Human Factors in Software and Systems Engineering, Vol. 140, 2024, 76–85
https://doi.org/10.54941/ahfe1005040

Green Software Engineering Practices: Familiarity, Skills and Understandability Among Mid-Bachelor ICT Students

Kimmo Tarkkanen

Faculty of Engineering and Business, Turku University of Applied Sciences, Turku, Finland

ABSTRACT

The need for greener software engineering is apparent due to the increase in energy consumption and carbon emissions in the ICT sector, which pose significant challenges to environmental sustainability. By integrating green practices into the skillset of ICT students, they can contribute to sustainable development in the field. To understand what green software engineering skills mid-bachelor level ICT students already possess, and what they lack, this empirical research presented 154 green software engineering practices to 40 participants in a survey, which asked about their familiarity, ability to implement (skills), and understanding of each practice. Results implicate weakest knowledge and skills in server-side, system- and technology-specific practices, which are further described with vague acronyms that cause ambiguity. Reflections on contributions to the local context and implications for further research are discussed.

Keywords: Sustainability, Green ICT, Software engineering, Education

INTRODUCTION

In the era of digital transformation, the Information Technology (IT) sector plays a pivotal role in driving global innovation and economic growth. However, this rapid advancement has also led to an increase in energy consumption and carbon emissions, posing significant challenges to environmental sustainability. As a result, there is a growing need for green software engineering - a discipline that focuses on designing, developing, and maintaining software systems with minimal environmental impact.

In the realm of green software engineering, certain practices are more important, some are more popular and some easier to implement than others. The optimization of algorithms and code can lead to reduced CPU usage and energy consumption. Emphasizing software longevity and backward compatibility can mitigate electronic waste by reducing the demand for new hardware. Many other practices for sustainability and software exists, but developers lack knowledge and tools that would help them coding and maintaining energy-efficient software (Pinto & Castor, 2017). Interestingly, even the Guide to Software Engineering Body of Knowledge (SWEBOK) have been found to forget sustainability topic (Gibson et al., 2017) and vice versa, the sustainability topic researchers have been found to miss areas of SWEBOK (Pinto & Castor, 2017).

As the future professionals of the IT industry, students will be at the forefront of creating software solutions that are not only efficient and effective but also environmentally friendly. By integrating green practices into students' skillset, they can contribute to reducing the carbon footprint of the IT sector and promote sustainable development.

Understanding what green software engineering skills IT students already possess, and what they don't, is beneficial for couple reasons. Firstly, it provides insights into the current state of green software engineering education, highlighting areas of strength and identifying gaps that need to be addressed. Secondly, it helps in designing targeted educational interventions to enhance students' green software development skills. Indirectly, it aids in shaping the future of the IT industry, ensuring that sustainability becomes an integral part of software development processes.

This article delves into an empirical study conducted among ICT students. The research question is: With which green software engineering practices are the students least familiar and skilled, and which ones do they not understand well? A total of 40 participants, who had an average of 125 study credits, were presented with a comprehensive list of 154 green software development practices extracted from three easily accessible web sources. Participants were asked to evaluate each practice based on their familiarity, their ability to implement it (skills), and their understanding of it (clarity), providing responses in a simple dichotomic scale. Practices describe themselves and include both sustainability and green (e.g. only environmental) dimensions and this article uses the terms as synonyms.

The data gathered from this study offers a perspective into the current state of green software engineering education in a one higher education unit in Finland. By analyzing the responses, we can identify the practices that are well-known and understood, those that are known but not implemented, and those that are simply unclear to the students. This information can shape the future curriculum and training programs, ensuring they are tailored to address the identified gaps and enhance the students' skills in green software engineering.

GREEN SOFTWARE ENGINEERING PRACTICES IN EDUCATION

The sustainability topic in software engineering education was under-represented in the curricula in 2017 (Torre et al., 2017). The focus of teaching was on energy efficiency delivered through a fact-based approach, and sustainability topic suffered from lack of awareness, teaching materials and suitable technologies as well as the high effort required to teach it (ibid.). According to Penzenstadler & Fleischmann (2011) challenges are to motivate and interest students (and lecturers) for sustainability, to identify spheres of activity for software engineers, to build up competence fields for solutions, and to incorporate sustainability into the syllabus.

