





# Proposal for the integration of Sustainability in the Systems Engineering Program through Requirements Engineering

Luis Romero-Untiveros<sup>1</sup>; Juan Lara-Herrera<sup>2</sup>; Katherine Paredes-Guerrero<sup>3</sup>; Enrique Chiroque-Zanabria<sup>4</sup>  
<sup>1,2,3,4</sup> Universidad de Ciencias y Humanidades, Peru, [luromero@uch.edu.pe](mailto:luromero@uch.edu.pe), [jlara@uch.edu.pe](mailto:jlara@uch.edu.pe), [kparedes@uch.edu.pe](mailto:kparedes@uch.edu.pe), [echiroque@uch.edu.pe](mailto:echiroque@uch.edu.pe)

**Abstract**– This study presents a methodology for integrating sustainability into the teaching of the Requirements Engineering course by employing active learning strategies, tools such as BERT (Bidirectional Encoder Representations from Transformers), and the conceptual framework of the Karlskrona Manifesto. The proposal was implemented with 60 Systems Engineering students over an academic semester, aiming to strengthen their ability to develop sustainable software and reflect on the social, environmental, technical, economic, and individual impact of their design decisions.

The approach comprised three phases: initial analysis, design and implementation of active strategies, and final evaluation. In the initial phase, students' perceptions were assessed, and sustainability metrics were defined, including energy consumption, accessibility, and social relevance. Throughout the semester, students engaged in collaborative projects addressing real-world problems, incorporating environmental impact simulators and the principles of the Karlskrona Manifesto. At the end of the course, progress in knowledge, skills, and perceptions was measured through surveys and interviews.

The results indicated a 70% increase in students' ability to apply sustainability metrics and an 85% improvement in their perception of the relevance of these concepts. Additionally, the projects achieved a 20% reduction in energy consumption and an 85% compliance rate with accessibility standards. The methodology proved to be effective and replicable, validating the use of tools such as BERT and integrative frameworks like the Karlskrona Manifesto to train engineers committed to sustainability.

**Keywords**– Sustainability in Requirements Engineering, Active learning in Software Engineering, BERT and Sustainability metrics, Karlskrona Manifesto and Sustainability education, Sustainable software development.

## I. INTRODUCTION

Sustainability has become a fundamental component in the education of systems engineers, who play a strategic role in developing responsible technological solutions. This approach responds to the increasing demands from regulators and consumers for systems that minimize their environmental impact while maximizing resource efficiency [1].

In the field of software development, practices such as carbon footprint assessment and energy-efficient design have been adopted as key strategies for sustainably managing technological systems. These practices, combined with innovative methodologies such as eco-design, enable the integration of environmental criteria from the early design stages, maximizing positive environmental impact and optimizing computational resources [2].

As software systems become increasingly complex, the need to consider energy and environmental impact during their development has become more evident. Recent research has emphasized the importance of using sustainability frameworks such as the Karlskrona Manifesto, which highlights the multidimensional nature of sustainability by encompassing environmental, economic, social, technical, and individual dimensions. This framework provides a comprehensive approach to analyzing and mitigating the impacts associated with technological systems [3].

From an educational perspective, learning assessment indicators play a crucial role in measuring the understanding and application of sustainability principles. Specific metrics, such as the energy consumption of developed systems and their level of accessibility, have been defined to enable students to understand and apply these principles in practical engineering scenarios [4]. Additionally, these initiatives are complemented by efforts to integrate sustainability into other courses within the academic curriculum, such as Programming and Software Architecture, thereby strengthening an interdisciplinary approach in engineering education [5].

The integration of advanced tools and innovative educational methods further contributes to this landscape. For instance, massive open online courses (MOOCs) provide an effective platform for training in applied sustainability, incorporating practical simulations and real data analysis to address complex challenges, such as cybersecurity and optimization in advanced electrical systems. This approach fosters not only technical learning but also a responsible and critical perspective toward the design and development of sustainable technological solutions [6].

