion

The paper makes a first proposal on content that could become part of a syllabus and handbook for a training course on software engineering for sustainability for practitioners. The content was derived by brainstorming and discussions of the authors. With this first proposal, we hope to stimulate the discussion on what is essential knowledge about software engineering for sustainability for practitioners. The proposed content needs to be validated and discussed with fellow researchers and practitioners to better understand its relevance also from an industrial perspective. Furthermore, support from organisations that promote training in industry, such as IREB (International Requirements Engineering Board), is needed to make our vision a reality. We believe that the time is ripe for such standardised training and foresee that certification will become important in this regard. Certification will help participants to elevate their careers and with them as employees, companies can communicate their competence in this topic. In addition, we expect that users, customers of software, will ask for software developed by developers who know about software engineering for sustainability.

References

[1] L. M. Hilty and B. Aebischer, "Ict for sustainability: An emerging research field," in *ICT innovations for Sustainability*. Springer, 2015, pp. 3–36.

[2] C. Becker, S. Betz, R. Chitchyan, L. Duboc, S. M. Easterbrook, B. Penzenstadler, N. Seyff, and C. C. Venters, "Requirements: The key to sustainability," *IEEE Software*, vol. 33, no. 1, pp. 56–65, 2015.

[3] P. Bourque and R. E. Fairley, "Guide to the Software Engineering Body of Knowledge, Version 3.0," http://www.swebok.org, 2014.

[4] BKCASE Editorial Board, "Guide to the Systems Engineering Body of Knowledge (SEBoK)," https://www.sebokwiki.org, 2020.

[5] Cabot et al., "Integrating sustainability in decision-making processes: A modelling strategy," in *2009 31st International Conference on Software Engineering-Companion Volume*. IEEE, 2009, pp. 207–210.

[6] R. Chitchyan, S. Betz, L. Duboc, B. Penzenstadler, S. Easterbrook, C. Ponsard, and C. Venters, "Evidencing sustainability design through examples," 2015.

[7] S. Oyedeji and B. Penzenstadler, "Karlskrona manifesto: Software requirement engineering good practices." in *RE4SuSy@ RE*, 2018, pp. 15–23.

[8] J. A. Tainter, "Social complexity and sustainability," *ecological complexity*, vol. 3, no. 2, pp. 91–103, 2006.

[9] Duboc et al., "Requirements engineering for sustainability: an awareness framework for designing software systems for a better tomorrow," *Requirements Engineering*, vol. 25, no. 4, pp. 469–492, 2020.

[10] M. Hoveskog, F. Halila, M. Mattsson, A. Upward, and N. Karlsson, "Education for sustainable development: Business modelling for flourishing," *Journal of Cleaner Production*, vol. 172, pp. 4383–4396, 2018.

[11] B. Penzenstadler and C. Venters, "Software engineering for sustainability: Tools for sustainability analysis," in *Digital Technology and Sustainability: Engaging the Paradox*. Routledge, 2018, pp. 103–121.

[12] A. S. Morris, *ISO 14000 environmental management standards: Engineering and financial aspects*. John Wiley & Sons, 2004.

[13] R. Frost, "Iso 26000 putting social responsibility to work," in *International Trade Forum*, no. 3. International Trade Centre, 2010, p. 28.

[14] T. P. Hughes, S. Carpenter, J. Rockström, M. Scheffer, and B. Walker, "Multiscale regime shifts and planetary boundaries," *Trends in ecology & evolution*, vol. 28, no. 7, pp. 389–395, 2013.

[15] B. Friedman, P. H. Kahn, and A. Borning, "Value sensitive design and information systems," *The handbook of information and computer ethics*, pp. 69–101, 2008.

[16] Omidyar Network, "Ethical Explorer," https://ethicalexplorer.org/, p. 1, 2020.

[17] RRI Tools Consortium, "Welcome to the RRI Toolkit," https://www.rri-tools.eu/-/ethical-os-toolkit, p. 1, 2019.

[18] "sdgimpactasses tool," https://sdgimpactassessmenttool.org/, accessed: 2021-1-19.

[19] A. Dunne and F. Raby, *Speculative everything: design, fiction, and social dreaming*. MIT press, 2013.

[20] Seyff et al., "Tailoring requirements negotiation to sustainability," in *26th IEEE International Requirements Engineering Conference, RE 2018, Banff, AB, Canada, August 20-24, 2018*. IEEE Computer Society, 2018, pp. 304–314. [Online]. Available: https://doi.org/10.1109/RE.2018.00038

[21] P. Lago, "Architecture design decision maps for software sustainability," in *41st Intl. Conference on Software Engineering: Software Engineering in Society (ICSE-SEIS)*. IEEE, 2019, pp. 61–64.

[22] F. Albertao, J. Xiao, C. Tian, Y. Lu, K. Q. Zhang, and C. Liu, "Measuring the sustainability performance of software projects," in *2010 IEEE 7th International Conference on E-Business Engineering*. IEEE, 2010, pp. 369–373.

[23] N. Condori-Fernandez and P. Lago, "Characterizing the contribution of quality requirements to software sustainability," *Journal of systems and software*, vol. 137, pp. 289–305, 2018.

[24] Penzenstadler et al., "Iterative sustainability impact assessment: When to propose?" under review.