Possibilities to integrate sustainability to higher education were presented in a matrix by Rusinko (2010): The integration of sustainability is made to existing or new structures (such as courses and programs), and it focuses either on discipline-specific or cross-disciplinary practices. For integrating sustainability into software education, Penzenstadler & Fleischmann (2011) used three major steps: introductory seminars, building up with a lecture series, and integrating with general lectures on software engineering after teach-the-teacher seminars. Since, full student programs have been developed (see Lago, 2014), for example Erasmus Mundus Joint Master's Degree in Pervasive Computing and Communications for Sustainable Development (PERCCOM) that contains many courses that spread to several countries (Porras et al., 2016).

To date, the number of publications on the intersection of sustainability and computing and the training implementations reported in them has continued to grow and evolve (cf. Peters et al., 2023; Venters et al., 2023). For example, Alotaibi (2021) investigated the extent of awareness about sustainability among students and how their study programs and course contents incorporate the topic. The results show that 71% of students are unaware of the term sustainable development, and only 12% of them have an idea of how sustainable development is related to the software development (Alotaibi, 2021).

Heldal et al. (2024) examines competencies and skills that organizations in IT industry need to achieve their sustainability goals. Organizations have promoted external and in-house training courses to integrate sustainability to their software development processes and to translate environmental and social benefits into economic ones. Their study discusses and reveals more general sustainability skills needed than pointing the IT specific practices and actions that organizations in industry possess or don't, as they conclude that "*relevant topics are knowledge and skills on core sustainability concepts, system thinking, soft skills, building the business case for sustainability, sustainability impact and measurements, values and ethics, standards and legal aspects, and advocacy and lobbying.*"

Indeed, intended learning objectives can be divided into topic-specific competencies, cross-cutting competencies, awareness, and practical experience gains (Peters et al., 2023). Topic-specific competencies are either computing- or sustainability-specific, or their combinations. Cross-cutting competencies can be called key sustainability competencies, like communication and creativity, which are useful across different domains. This comprehensive, systematic literature review by Peters et al. (2023) examined how sustainability is being taught in computing education and concluded that research and educational implementations lack radical systemic change and innovativeness in thinking about sustainability education in computing. Moreover, they found that most research articles on the topic are experience reports with limited empirical research (Peters et al., 2023).

Is it the lack of empirical research (Peters et al., 2023), or the fact that the literature only occasionally refers to real everyday practices, to the actual teaching content at the intersection of sustainability and computing? In other words, to those green software engineering practices and everyday actions that constitute the energy-efficient software or allow for the reduction of GHG emissions in the daily work of computing professionals. While these studies may present the fact-based approach (cf. Torre et al., 2017), which is no longer seen as the best method, Venters et al. (2023, p. 8) articulate a pertinent question when considering trends in sustainability education and skills: "*It is back to the old question of how to strike a balance between* *generalist (knowledge of computing and sustainability) and specialist (e.g., software architecture and sustainable software infrastructure).*"

DATA COLLECTION AND ANALYSIS

Green software development practices were collected from three sources, namely from the websites https://sustainablewww.org[1], https://medium.com[2], and https://www.greenit.fr[3]. These webpages and their lists of practices are referred as sources 1, 2, and 3 respectively. Sources were visited between September and December 2022 and in a respective order 17, 22, and 115 green software and web development practices were extracted.

The selection of these specific sources was based on their free availability, easy accessibility, and high ranking in the search engine results. Such characteristics of sources aim to simulate the situation in which a junior software developer seeks information about how to develop sustainable software and ends up visiting these webpages for information. Moreover, the aim was to find lists that include as practical guidance as possible that could directly be applied to the software or web development project. For example, "Vanilla JavaScript - Write clean vanilla JavaScript instead of adding weight by using JQuery, Typescript etc." (Source 1); "Don't conceive an API, CRM or third-party-services-centric interface" (Source 2); "Disable Apache's DNS Lookup" (Source 3).