This study proposes an educational model specifically designed for Systems Engineering students, using the Requirements Engineering course as a case study. This model combines innovative methodological approaches with the assessment of environmental and social impacts, fostering a comprehensive perspective on sustainability among future engineers. Furthermore, this approach is expected to be replicable in other courses within the curriculum, enhancing the holistic training of engineering professionals [7].

## II. METHODOLOGY

The Requirements Engineering course, offered in the fourth semester of the Systems Engineering program at a university in Northern Lima, follows a theoretical-practical

approach aimed at enabling students to develop software models based on user requirements.

The study adopted a mixed-methods approach, combining quantitative instruments (surveys) and qualitative techniques (interviews) to strengthen the validity and depth of the results.

Throughout the course, participants acquire competencies in identifying, capturing, and managing requirements using international standards such as BPM, CMMI, and RUP, as well as agile methodologies like Scrum. Additionally, specialized tools and techniques are integrated for prototype design, business process management, and the application of innovative trends.

The relationship between this course and the present study lies in its emphasis on sustainability within Requirements Engineering. The principles and methodologies covered in the course serve as a foundation for developing educational proposals that promote the integration of international standards, the use of advanced technological tools, and social responsibility in system design. Consequently, the course content aligns with the research objective: to establish educational strategies that foster sustainable practices in Requirements Engineering, contributing to students' professional development and digital transformation in organizations.

BERT (Bidirectional Encoder Representations from Transformers) is a language model developed by Google in 2018 that has revolutionized Natural Language Processing (NLP) [8]. Its architecture is based on transformers, allowing it to understand a word's context by considering both its left and right surroundings within a sentence—a feature known as bidirectionality. This approach significantly enhances the ability of machines to interpret and generate natural language. In the context of Requirements Engineering, BERT has been used to improve the completeness of requirements written in natural language, helping to identify potential omissions and ensuring that specifications are more precise and comprehensive.

The Karlskrona Manifesto for Sustainability Design is a declaration that underscores the responsibility of software designers to consider the long-term consequences of their decisions [9]. It proclaims that sustainability is a systemic and multidimensional aspect that should be integrated into all stages of system design. The manifesto advocates principles such as long-term thinking, the consideration of multiple sustainability dimensions (environmental, social, economic, etc.), and the importance of evaluating the impact of systems within a broader context. In the field of Requirements Engineering, these principles guide the incorporation of practices that ensure that developed systems not only meet current needs but are also sustainable and beneficial for future generations [10].

TABLE I  
PRINCIPLES OF THE KARLSKRONA MANIFESTO FOR SUSTAINABILITY DESIGN

Principle	Description
Holistic thinking	Promotes an integral perspective that considers the environmental, social, economic, and technical aspects of systems.
Long-term thinking	Prioritizes sustainable long-term impact over quick solutions that compromise the future.
Context awareness	Recognizes that sustainability varies according to social, cultural, and economic contexts and must adapt accordingly.
Multi-disciplinary collaboration	Encourages collaboration among disciplines to address sustainability challenges from multiple perspectives.
Human-centric design	Places people and their needs at the center of design, also considering future operations.
Minimization of negative impacts	Seeks to minimize negative impacts on the environment, society, and the economy throughout the system's lifecycle.
Stakeholder engagement	Involves all relevant stakeholders to ensure that sustainability goals are shared and understood.
Lifecycle perspective	Considers environmental, social, and economic impact throughout the system's lifecycle, from creation to disposal.
System thinking	Approaches problems as part of broader systems, understanding the interdependencies and cascading effects of decisions.
Innovation for sustainability	Promotes innovation with the goal of creating solutions that drive sustainability.

The methodology applied in the Requirements Engineering course integrated active learning strategies, tools such as BERT, and the principles of the Karlskrona Manifesto to guide the teaching and application of sustainability. This approach enabled students to understand and apply sustainability metrics in real-world projects while reflecting on the long-term impact of their design decisions.

#### A. Initial analysis and metric definition (beginning of the semester)

The methodology implemented in the Requirements Engineering course began with an initial analysis and metric definition phase at the start of the semester. This stage focused on assessing students' prior knowledge of sustainability in the context of software development and establishing clear metrics to measure the impact of their projects.

