Design and implementation of a lecture for teaching current Green Coding approaches and practices at the HTW Berlin

Dennis Junger [1], Volker Wohlgemuth [1]

Abstract: At the Industrial Environmental Informatics Unit (HTW Berlin), a test integration for a future 'Green Coding' course was developed to implement the concept of environmentally conscious and energy-efficient software development. The course titled "Current Development Trends in Environmental Informatics" now educates students on fundamental concepts and techniques in this field. The course emphasizes the development of software that is both energy efficient and environmentally friendly. Students learn how to optimize software applications' energy efficiency by designing energy-efficient algorithms and enhancing server and data center performance. The course covers essential principles of 'Green IT' and explores the 'Green Coding' sub-field. The course is divided into three modular parts: internal lectures, external lectures, and a practical part to manifest the theoretical knowledge. The content of the internal lectures was significantly influenced by the current state of research regarding 'Green IT' at the Industrial Environmental Informatics Unit of the HTW Berlin, in which current and past master theses, papers, and publications were presented. Current known German 'Green Coding' researchers were invited for external lectures, partly on-site and online. A practical measurement course lasting several days, which provided the students with several test setups according to the "Blue Angel for software products" standard to pursue their research questions, was created to deepen their knowledge. The following article presents and discusses the topics and results of these short papers and evaluations of the course concept and the course itself.

Keywords: Green Coding, Potentials of Green Coding, Curricula, Software Engineering, ISOC Foundation, Trier University of Applied Sciences, HTW Berlin, University of Applied Sciences

1 Introduction: Motivation and Relevant Terminology

Sustainability as relating to information and communication technology (ICT) generally and software development/engineering more specifically is a research field of growing interest. While the focus of sustainability considerations in ICT, for the longest time, was on hardware, software behavior is the driver of hardware energy consumption and additionally affects the hardware life-cycle e. g. as ever more demanding software systems quickly outgrow older hardware [Wi95]. To enable sustainable software production, access to models, methods, and tools geared toward sustainability is elementary. Furthermore, it is essential to make future IT professionals and researchers aware of the sustainability aspects of ICT and equip them with the skills to consider them in their work adequately. The terms 'Green IT' and 'Green By IT' separate the field into two main areas, 'Green By IT' designates ICT that aims to have a positive impact on sustainability in other areas, whereas 'Green IT' deals with making ICT itself more sustainable [Uh12]. 'Green Coding' is a part of 'Green IT'. 'Green

[1] HTW Berlin, University of Applied Sciences, Industrial Environmental Informatics Unit, Wilhelminenhofstr.75A, 12459 Berlin, Germany dennis.junger@htw-berlin.de, volker.wohlgemuth@htw-berlin.de

IT' focuses on all the aspects of reducing emissions of ICT and focuses on the methods and tools that can be leveraged during development to impact software sustainability in all of its life-cycle phases (development, usage, and disposal phase). The authors expected the conception and evaluation of the 'green coding' course to increase the understanding of green coding and thus to make the future generation of computer scientists more aware of green coding. The current state of the art will be presented in the following article. Subsequently, the conception of the course, divided into the selected didactic concept and the grading concept, as well as the structure of the theoretical and practical parts, are presented to be able to classify and evaluate them in the following evaluation part.