The practices were added to an online survey containing 154 practices in total. Each practice was put into a matrix, which contained three statements: 1) I already knew this practice is ecological 2) I already have skills to implement this practice in my software project, and 3) The practice is unclear. Participants answered each statement on a dichotomic scale 'Yes' or 'No', resulting 462 data points for each participant. In addition, only the number of study credits was compulsory information, while their name and the current software project were optional background information.

The data was collected between December 2022 and May 2023 from the students at Turku University of Applied Sciences, Finland. Participants were recruited from the students enrolled in the course Software Engineering and Modelling. Participants were 2nd year students representing two different study programs Engineers in ICT and Business IT. Currently, sustainability related courses are not in their curriculum, however, organizational policy encourages teachers to include the topic in their courses. Total number of participants taking the survey was 55, yet 15 participants answers were dismissed for this analysis due to incompleteness in their data (N = 40).

The data analysis of this article leans mostly on descriptive statistics and focuses on the questions that have direct practical benefits in the higher education domain. Thus, the question for the data analysis was the research question: With which green software engineering practices are the students

[1]The actual list of items has been removed, yet the site's blogs and other resources include similar content and as easy to reach.

[2]See article by Debomy (2020).

[3]See Bordage (2019). English version is available at http://www.ecometer.org/rules/ and more recently in https://github.com/cnumr/best-practices.

the most and the least familiar and skilled, and which they do not understand well? In addition, interest was raised in whether the number of study credits influences student's familiarity and skills of green practices, and whether there are differences in student's familiarity and skills between the sources.

RESULTS

Participants' study credits were 125 on average, standard deviation 15.4 and margin of error of the mean 4.9, and confidence interval 120 to 130 study credits. Thus, participants represent highly homogenous group of 2nd year ICT students in a higher education institution in Finland.

Over 30 % (48 out of 154) of practices did not get any 'unclear' answers. The unclarity was not high, of all practices the average unclarity rate was 5 %. The most unclear practice was "98. Server: Use a CDN" (Source 3), which was answered unclear by 25% of participants. Indeed, five most unclear practices contained acronym, such as PWA, CMS, and HSTS, which indicate also the reason for unclarity.

Figure 1 shows the least skilled practices among the participants. Those include mostly 1) server-side operations, such as "Use all the CMS' levels of caching", and 2) system- or technology-specific operations, such as "Disable Apache's AllowOverride directive", as well as 3) above mentioned acronyms.