To achieve this, surveys were conducted among 60 students to identify their perception of sustainability and its relationship with software development. The survey questions covered the five dimensions defined in the Karlskrona Manifesto: technical, environmental, social, economic, and individual. The results of this initial assessment revealed that while 50% of students



TABLE III  
PROJECT PROPOSALS BASED ON IDENTIFIED UNIVERSITY NEEDS AND THEIR  
INTEGRATION WITH SUSTAINABILITY PRINCIPLES

Area	Need	Involved stakeholders	Proposed solution	Principle addressed
Administrative	Manual resource management processes take longer than expected	Administrative staff	Automated management system to optimize resource allocation	Reduction in time and resource waste
Academic	Lack of access to real-time student performance reports	Faculty	Platform that generates real-time academic performance reports	Data efficiency and improved decision-making
Technical	Equipment wear and tear in laboratories due to inefficient maintenance	Technical staff	Predictive monitoring system for equipment maintenance	Extended equipment lifecycle and lower resource consumption
Student life	Limited visibility of available extracurricular activities	Students and administrative staff	Mobile application to centralize and promote sustainable activities	Promotion of sustainable activities and reduction of redundancies

#### B. Design and implementation of active strategies (during the semester)

The core phase of the course focused on the design and implementation of active learning strategies through collaborative projects and the use of advanced tools, enabling students to apply sustainability principles to real-world problems. Students worked in teams to develop sustainable applications, each designed to address specific challenges.

The design of these active strategies began with the formation of work teams, each with well-defined roles such as analysts, developers, and project leaders. This structure facilitated a clear division of responsibilities and effective collaboration. Each team selected a problem to solve within a specific area, such as administration, resource management, or academic support, and designed a project integrating sustainability principles. These projects were required to meet three fundamental objectives: optimization of energy resources, universal accessibility, and social impact generation.

To guide implementation, an iterative approach based on agile methodologies such as Scrum was adopted. Each team worked in sprints, which included planning, development, evaluation, and feedback phases.

First, the teams focused on optimizing energy resources by implementing efficient algorithms to reduce the energy consumption of their applications. Additionally, they ensured accessibility compliance by adhering to international standards such as WCAG (Web Content Accessibility Guidelines), promoting inclusivity for all users.

Finally, each team evaluated the social impact of its system, analyzing how their developments could improve people's quality of life, aligning with the social and individual dimensions of the Karlskrona Manifesto. Intermediate results showed an average reduction of 20% in energy consumption and 85% compliance with accessibility standards.

TABLE IV  
ENERGY CONSUMPTION REDUCTION

Team	Energy reduction (%)	Implementation details
1	18	Optimization of algorithms to reduce unnecessary cycles
2	22	Use of more efficient data structures
3	19	Elimination of redundant processes in business logic
4	20	Optimization of database access
5	21	Implementation of a caching system to reduce load
6	21	Reduction of unnecessary calls to external services
7	19	Use of data compression to minimize transfer
8	23	Code refactoring to improve efficiency
9	20	Optimization of SQL queries and index control
10	22	Implementation of asynchronous processes in repetitive tasks
Average	19,9	

TABLE V  
LEVEL OF COMPLIANCE WITH ACCESSIBILITY CRITERIA

Team	Accessibility compliance (%)	Implementation details
1	82	Implementation of ARIA tags for accessible navigation
2	83	Improvements in color contrast following WCAG standards
3	85	Addition of automatic subtitles in multimedia content
4	87	Adjustment of font sizes to meet standards
5	84	Adaptive design for different devices and screen sizes
6	83	Validation of accessible forms with clear error messages
7	84	Fully functional keyboard navigation
8	89	Incorporation of textual descriptions in images
9	88	Reduction of interaction response times
10	83	Compatibility with screen readers across all views
Average	85.6	Average accessibility compliance.