2 Background and Related Work

2.1 Current and Suggested Green Coding Practices for Instruction

To determine the course's relevant 'Green Coding' content, we need to explore existing concepts of environmentally friendly software development and its impact on energy consumption. In addition to models that make the software development cycle itself green, for example, using a sustainability report such as the one designed by the UCB in the 'GREENSOFT model' [Na11, Di13], it is not only the engineering process itself that is important but also the methods for technically measuring and improving energy consumption. These methods and models are considered suitable for providing software developers with insights into their code's potential for energy optimization. ([Ke13, Hö13]). A recurring theme in numerous studies is the absence of a universally optimal solution regarding energy and resource consumption[Pe17]. This demonstrates the complexity of software sustainability and highlights why IT professionals need to be able to assess which measures are suited for a specific project. However, the exact steps necessary to improve upon a given piece of source code's induced energy consumption are not as clear. This is because every software and every algorithm has different solutions and third-party code such as libraries, frameworks, and Software Development Kits (SDK) are only sometimes known or disclosed. Some work provides concrete hints [Hö14] or details where the energy hot spots occur in the source code [Wa12, Ve18]. To develop a general approach to sensitize the students, it is essential to understand the basics and to get a feeling for the impact of software on the environment. Topics like the impact of hardware and software on the desktop device or in the cloud, for example, data centers, are reasonable. Unfortunately, the sending of data through networks can be taught poorly, so this needs to be addressed. Further steps are to learn the basics of software energy consumption measurement so that students can apply this in future implementations. It has been observed that integrating green software into the computer science curriculum presents a notable difficulty, as engineers often need more knowledge, expertise, and tools to create environmentally sustainable software. According to Saraiva [Sa21], recent studies have shown that just these current tools should be taught. To address this, current projects were also presented. Recent projects and foundations, such as the

green web foundation[2], Sustainable Digital Infrastructure Alliance (SDIA)[3] and the Green Software Foundation (GSF)[4], which are each concerned with individual approaches to solutions, should be presented so that students learn explicit solution strategies and message groups in addition to the holistic theoretical approach. Furthermore, the first attempts to implement online software courses like this one of the GSF are presented[5]. Besides the parallel literature analysis for the research project "Potentials of Green Coding"[Ju23], which summarizes state of the art as of 2010, similar conclusions could be drawn. Deciding which content is relevant for providing a reasonable basis is difficult. It is also a challenge to implement curricula and discussions as an initial step [Ma11]. In 2010 [Ca10] invented a hypothetical agenda for integrating computing education for sustainability. The focus of [Ca10] is on waste management, eco-labeling, energy management, carbon management, reuse, environmental reporting, and open-source software. Most of these courses are already lectured in the HTW Masters program of the Industrial Environmental Informatics Unit[6], where the pilot course will be part.

2.2 Integration of Green Coding Concepts into Existing Study Program Curricula

It is critical to greening ICT to give future IT professionals the skills needed to produce sustainable software. The focus of research question 3 of the "Potentials of Green Coding" project is to find possibilities for implementing the approaches into the education of future IT professionals. As most software developers have an academic background, this review focuses on implementing aspects of 'Green IT' into the curricula of higher education institutions. The basis for this idea is a statistic from 2022 [St22], which describes the levels of formal education for software developers worldwide in 2022, according to which 75,19 % of the software developers have a higher education background.

2.3 Current Implementation of Green Coding Modules and Courses

To address this question, the initial approach involved researching the courses and programs offered by higher education institutions. However, the search turned out to be more difficult than anticipated. Many universities provide specialization details for their study subjects in module handbooks, while others only make course information and disciplines accessible through an application management system after creating an account. These factors influenced our search, which primarily relied on online sources. In addition to sources from cooperating universities, we consulted job platforms such as LinkedIn[7] and

[2] https://www.thegreenwebfoundation.org/ [2023-05-22]
[3] https://sdialliance.org/ [2023-05-22]
[4] https://greensoftware.foundation/ [2023-05-22]
[5] https://learn.greensoftware.foundation/ [2023-05-22]
[6] https://bui.htw-berlin.de/ [2023-05-22]
[7] https://www.linkedin.com/ [2023-05-16]

institutions known in the 'Green Coding' community. Currently, courses designed for Green Software Engineering or 'Green Coding' are mainly available in Europe. We found the following universities and universities of applied sciences in Europe that teach related courses.