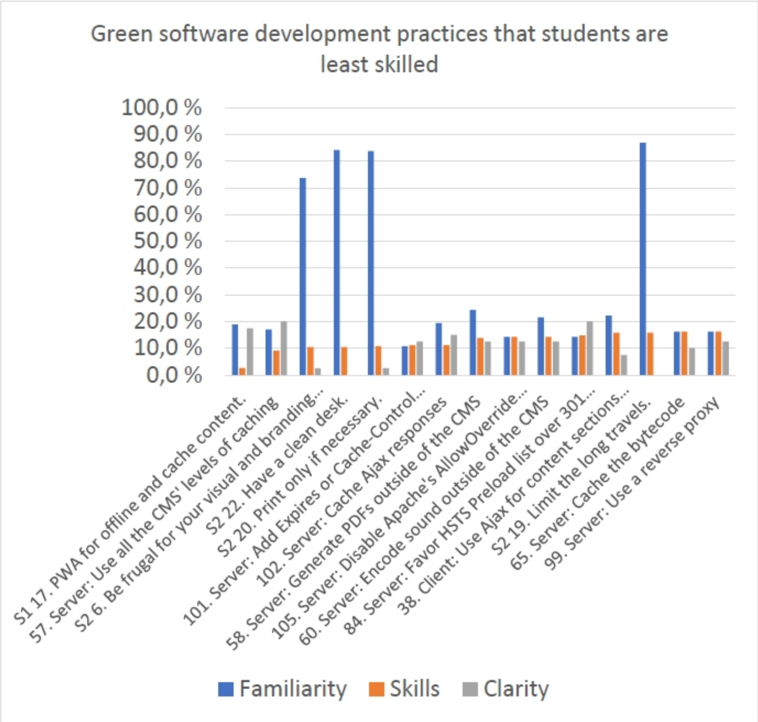

Figure 1: Most unskilled green ICT practices among participating students. S1 and S2 refer to the source of practice (others are S3).

Figure 2 shows the least familiar practices to participants. For some reason, the most unfamiliar practice, with 0% (nobody knew), was "Check your website's environmental impact", although over 25% of participants answered they have skills to implement the practice. Most probably, students have not heard about website impact assessment tools, such as Google Lighthouse or ecograder.com, that are fairly new tools, yet they believe they have skills to use these tools in their software projects. Naturally, familiarity gets overall higher scores than skills.

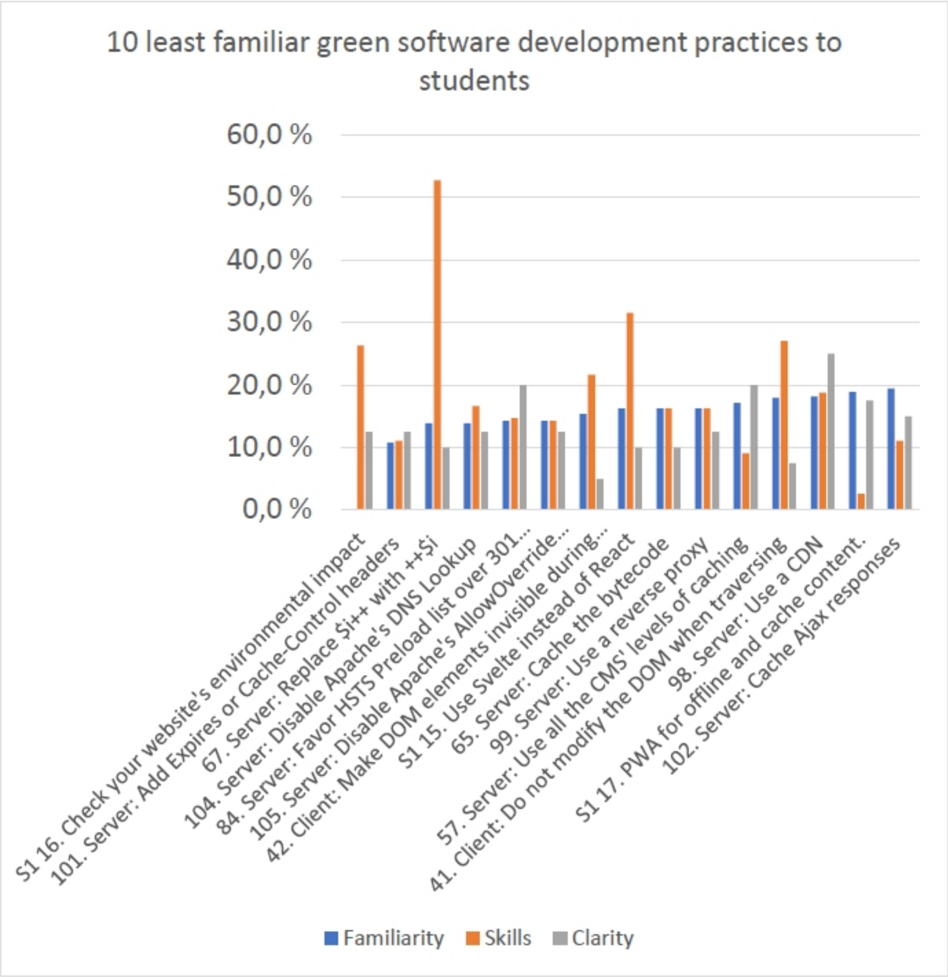

Figure 2: Most unfamiliar green ICT practices among participating students. S1 and S2 refer to the source of practice, and unless mentioned the source is 3.