The Karlskrona Manifesto was used as a conceptual framework to guide students in reflecting on design decisions from a systemic perspective. This manifesto emphasizes that sustainability is not merely an inherent property of systems but is intrinsically connected to their social, environmental, and technical contexts. During class sessions, students analyzed how seemingly technical decisions, such as choosing development technologies, could have a significant impact on social and environmental dimensions. These analyses were complemented by explanations and real-world examples presented in class, where teams assessed practical sustainability scenarios using the five dimensions proposed in the manifesto: environmental, technical, social, economic, and individual.

To strengthen technical analysis, digital simulators were used to evaluate sustainability metrics such as energy consumption and carbon footprint in real time. Although Joulemeter, a tool previously used for this purpose, is no longer publicly available, modern alternatives such as Visual Studio, which now includes energy estimation capabilities, were proposed. Visual Studio enables students to analyze the energy impact of their applications and optimize their consumption during development. Additionally, other complementary tools were suggested, including Intel Power Gadget, which measures real-time energy consumption and processor efficiency, and

EnergyScope, which provides detailed metrics on energy impact in complex systems.

These tools allowed teams to rigorously evaluate their projects with a technical focus aligned with sustainability principles. For instance, a team developing an inventory management system used these tools to analyze different hardware configurations, achieving a 15% reduction in energy consumption.

Beyond facilitating quantitative evaluation, these tools also promoted informed and responsible decision-making in system design. Throughout the semester, students received continuous formative feedback, focusing on compliance with established sustainability metrics and the effective use of tools such as BERT and the Karlskrona Manifesto principles.

This feedback process not only reinforced learning but also helped teams refine their projects and better understand the importance of sustainability in all phases of software development.

### C. Evaluation of learning impact and student perception (end of the semester)

The final phase of the course focused on evaluating the impact of learning, student perception, and performance in applying sustainability principles through the developed projects.

To achieve this, exit surveys were conducted to compare the initial and final results of the semester. These surveys revealed a 70% increase in students' knowledge of specific sustainability metrics and an 85% improvement in their perception of sustainability's relevance in software development. These findings demonstrated the positive impact of the methodological approach on students' understanding and appreciation of sustainability.

Additionally, qualitative interviews were conducted to gain deeper insights into students' experiences. Many participants emphasized that the Karlskrona Manifesto was a key tool for addressing complex problems from a systemic perspective, encouraging them to reflect on the long-term impact of their design decisions. These qualitative reflections underscored the importance of integrating multidimensional perspectives into sustainability education.

TABLE VI  
EXIT SURVEY RESULTS

Principle	Initial knowledge (%)	Final knowledge (%)	Initial relevance perception (%)	Final relevance perception (%)
Holistic thinking	30.0	51.0	50.0	92.5
Long-term thinking	35.0	59.5	55.0	100.0
Context awareness	40.0	68.0	60.0	100.0
Multi-disciplinary collaboration	32.0	54.4	52.0	96.2
Human-centric design	28.0	47.6	48.0	83.8
Minimization of negative impacts	31.0	52.7	50.0	92.5
Stakeholder engagement	36.0	61.2	55.0	100.0
Lifecycle perspective	34.0	57.8	53.0	98.5
System thinking	29.0	49.3	48.0	92.9
Innovation for sustainability	33.0	56.1	54.0	99.9
Average	32.8	55.7	52.7	87.2

Finally, the results of the developed projects were evaluated. All teams successfully met the predefined sustainability metrics, demonstrating the effectiveness of the pedagogical strategies and the use of advanced tools such as BERT and the Karlskrona Manifesto principles.

Students highlighted the usefulness of these resources in ensuring that their systems were not only technically efficient but also socially responsible and environmentally sustainable. These results demonstrated how the adopted educational approach fostered a comprehensive understanding of sustainability and its practical application in software projects.

### III. DISCUSSION

This research highlights the positive impact of the methodological approach in teaching sustainability applied to software development, as evidenced by a 70% increase in students' knowledge of sustainability metrics and an 85% improvement in their perception of its relevance. These findings

align with the analysis conducted in [11], which emphasizes that although the ICT industry has made significant progress in hardware energy efficiency, software continues to substantially contribute to the increase in energy consumption due to its growing complexity and lack of energy optimization.

