Challenges in Software Engineering for Green Software Development

Saroj Singh[1], Alok Kumar Gupta[2], Kamal Soni[3], Sonam Singh[4]

[1, 2, 3, 4] Department of Computer Science & *Engineering, Babu Banarasi Das Engineering College, Lucknow, U.P., India*

---***---

Abstract – *Green software, also known as energy-efficient software, has just lately become a research focus in the software engineering community, but recent interviews with software engineers indicate that they still lack the necessary skills, knowledge, and resources to build greener software. According to past studies, there are currently no courses or teaching materials in engineering that teach green software. However, many universities' courses introduce students to sustainable energy resources in their final year of engineering. These findings highlight the need of motivating students to participate in green software efforts. We investigated the difficulties and challenges encountered by students in developing green software and proposed some solutions to contribute to green software because we firmly believe that "green thinking" and the widespread use of green software projects should be encouraged among students from the start of their curriculum, as this may significantly benefit our environment, society, and the students themselves as they begin their careers.*

Key Words: **Green Software Project, Green Computing, Green Education, Sustainability, Software Engineering.**

1. INTRODUCTION

The surge in data usage and the number of computing devices can be attributed to the computing technology's transformation over the previous few decades into an indispensable component of the global infrastructure. The utilization of computing resources in an environmentally beneficial manner is made possible by green computing. It can also be described as the study of creating, utilizing, and disposing of computing equipment in a way that lessens their impact on the environment. Since people are now shown their interest towards environment and now using only green products and thus, the green movement and sustainability are topics that the world is becoming more aware of and concerned about. Nowadays, people are more technology dependent and using computer software almost for every task to solve their day to day life problems and since computers and the software they run are so important to our society and the health of our planet, they bear a special duty in these areas and thus, this accelerate the need of developing green products in the field of computer science. The ubiquitous use of computers, including laptops, powerful mobile phones, consumer gadgets, and massive data centers, is altering how software engineers create software. After much research and development in green cloud, the focus is now shifter toward the software development process. Indeed, there are new issues that developers must take into account while creating software systems in the approaching era of edge computing, AI, and the Internet of Things (IoT). Energy consumption is now the primary bottleneck for constructing such systems, whereas in the previous century both computer makers and software developers were primarily focused on producing incredibly fast computer systems [16].

The 21st century's primary software engineering process is green software engineering. Software engineers used to be primarily concerned with the creation of hardware or software, placing less emphasis on sustainability. Now, we have approached the time where it is required ecological activity on a global scale to lessen the effects of climate change. For a variety of reasons, software and IT are crucial to environmentally friendly activities. Leaders in the software industry must adopt green IT and green coding because IT systems alone consume 10% of the world's electricity.

The use of software and IT to run our society and run our personal lives is constantly expanding. However, as they expand, so does their energy need? Data centers alone will use almost 10% of the world's electricity by 2030 [15]. One-third of the world's need for energy will be met by energy consumption, including Internet, telecommunication, and embedded devices. Knowing that end users only consume what we have to offer, the software development community needs to take a proactive stance in favor of eco-friendly practices. Today, green IT is in demand. Every single piece of code we create today may continue to operate on countless processors years from now, using energy and accelerating climate change.

The terms "green IT" and "green coding" refer to a paradigm shift in how software engineers, developers, testers, and IT administrators may make their solutions and services more energy efficient. Every software developer contributes in some way. Despite substantial advances in hardware energy efficiency, the programming language and software engineering communities have just recently begun conducting research on designing energy efficient software, or green software. The ICT4S and IGSC conferences, the GREENS, RE4SuSy, and MeGSuS workshops, as well as the most recent research publications in the fields of green data structures [17, 20, 21], green software libraries [18], green rankings of programming languages [22], green programming practices/patterns [23, 24, 25, 26, 27], and green software metrics and development[28, 29], green computing research and education infrastructure [32], green repositories [34], energy optimization for databases [36], green web [30], green cloud [31, 33, 35], green AI [7, 22, 41], and green computing research show that, despite being in its early stages, green software research is quickly gaining attention.

Software development has a double-edged sword when it comes to sustainability. Through innovations in energy, manufacturing, agriculture, transportation, and other areas, software solutions will be essential for the environment both today and in the future (Infotech, 2021)[19]. These are essential for solving issues like deforestation and lowering emissions, for example. But software also contributes to a fast expanding carbon footprint (Podder et al., 2020)[15]. Despite the fact that research into green software is expanding quickly, recent interviews with software developers reveal that they still lack the skills and resources necessary to create greener software [16, 37, 38, 39]. For instance, the number of questions posted on Stack Overflow related to software energy consumption increased rapidly, but most of them were not answered or poorly answered [39]. The results of a targeted study of 33 academics on the use of green and sustainable software engineering in higher education were reported [6].

The key conclusions showed that there is a lack of sustainability content in the software project made by students as part of their curriculum activity of engineering, and the main causes are:

1. lack of awareness about sustainable software,

2. lack of teaching resources,

3. high level of work required to learn principles of

   sustainable Software engineering

4. lack of technology and tool support.