Figure 3 shows practices from the Source 2 divided into groups by their corresponding job title. The interpretation is that the participants considered themselves most skilled in Web Developers' green practices while UI designers' green practices were the most familiar, yet students were most unskilled with them.

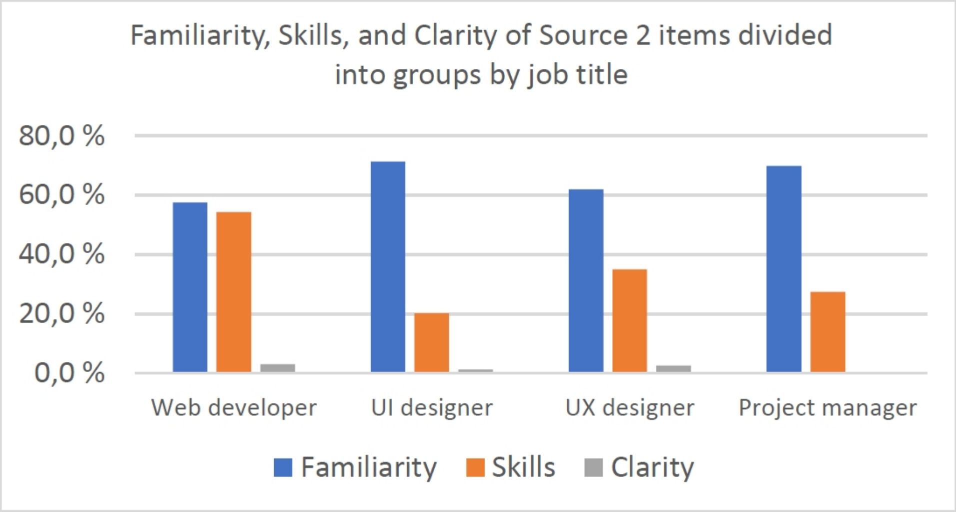

Figure 3: Familiarity, skills, and clarity of source 2 practices divided into groups by job title.

Overall, the number of study credits of participants was not connected with familiarity and skills of practices. The only correlation between number of credits and familiarity and skills points was with the practices in the Source 2 (medium.com). The correlation coefficient calculated from the sample (N = 40) was 0.4706 and p-value (2-tailed) 0.002, which means statistically significant correlation ($p<0.05$) between credits and skills (Figure 4). Explanatory factor is 0.22, which means that 22% of the variance of the scores in skills in Source 2 can be explained by the number of study points. For the familiarity points, the correlation coefficient was 0.348 and p-value 0.028 ($p<0.05$).

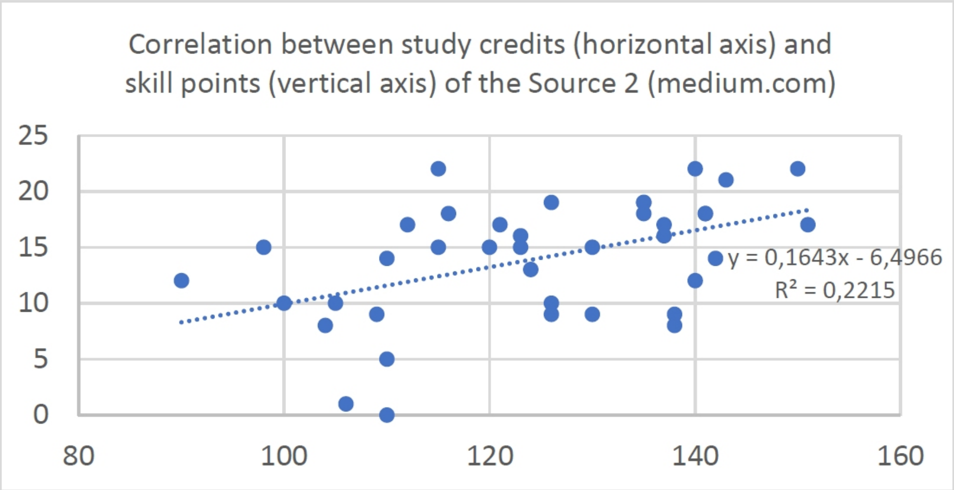

Figure 4: Relationship of study credits and skill points of source 2.