- England - University of Bristol
- Finland - LUT University
- Germany - Hasso Plattner Institute (HPI)
- Germany - University of Applied Sciences HTW Berlin
- Germany - Umwelt-Campus Birkenfeld (UCB)
- Netherlands - Amsterdam School of Data Science
- Netherlands - DELFT University of Technology
- Sweden - Mälardalen University
- Sweden - Royal Institute of Technology (KTH)
- Sweden - University of Gothenburg
- Ukraine - Kharkiv Aviation Institute (current status unknown)

The following two universities were identified within the United States of America:

- Carnegie Mellon University
- The University of California Berkeley

2.4 Creating a Green Coding Curriculum

For future curricula, Drake and Reid [Dr18] propose to, on the one hand, teach the big picture of a topic and, on the other hand, the main competencies. To convey this knowledge, technical, practical, and participatory interests should be addressed [Fr06]. Mishra and Mishra [Mi21] describe how a sustainable software engineering curriculum based on ACM/IEEE could be created. Since it is challenging to design entire curricula such as the PERCCOM [Ko19] Masters program, the first step should be to develop individual courses, workshops, discussion groups [Ma11] or university initiatives like in the HAW Hamburg [Ei23]. This can be followed by a module like "Software Engineering Sustainability", introduced at the 'Kharkiv Aviation Institute' [Tu19].

3 Conception of a Green Coding Course at HTW Berlin

In the context of the third research question, which was led by HTW Berlin, the conception of a module in the Master's program "Industrial Environmental Informatics" took place as

a pilot project. A test integration at HTW Berlin implemented the concept. The teaching subject 'Current Development Trends in Environmental Informatics'[8] now introduces students to the basic concepts and techniques of environmentally conscious and energy-efficient software development. The course focuses on teaching students how to develop energy-efficient and environmentally friendly software by optimizing the energy efficiency of software applications by, for example, writing energy-efficient algorithms or optimizing the performance of servers and data centers. For this purpose, the basic concepts of 'Green IT' are taught, and the Green Coding subfield is addressed. The course is divided into three modular parts: internal lectures, external lectures, and a practical part to manifest the theoretical knowledge. Further grading was divided into a theoretical and a practical part. The academic content was examined through oral examinations focusing on 'Green IT', 'Green Computing', and 'Green Coding'.

3.1 Didactic Concept

A well-designed didactic concept aims to optimize the learning experience by promoting active engagement, fostering critical thinking, and facilitating the acquisition and application of knowledge and skills. For the course design, it was important that the learning objectives be achieved through theoretical and practical exercises. The students learn about selected current technologies that are relevant for the development of 'Environmental Management Information Systems' (EMIS) (e.g. 'mobile computing', 'Green IT'). The students learn to assess current technologies regarding their relevance to industrial environmental informatics. The course content should reflect the most relevant principles identified by previous research in the "Potentials of Greencoding" project by providing relevant literature, multimedia materials, technologies, tools and online resources, primarily via the learning platform Moodle. Feedback through sub-tasks and collaboration in the practical part will deepen the learning experience and guide learning. The practical units should activate the students and prepare them for later tasks. To check the effectiveness of the didactic concept, there will be evaluations and revisions, which will assess the course itself as well as the knowledge level of the students and the perceived knowledge level by means of anonymous surveys and evaluations to present the outcome as a key figure.

3.2 Grading System

To ensure a comprehensive and differentiated grading system that accurately reflects the skills acquired by participating lecturers, a multi-level grading system was implemented. As a prerequisite for final exam admission, students were required to evaluate the impact of ICT hardware. In this context, a lecturer of the Öko-Institut e. V. introduced the digital

[8] https://bui.htw-berlin.de/fileadmin/HTW/Zentral/Rechtsstelle/Amtliche_Mitteilungsblaetter/2013/30_13.pdf [23.06.2023]

carbon footprint online calculator with a life cycle cost module where the students could contribute to a new functionality based on their theoretical research.

3.3 Theoretical Course Content - Presentations and Talks

The content of the internal lectures was significantly influenced by the current state of research regarding 'Green IT' at the Industrial Environmental Informatics Unit of the HTW Berlin, in which current and past master theses, papers, and publications were presented. External researchers gave lectures, partly on-site and online. For example, experts from the Öko-Institut e. V. gave lectures on the life cycle costs of ICT, the digital footprint of ICT, and the "Blue Angel for software products". The lectures of the LMU Munich[9] presented the basics of 'Green IT' in data centers and Green Coding Berlin GmbH[10] showed the current state of the art in the software industry. To learn more about the topic of implementation, the 'Greensoft Model' [Na11] was presented by the participating researchers of the UCB, with a focus on 'Green Software' and 'Green Software Engineering'.