For instance, [11] highlights that software design has evolved toward high-level programming languages that, while enhancing productivity, increase energy consumption. This finding is consistent with our observation that practical activities, such as algorithm optimization in student projects, led to an average 20% reduction in energy consumption, underscoring the potential for improving sustainability through better programming practices.

Regarding accessibility, this study revealed an 85% compliance rate with WCAG standards, reflecting a strong commitment to inclusion. According to [11], digital service design should incorporate technical and social sustainability principles to avoid the phenomenon of "black software," characterized by inefficient and resource-intensive systems. This proactive approach to accessibility not only validates Manner's observations but also reinforces the role of software as a facilitator of sustainability across multiple dimensions.

Furthermore, [11] underscores the necessity of educating future software developers about the impact of design on global energy consumption, arguing that hardware improvements alone are insufficient to offset the increasing energy demands of software. In our research, this aspect was addressed by integrating sustainability metrics into student projects, fostering a deeper understanding of software's impact within the ICT ecosystem.

Several studies reinforce the importance of integrating sustainability into software development education. Bambazek et al. [3] highlight that embedding sustainability concerns at the requirements engineering stage significantly enhances the long-term social and environmental benefits of software systems. Similarly, Khalifeh et al. [4] emphasize that applying structured conceptual frameworks improves the alignment between technological innovation and sustainability goals. These findings support the effectiveness of using strategies such as BERT-based analysis and the Karlskrona Manifesto to cultivate sustainability-oriented thinking among future engineers.

Finally, this study also contributes to the ongoing discussion on the relationship between sustainability and ICT education. The software industry must adopt a more responsible and conscious approach to software development. The findings of this research support this position, emphasizing that educational interventions can serve as effective catalysts for sustainable transformation in software development.

Additionally, the results obtained in this study align with previous research, such as the study on the performance of environmental management systems according to the ISO 14001 standard [12]. This work highlights that implementing management systems based on international standards, including structural models with latent and manifest variables,

not only improves organizations' environmental performance but also allows for evaluating and optimizing practices through specific sustainability indices. These findings reinforce the importance of a structured approach to integrating sustainability into software projects, as the use of clear metrics and methodologies—such as those applied in this study—facilitates planning, continuous improvement, and results evaluation.

In particular, the use of key indicators and analytical tools to measure environmental impact is consistent with the strategies implemented in this research. By applying metrics related to energy consumption and accessibility, the hypothesis that incorporating sustainable practices from the early design stages can have a significant impact on both technical efficiency and social sustainability was validated. This approach highlights the crucial role of education in preparing future developers to effectively address the systemic challenges of sustainability.

Although the number of participants in this study was 60, future research could benefit from expanding the sample size to enhance the robustness and generalizability of the results.

#### IV. RESULTS

The results obtained reflect the positive impact of the methodology on student learning and performance, as well as the quality of the developed projects.

##### A. Learning indicators

The findings from the learning indicators demonstrate a significant improvement in students' ability to understand and apply sustainability principles in software development.

At the beginning of the semester, only 50% of participants considered sustainability a fundamental aspect of their engineering education, indicating a limited perception of its importance in professional practice. However, after implementing an active learning methodology—which included collaborative projects and tools like BERT—this perception increased to 85%.

This shift suggests that the combination of theory and applied practice enabled students to internalize sustainability as a key criterion in decision-making within Requirements Engineering. Additionally, discussion sessions based on the Karlskrona Manifesto promoted a deeper understanding of the interconnections between the technical, environmental, and social aspects of software development, strengthening this new perspective.

In terms of practical skills, the integration of BERT for non-functional requirements analysis significantly improved students' ability to identify and classify sustainability-related elements. Compared to manual methods, using BERT reduced key requirement omissions by 25%, optimizing the documentation and analysis process in the early stages of software development. This improvement was fundamental for teams to structure projects that aligned with sustainability

metrics, ensuring that energy consumption, accessibility, and social viability were considered from the initial phases.