Considering the question whether there were differences between sources i.e. from which sources the practices were most familiar and skilled, the different number of practices caused the points to be harmonized with a weighting factor. Then, null hypothesis was that the mean values of practice familiarity and skills are the same between sources. That hypothesis is falsified when comparing the means between the different sources by ANOVA single factor analysis, which shows statistical significance ($p = 0.002$). Average (weighted) points indicate that source 2 practices were most familiar and skilled (Table 1). Comparing the means between individual pairs of sources shows that Source 2 (medium.com) has statistically significant difference with both Source 1 (sustainablewww.org) ($p = 0.01$) and Source 3 (greenit.fr) ($p = 0.002$), and the difference stays with both familiarity and skills.

Table 1. Source 2 medium.com got highest points in familiarity and skills.

	Source 1 (sustainableww.org)	Source 2 (medium.com)	Source 3 (greenit.fr)
Average skills + familiarity points (weighted)	153,77	190,23	137,83
St. deviation	49,21	74,91	70,05
Margin of error	15,74	23,96	22,40

DISCUSSION AND CONCLUSION

Research literature suggests that learning sustainability is facilitated when learners possess topic-specific, cross-disciplinary soft skills, such as systemic and critical thinking. However, in the ICT field proficiency in tasks like coding (for efficiency and processing speed), designing software architectures (for simplicity), or creating appealing interfaces (with multimedia considerations) is also essential. Sometimes sustainability training seems similar with training a bike riding only with good coordination and communication skills, rather than pedalling (fast enough), steering (not too steeply) or tilting (carefully) in practice. On the other hand, biking nor sustainability skills do not always need specific learning of the details of the actions and operations but some are learnt inherently within ICT education. The question arises what these practices are that ICT students may already feel familiar with, that they have skills and understanding of. This research aimed at studying how well mid-bachelor ICT students know, can implement, and understand sustainable software engineering practices extracted from three different web sources.

Results implicate that their understanding, familiarity, and skills are weakest in server-side, system- and technology-specific practices, which, in the survey, were introduced with vague acronyms. In addition, an interesting difference was found in the green practices of a UI designer, which were best known, but the least skilled among mid-bachelor students. The group of participating students were homogenous and represented well 2nd year students with a low variance of study credits in one applied university to which context the results can be generalized. The number of study credits and scores correlated only with practices of the source 2. Of the studied sources, sustainability practices in source 2 got highest scores, yet there may be several reasons. Firstly, the survey containing source 2 practices were answered always last, which can bias especially the familiarity result due to parallel practices introduced before in other two sources. Second, source 2 contains practices that were described in the survey with about 100 words, while for example source 3 practices could contain only few words in the survey (although it has more comprehensive explanations elsewhere). Due to such survey-technical reason the result may be biased. Third, practices in source 2 covered not only IT-specific but also more general practices than other sources, such as printing or traveling related practices, which may have resulted in higher scores in familiarity or skills. Moreover, weighting points between surveys that are of different sizes may be a source of bias in results.

The results indicate the need for further empirical research into the green software engineering practices that students either possess or lack. In the future research studies, more balanced sources and practices, and appropriate group sizes should be considered, together with more comprehensive statistical analyses, for example, of the relation between skills and familiarity, individual practices and groups of practices. Moreover, future studies need to re-consider the ambiguity factor, e.g. how practices are presented, which may have had effect on answers about the other two, skills and familiarity dimensions. However, results lay down the baseline for future research within the same university and same study programs, and thus, allows critical thinking to foster sustainability in future curriculum, course content and practical assignments. For example, system- and technology-specific practices that were least familiar to students can be turned to the corresponding terminology and operations in locally used technologies and systems more familiar to students and teachers. Similarly, we can try to ensure that well-known UI practices will be deployed and learnt in the later studies.