3.4 Practical Course Content - Measurement Laboratory

To deepen the knowledge and gain experience with the "Blue Angel for software products" and the underlying methodologies and measurement methods, a measurement lab was designed. The following describes the design and implementation of the measurement lab.

3.4.1 Set-up and Preparation of the Measurement Laboratory

In the context of previous research at the HTW Berlin in 2022 [Ju22], a test stand with a system under tests was modeled on the example of the "Blue Angel for software products". Creating a second test stand was essential to enable the students to run the measurements smoothly and in parallel. To be able to make measurements with an increased GPU share, an additional computer and power meter were prepared for a second measuring station with an extended graphical processing unit. To diverse student needs and enhance efficiency in the lab, the standard image was equipped with additional automation software called 'WinAutomate' and 'Actiona', alongside the existing 'Power Automate'. A standardized workflow was developed, based on the measurement type described in the earlier research in 2022 [Ju22], incorporating fixed measurement duration and cooldown periods. In the measurements carried out later, this was 120 seconds of automation followed by 60 seconds of cooldown time for fitting it to the planned measurement practical time. This was repeated for statistical means 30 times according to the Blue Angel [Na21] model.

[9] https://www.lmu.de/de [2023-05-22]
[10] https://www.green-coding.berlin/ [2023-05-22]

3.4.2 Measurement Laboratory and Its Results

The results of the measurement lab are presented below. The following topics were part of the laboratory and targeting to figure out differences in resource and energy usage of (proprietary and open source) applications in different fields, like image- (GIMP & Photoshop) and sound processing (Spotify & VLC Media Player) and statistical analyses (Matlab & R-Studio [Se23]). Also, programming technology topics, such as the differences between programming languages (C++ & Python) and (single- & multi-)threading technologies in GO[11] have been studied. The gathered results were presented to the entire course as lessons learned and written as a 10-page report.

3.5 Evaluation

To conduct a targeted evaluation, the participant's level of knowledge at the beginning and the end of the module needed to be captured and compared. For this reason, an anonymous Moodle survey was created for students that had been filled out by the students at the beginning and the end of the semester. The curve shown in figure 1 shows the survey results before and after course completion. The survey was designed in such a way that participants could choose between "Strongly Agree (1)", "Agree (2)", "Disagree (3)", and "Strongly Disagree (4)". A mean value was calculated before and after the course to determine the degree to which the content was conveyed. This mean value has to be seen about the respective questions. The questions were asked so that for each question the counter "Strongly Agree (1)" corresponds to the desired goal and "Strongly Disagree (4)" corresponds to the undesirable output. A detailed list of the questions with the corresponding results as RAW Data and detailed questions can be found in the supplementary material at `https://doi.org/10.5281/zenodo.7970613`. In the shown Figure 1, we find the evaluation of the surveyed fields of 'Green IT' for the review. There is continuous progress since each question's average value has improved. As a limitation of the survey, it should be mentioned that the students who were registered for the second exam period but already had to hand in their paper, probably have not yet worked on learning the lecture notes since the second period took place about three weeks after the paper hand-in. This could slightly worsen the survey results of the individual progress. The students successfully learned the criteria and indicators to evaluate the environmental impact of software (Question 4), got ideas about how to produce software that is more energy efficient (Question 7), learned how to evaluate (Question 15) and compare (Question 18) software in terms of power and resources and also knew how to conclude energy consumption to the cause of the problem (Question 19). If you would like more details about the evaluation, follow the link to the above RAW data.