The use of BERT enabled students to develop a more analytical and sustainability-oriented mindset, better preparing them to tackle real-world challenges in the software industry and facilitating the implementation of sustainable engineering decisions.

##### B. Impact on projects

The student projects demonstrated a significant impact on sustainability, particularly in energy efficiency, accessibility, and social contribution.

One of the most notable achievements was the 20% average reduction in energy consumption of the developed applications. This was achieved through algorithm optimization and the efficient use of hardware configurations. Students applied strategies such as minimizing background processes, using optimized data structures, and selecting lower-energy-consuming technologies. Simulation and performance analysis allowed them to identify critical points where resource usage could be reduced without compromising functionality. This approach not only strengthened their ability to design energy-efficient solutions but also promoted a culture of environmentally conscious software development.

Regarding accessibility, 85% of the projects fully complied with WCAG standards, ensuring that applications were inclusive and accessible to people with disabilities. Throughout the course, teams were trained in accessibility best practices, including the use of semantic labels, proper contrast in graphical interfaces, and screen reader compatibility. This integration not only enhanced the usability of the applications but also reinforced the importance of designing software for a broader audience.

Finally, the social contribution of the projects was evaluated based on their impact on specific communities, aligning with the Karlskrona Manifesto principles. Students analyzed how their solutions could improve users' quality of life and reduce inequalities in technology access, helping them understand sustainability not just as a technical concern but also as an ethical and social responsibility in software engineering.

##### C. Validation of the approach

The validation of the methodological approach applied in the course focused on the practical incorporation of Karlskrona Manifesto principles and the systematic reflection on the impact of design decisions on various sustainability dimensions. This conceptual framework was essential in guiding students in discussions on sustainability, enabling them to identify how technical decisions influence the environmental, social, economic, technical, and individual dimensions in an interconnected manner.

The Karlskrona Manifesto was introduced as a core element, where case studies illustrating the challenges and opportunities of implementing sustainability in software



projects were analyzed. Students reflected on complex issues, such as the selection of low-impact technologies or the inclusion of vulnerable groups through accessible interfaces and discussed how these decisions affected not only the developed system but also its broader context. These reflections translated into tangible improvements in their project designs.

One component of the validation process was assessing the impact of practical activities on students' understanding of sustainability. Collaborative projects were designed to address real-world challenges, allowing teams to explore the limits of sustainable principles in complex scenarios. This practical experience reinforced the importance of considering the socio-technical context of developed systems, exposing students to challenges such as resource constraints, compliance with international standards, and end-user expectations.

Additionally, students proposed projects that identified internal needs within various university areas. As part of this process, they engaged with key stakeholders, including administrative staff, faculty, and technical personnel, to validate the development of systems addressing real and specific problems. This interaction not only fostered a deeper understanding of requirements but also enabled students to integrate sustainability principles directly into solutions relevant to the university community. These practical experiences strengthened both their technical and social skills, demonstrating how technological solutions can address concrete needs while promoting positive community impacts.

During discussions and workshops, students emphasized how the Karlskrona Manifesto had broadened their perspective on software development. Many recognized that before taking the course, their understanding of sustainability was limited to technical aspects, such as energy efficiency. However, through the application of the manifesto, they developed a more holistic perspective.

The validation process also included feedback sessions with stakeholders involved in the projects. These interactions not only allowed teams to refine their systems to better meet real-world needs but also highlighted the importance of applying sustainability principles to practical problems. Stakeholders confirmed that the proposed solutions met their expectations while providing additional benefits in terms of efficiency and accessibility.

In conclusion, the validation of the approach used in the course demonstrated that integrating Karlskrona Manifesto principles, developing practical projects, and collaborating with real stakeholders enables students to effectively understand and apply sustainability in software design. This experience enhanced their ability to reflect on the long-term impact of their design decisions, fostering a more ethical and responsible approach in their future professional practice.

## ACKNOWLEDGMENT

We thank the students who participated in the course and the university's faculty and administrative staff for their collaboration with the study.