Ultimately, we want our ICT students to become professionals, who are competent at practicing sustainable operations in the field. Next, both students and education institutions, as well as employing companies need to prove it, that we are not only talking about sustainability but can act on it – be it materialized as organizational or personal certificates, diplomas and badges that one needs to present in a job interview or purchasing contract – and for that purpose we need more empirical research studies that lay foundation on skills and practices included.

ACKNOWLEDGMENT

The author would like to acknowledge people and partners in EU/ERDF-funded MitViDi-project (Green Metrics for Public Digitalization Acquisitions).

REFERENCES

Alotaibi, Y. (2021). Investigating the higher education curriculum for sustainable software development. International Journal of Computing and Digital Systems, 10, 605–611.

Bordage, F. (2019). Ecoconception web : les 115 bonnes pratiques, 3e edition. Eyrolles. In English: Web ecodesign: 115 good practices, 3rd edition, published in April 2019. Retrieved from https://greenit.fr.

Debomy, P. (2020). How to make a Green IT/Low-carbon Website? Retrieved from the website: https://pasomy.medium.com/how-to-make-a-green-it-low-carbon-website-8062f550a1f7.

Gallagher, K., Creswell, R., Lambert, B., Robinson, M., Lei, C. L., Mirams, G. R., Gavaghan, D. J. (2024). Ten simple rules for teaching sustainable software engineering. arXiv preprint arXiv:2402.04722.

Heldal, R., Nguyen, N. T., Moreira, A., Lago, P., Duboc, L., Betz, S., Coroamă, V. C., Penzenstadler, B., Porras, J., Capilla, R., Brooks, I., Oyedeji, S., Venters, C. C. (2024). Sustainability competencies and skills in software engineering: An industry perspective. Journal of Systems and Software, Vol. 211, 2024.

Lago, P. (2014). A master program on engineering energy-aware software. In EnviroInfo, 469–476.

Penzenstadler, B., Fleischmann, A. (2011). Teach sustainability in software engineering? In 2011 24th IEEE-CS Conference on Software Engineering Education and Training (CSEE&T) (pp. 454–458). IEEE.

Peters, A. K., Capilla, R., Coroamă, V. C., Heldal, R., Lago, P., Leifler, O.,... & Venters, C. C. (2023). Sustainability in computing education: A systematic literature review. ACM Transactions on Computing Education.

Pinto, G., Castor, F. (2017). Energy efficiency: A new concern for application software developers. Communications of the ACM, 60(12), 68–75.

Porras, J., Seffah, A., Rondeau, E., Andersson, K., Klimova, A. (2016). PERCCOM: A master program in pervasive computing and communications for sustainable development. In: Proceeding 2016 IEEE 29th Conference Software Engineering Education and Training, CSEEandT 2016, pp. 204–212 (2016).

Rusinko, C. A. (2010). Integrating sustainability in higher education: A generic matrix. International Journal of sustainability in higher education, 11(3), 250–259.

Torre, D., Procaccianti, G., Fucci, D., Lutovac, S., Scanniello, G. (2017). On the Presence of Green and Sustainable Software Engineering in Higher Education Curricula, 2017 IEEE/ACM 1st International Workshop on Software Engineering Curricula for Millennials (SECM), Buenos Aires, Argentina, 2017, pp. 54–60, doi: 10.1109/SECM.2017.4.

Venters, C. C., Capilla, R., Nakagawa, E. Y., Betz, S., Penzenstadler, B., Crick, T., Brooks, I. (2023). Sustainable software engineering: Reflections on advances in research and practice. Information and Software Technology 164, 107316.