[11] `https://go.dev/` [2023-05-22]

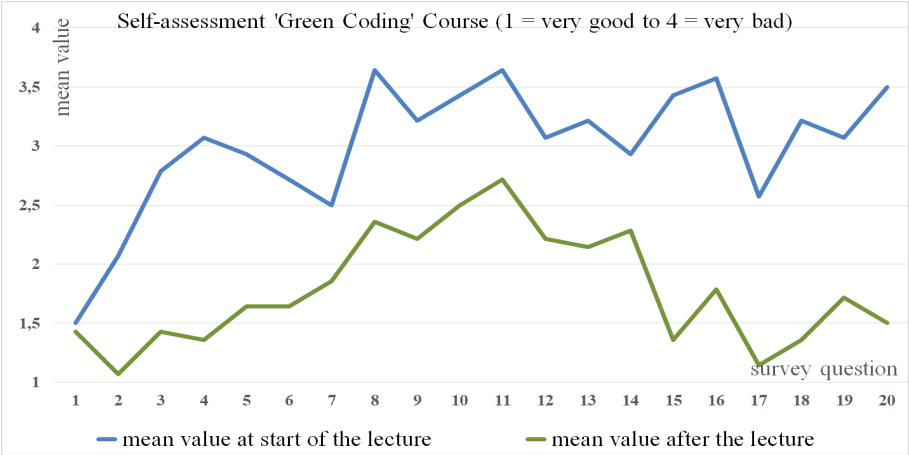

Fig. 1: Self-assessment 'Green Coding' - Pilot Integration

3.6 Challenges During the Lab and Solution Approaches

As described in the 3.2 grading section, a part of the final grade was designed for students to review each other's learning and the difficulties of the topic through lessons learned, which were also made available to the instructors. The most frequently mentioned feedback is listed below to identify and address weaknesses in future course designs. The measurements had to be repeated several times for the statistical mean. This resulted in long waiting and idle times during the scenarios. For the future, it's essential to define measurement blocks parallel to theoretical input so students can use their time more wisely. In addition, an open-door measurement lab would be good. Also to post-process the action, resource, and energy data and evaluate these data with 'OSCAR'[12] was very time-consuming and difficult. To solve this, 'OSCAR' and the measuring station should be reviewed or even extended to compare several measurements with the baseline. Besides data evaluation problems, there is also a problem with comparing scripts, due to no clear measurement instructions for this task. A universal approach could be researched and developed.

4 Conclusion and Outlook

It was possible to find the state of the art regarding the teaching of 'Green Coding' content. This was divided into teaching strategies and teachable topics, nonetheless supported by the previous quantitative literature analysis of the authors as project participants of "Potentials of Green Coding". The focus of 'Green Coding' could be identified and compared to current universities offering it. An example implementation with theoretical and practical

[12] `https://oscar.umwelt-campus.de/` [2023-05-16]

parts could be designed and implemented. The evaluation of the example showed mainly positive study results but also weaknesses in the organizational implementation of the lab part. Weaknesses in the current measurement methodology, mainly in the data post-processing, were identified. As a by-product of the course concept students could contribute to current 'Green Coding' research by means of papers that now can and should be published successively at conferences.

Acknowledgements

This work was funded by the Internet Society Foundation project "Potentials of Green Coding". We would also like to thank our partner, the German Informatics Society, the Institute for Software Systems at Environmental Campus Birkenfeld, especially Stefan Naumann, Achim Guldner, Max Westing, and Franziska Mai for talking and supporting the measurement lab. Also, we want to thank Arne Tarara from Greencoding Berlin GmbH, Jens Gröger from Öko-Institut e. V. and Maximilian Höb from LMU Munich for their talks about their 'Green Computing' and 'Green Coding' expertise.

Bibliography

[Ca10] Cai, Yu: Integrating sustainability into undergraduate computing education. In: Proceedings of the 41st ACM Technical Symposium on Computer Science Education. SIGCSE '10, pp. 524–528, 03 2010.

[Di13] Dick, Markus et al.: Green software engineering with agile methods. In: 2013 2nd International Workshop on Green and Sustainable Software (GREENS). pp. 78–85, 2013.

[Dr18] Drake, Susan et al.: Integrated Curriculum as an Effective Way to Teach 21st Century Capabilities. Asia Pacific Jorunal of Educational Research, 1:31–50, 01 2018.