In reality, the findings of all of these recent studies suggest that academics should not only support cutting-edge research in green software design, but also educate future software engineers on how to produce greener software. Obviously, it is preferable to begin a career as a software engineer with this knowledge. Being ecologically friendly and sustainable is a current requirement. Everyone can participate in a long number of eco-conscious initiatives that work to address environmental problems. Most of us, may not be aware, though, that software may also be created, developed, and used sustainably. But unfortunately, our social and environmental responsibilities are frequently ignored in today's undergraduate computer science education [40]. In this rapidly evolving era of artificial intelligence, supercomputing, cloud computing, the internet of things, and edge computing, we believe that "green thinking" and widespread acceptance of green software development among computer science students will significantly help our world.

This study's goals are to identify methods and resources for attaining environmental sustainability in software development and to identify the barriers to adoption that computer science students (software engineer's) face. People and the environment are constantly being impacted by climate change. The value of sustainability in decreasing our environmental impact has risen in popularity, not just among the general public but also within the business community (Maryville 2019). The Brundtland Report (United Nations World Commission on Environment and Development, 1987) defines sustainable development as follows:

"Sustainable development is the development that meets the needs of the present without compromising the ability of future generations to meet their own needs."

We examined studies in the subject of green software/computing in this study. We presented the findings of the study on their thoughts on green software education. These studies reveal that undergraduate software engineers face numerous problems, including a shortage of instructional material for teaching green software. Furthermore, this study identifies important problems in delivering green software to software engine students.

2. PERSPECTIVES OF GREEN SOFTWARE DEVELOPMENT

The ubiquitous use of software has fundamentally altered virtually every aspect of our lives and employment. Many applications necessitate the use of large, complex software. In the future, software will continue to grow in relevance and use. Software, too, has a significant impact on our environment. Many people are unaware that software, like computer hardware, may have an impact on the environment. Although software does not directly consume energy, it does direct and influence how computer hardware performs, which has an indirect impact on energy consumption and carbon emissions. Computationally inefficient software can drastically increase energy consumption. Software, like computer hardware, contributes to the issue of environmental sustainability. As a result, creating and implementing software to optimize its energy consumption is critical. As a result, in addition to functionality, security, scalability, and accessibility, we must also consider the program's energy efficiency and sustainability as critical characteristics while developing greener software. Furthermore, we must develop software that uses the fewest computational and memory resources and may be utilized for an extended length of time. This is becoming increasingly important as the usage of software in all aspects of our lives grows. Although the role of software in the sustainability discussion has been largely disregarded until recently, it is now time to give it the attention it deserves. The mission statement of the Green Software Foundation declares as much. Businesses and the IT sector should employ green software as part of their sustainability goals. However, developing energy-efficient software is a tough task that necessitates the use of rules, best practices, models, and tools to measure and reduce the influence of software on the energy consumption of the underlying hardware. A 2015 IEEE IT Professional paper presents a conceptual framework for designing and developing greener software. It provides a coherent perspective of the techniques, models, and tools developed to date.

A broad word used to describe ecologically friendly IT gear, software, systems, applications, and practices is "Green IT." There are three overlapping strategies it includes for enhancing environmental sustainability.

- **IT's Greening**: This inward-looking strategy stresses reengineering IT systems and procedures to improve energy efficiency, maximize use, reduce carbon footprint, and comply with regulatory requirements. The objective of the Green Software Foundation is to green IT by providing green software.

- **Green by IT**: "Greening by IT" refers to the use of information technology to aid in the building of a sustainable environment. This forward-thinking strategy focuses on providing ground-breaking solutions to critical industries such as manufacturing, energy, business, agriculture, healthcare, and buildings, which include homes, offices, and other structures, in order to reduce emissions and resource consumption while enabling expansion.

- **Green Awareness Promotion**: Numerous people are still unaware of the terrible effects of the climate issue and are not acting to stop it. IT has the power and responsibility to inspire people, aid in their education, and increase their involvement in the battle against climate change. The promotion of "green" projects and targeted behavioral changes can thus be assisted by IT.

Making IT gear more effective and ecologically friendly has received a lot of attention and advancement over the years. However, we must extend our green activities all the way to the software in order to reap the full benefits. Hence the renewed interest in developing eco-friendly software.

In order to lower carbon emissions caused by software creation and use, we underline the importance of generating awareness of greening software development among the computer science students and look at ways to do so in this study.

3. REVIEW OF RELATED LITERATURE

The current climate problem is worsening. To tackle this watershed moment in our history, which affects all of us and future generations, quick and relentless action is essential. Although there are numerous tools and solutions that investigate ways to improve energy efficiency and reduce carbon dioxide emissions in hardware creation, green computing is a topic that has been thoroughly researched by a number of experts. There is a desire to learn more about green software development as a result of growing industrial operations, competitive awareness, and global environmental consciousness. Many businesses aspire to attain environmental sustainability. There are numerous opportunities for the information technology (IT) sector, including hardware, software, firmware, and communications, to contribute to lowering overall global emissions. According to a Global e-Sustainability Initiative research, IT solutions, or "greening by IT," can help reduce CO2 emissions by nearly ten times more than they produce. However, 'greening IT,' or reducing IT's own carbon emissions and footprint, is critical.