## REFERENCES

- [1] V. P. Rodrigues, D. C. A. Pigosso, and T. C. McAlloone, "Process-related key performance indicators for measuring sustainability performance of ecodesign implementation into product development," *Journal of Cleaner Production*, vol. 139, pp. 416-428, August 2016. Available: <http://dx.doi.org/10.1016/j.jclepro.2016.08.046>.
- [2] V. Camañes, R. Tobajas, and A. Fernandez, "Methodology of Eco-Design and Software Development for Sustainable Product Design," *Sustainability*, vol. 16, no. 7, p. 2626, March 2024. Available: <https://doi.org/10.3390/su16072626>.
- [3] P. Bambazek, I. Groher, and N. Seyff, "Requirements engineering for sustainable software systems: a systematic mapping study," *Requirements Engineering*, vol. 28, pp. 481-505, June 2023. Available: <https://doi.org/10.1007/s00766-023-00402-1>.
- [4] A. Khalifeh, P. Farrell, M. Alrousan, S. Alwardat, and M. Faisal, "Incorporating sustainability into software projects: a conceptual framework," *International Journal of Managing Projects in Business*, vol. 13, no. 6, pp. 1339-1361, July 2020. Available: <https://doi.org/10.1108/IJMPB-12-2019-0289>.
- [5] E. Kern, M. Dick, S. Naumann, and T. Hiller, "Impacts of software and its engineering on the carbon footprint of ICT," *Environmental Impact Assessment Review*, vol. 52, pp. 53-61, August 2014. Available: <http://dx.doi.org/10.1016/j.eiar.2014.07.003>.
- [6] R. Melendez-Norona, M. M. Larrondo-Petrie, and E. Sagredo, "Massive Open Online Course in Electrical Microgrids Cybersecurity, State Estimation and Optimization utilizing Open Science Datasets," *21st LACCEI International Multi-Conference for Engineering, Education, and Technology*, Buenos Aires, Argentina, July 2023. Available: <https://doi.org/10.18687/LACCEI2023.1.1.1671>.
- [7] F. Cluzel, B. Yannou, D. Millet, and Y. Leroy, "Eco-ideation and eco-selection of R&D projects portfolio in complex systems industries," *Journal of Cleaner Production*, vol. 112, pp. 4329-4343, August 2016. Available: <http://dx.doi.org/10.1016/j.jclepro.2015.08.002>.
- [8] A. F. Subahi, "BERT-Based Approach for Greening Software Requirements Engineering Through Non-Functional Requirements," *IEEE Access*, vol. 11, pp. 103001-103018, September 2023. Available: <https://doi.org/10.1109/ACCESS.2023.3317798>.
- [9] C. Becker, R. Chitchyan, L. Duboc, S. Easterbrook, B. Penzenstadler, N. Seyff, and C. C. Venters, "Sustainability Design and Software: The Karlskrona Manifesto," *International Conference on Software Engineering (ICSE'15)*, Florence, Italy, May 2015, pp. 467-476. Available: <https://doi.org/10.1109/ICSE.2015.179>.
- [10] R. Chitchyan, C. Becker, S. Betz, L. Duboc, B. Penzenstadler, N. Seyff, and C. C. Venters, "Sustainability Design in Requirements Engineering: State of Practice," *International Conference on Software Engineering (ICSE'16)*, Austin, USA, May 2016, pp. 533-542. Available: <https://doi.org/10.1145/2889160.2889217>.
- [11] J. Manner, "Black software—The energy unsustainability of software systems in the 21st century," *Oxford Open Energy*, vol. 00, pp. 1-8, 2023. Available: <https://doi.org/10.1093/oenenergy/oiac011>.
- [12] M. Hadini, M. B. Ali, S. Rifai, O. Bouksour, and A. Adri, "Assessment Tool for Environmental Management System Performance According to the ISO 14001," *International Journal of Management*, vol. 10, no. 5, pp. 73-83, October 2019. Available: <http://iaeme.com/Home/issue/IJM?Volume=10&Issue=5>.