[Ei23] Eickstädt, Elina et al.: Computer Science for Future - Sustainability and Climate Protection in the Computer Science Courses of the HAW Hamburg, 2023.

[Fr06] Fraser, Sharon P. et al.: The curriculum? That's just a unit outline, isn't it? Studies in Higher Education, 31(3):269–284, 2006.

[Hö13] Hönig, Timo et al.: Proactive Energy-Aware System Software Design with SEEP. Softwaretechnik-Trends, 33(2):6–7, 2013.

[Hö14] Hönig, Timo et al.: Proactive Energy-Aware Programming with PEEK. In: 2014 International Conference on Timely Results in Operating Systems. USENIX, 2014.

[Ju22] Junger, Dennis et al.: Conception and test of a measuring station for the analysis of the resource and energy consumption of material flow-oriented environmental management information systems (EMIS). In: EnviroInfo 2022. Gesellschaft für Informatik e.V., 09 2022.

[Ju23] Junger, Dennis et al.: Potentials of Green Coding - Findings and Recommendations for Industry, Education and Science, 2023. to be published.

[Ke13] Kern, Eva et al.: Green software and green software engineering–definitions, measurements, and quality aspects. In: First International Conference on Information and Communication Technologies for Sustainability (ICT4S2013), 2013b ETH Zurich. pp. 87–91, 2013.

[Ko19] Kor, Ah-Lian et al.: Education in Green ICT and Control of Smart Systems : A First Hand Experience from the International PERCCOM Masters Programme. IFAC-PapersOnLine, 52(9):1–8, 2019. 12th IFAC Symposium on Advances in Control Education ACE 2019.

[Ma11] Malik, Alam Sher et al.: Twelve tips for developing an integrated curriculum. Medical teacher, 33(2):99–104, 2011.

[Mi21] Mishra, Alok et al.: Sustainable Software Engineering: Curriculum Development Based on ACM/IEEE Guidelines. In: Software Sustainability. Springer, Cham, 2021.

[Na11] Naumann, Stefan et al.: The greensoft model: A reference model for green and sustainable software and its engineering. Sustainable Computing: Informatics and Systems, 1(4):294–304, 2011.

[Na21] Naumann, Stefan et al.: The eco-label blue angel for software - Development and components. In: Advances and New Trends in Environmental Informatics: Digital Twins for Sustainability. Springer, pp. 79–89, 2021.

[Pe17] Pereira, Rui et al.: Energy Efficiency across Programming Languages: How Do Energy, Time, and Memory Relate? In: SIGPLAN International Conference on Software Language Engineering. ACM, New York, USA, pp. 256–267, 2017.

[Sa21] Saraiva, João et al.: Bringing Green Software to Computer Science Curriculum: Perspectives from Researchers and Educators. In: ITICSE. ACM, 2021.

[Se23] Seegert, Tim et al.: Energy and resource comparison of current applications with a focus on statistical analyses and evaluations using the example of Matlab and R/RStudio, 2023. to be published.

[St22] Statista: Worldwide levels of formal education for software developers, 2022. `https://www.statista.com/statistics/793568` [2/24/2023].

[Tu19] Turkin, Igor et al.: Software Engineering Sustainability Education in Compliance with Industrial Standards and Green IT Concept. In: Green IT Engineering: Social, Business and Industrial Applications. Springer, pp. 579–604, 2019.

[Uh12] Uhl, Axel et al.: Beyond Green IT - Die Symbiose von IT und nachhaltiger Energie. In: Smart Energy. Springer, pp. 193–205, 09 2012.

[Ve18] Verdecchia, Roberto et al.: Code-Level Energy Hotspot Localization via Naive Spectrum Based Testing. In: Advances in Environmental Informatics: Managing Disruption, Big Data and Open Science. Springer, 2018.

[Wa12] Wang, Shinan et al.: Safari: Function-level power analysis using automatic instrumentation. In: 2012 International Conference on Energy Aware Computing. IEEE, dec 2012.

[Wi95] Wirth, Niklaus: A Plea for Lean Software. Computer, 28(02):64–68, 1995.