While there are study findings in energy-efficient hardware and its components to attain environmental sustainability, significant research is required to link hardware energy consumption to that of the software that runs on it. Software now supports a larger portion of our society, so its environmental effect and energy efficiency are more crucial. In order to attain environmentally friendly computing, green IT and green software are used. Designing new green and sustainable software or applying this notion to the current software systems to fulfill corporate need are both challenging jobs. There are many studies which indicated that, there are variances in energy usage, thus there are numerous opportunities to save energy.

M. Mohankumar, and M. Anandkumar [1] investigated that the technological, economic, environmental, social, and personal components of environmental sustainability receive no attention. In industrialized nations, research is mostly concentrated on environmentally friendly and sustainable software engineering. However, there is no more research on this in developing nations like China and India, therefore more research needs to be done on green software engineering. For the developed countries alone, developing nations are creating more software programs and ICT products, but creating effective software models requires resolving a significant problem with green sustainable software engineering. The environment and the earth, which are shared by all nations, should be the core topics of the software models that are being built. In the life cycle models for the software development process, this study analyses the green and sustainable software engineering method. This study examines current software paradigms and how they affect ecosystems. For the developed countries alone, developing nations are creating more software programs and ICT products, but creating effective software models requires resolving a significant problem with green, sustainable software engineering. The environment and the earth, which are shared by all nations, should be the core topics of the software models that are being built. In the life cycle models for the software development process, this study analyses the green and sustainable software engineering method. This study examines current software paradigms and how they affect ecosystems.

S. Tiwari (2011)[2] discovered that there has been a lot of hoopla about Green computing and giving greener computing solutions to nearly anything in the previous few years Green computing is a novel method that tries to develop computer systems that deliver higher processing and performance while using the least amount of power. Numerous research and surveys have already established that electricity costs account for the lion's share of total data center management expenditures. Green computing initiatives have hitherto been limited to reducing power usage inside the corporate context. This study attempted to demonstrate the significance of green computing beyond lowering power consumption and toward providing higher customer value and, ultimately, achieving the ultimate aim of sustainable IT growth.

R. Abdullah, S. Abdullah, and J. Din (2015) [3] discovered, and that the key asset in the software industry is the knowledge of software development personnel Knowledge is the most important asset for achieving success in green, sustainable development. There is no research on implementing Knowledge Management (KM) strategies to aid in the management of Green Software Development knowledge (GSD).

P. Lago(2015) [4] stated that with the increasing role played by software in supporting our society, its sustainability and environmental impact have become major factors in the development and operation of software-intensive systems. Myths and beliefs hide the real truth behind Green IT: IT is energy-inefficient because software is developed to make it so – intentionally or not. But how far are we from being able to control software energy-efficiency? What makes software greener? How can we transform measuring software energy consumption in a general practice? What architectural design decisions will result in more sustainable systems? How can we ensure that new-generation software will be both cloud-ready and environmental-friendly? And how can we make evident the economic and social impact of developing software with 'energy in mind'? These are a few of the challenges ahead for a more sustainable digital society. And concluded with directions for exciting challenges, promising opportunities, and ultimately inspiring research.

M. Patel (2017) [5] Green ICT was discovered to be an important strategic technology with numerous benefits such as cutting greenhouse gas emissions, lowering electricity costs, establishing a positive company image, and so on. The purpose of this study is to determine the awareness and attitude of IT/Computer engineering students toward Green ICT. This study also investigates younger people's acceptance of Green ICT practices, as well as the reasons, barriers, and their perspectives on IT. The current study was conducted by surveying IT/Computer engineering user groups at the LDRP-ITR, KSV University Campus, utilizing random sample procedures and questionnaire tools. Following the investigation, a few recommendations were made to expand the adoption of Green ICT among young people while minimizing the negative impact to the environment. Academic libraries are an essential component of academic institutions, serving an important role in teaching, learning, scholarships, and research for academic communities. Librarians take measures to raise awareness and adoption of Green ICT among academic communities in order to provide a nice and healthy environment for the community's betterment and, as a result, to protect the global environment.

D. Torre, G. Procaccianti, D. Fucci, S. Lutovac, and G. Scanniello (2017) [6] found during their study that, software is pervasive in our everyday lives in this decade. Its sustainability and environmental impact have become major factors to be considered in the development of software systems. Millennials–the newer generation of university students– are particularly keen to learn about and contribute to a more sustainable and green society. The need for training on green and sustainable topics in software engineering has been reflected in a number of recent studies. The goal of this paper is to get a first understanding of what is the current state of teaching sustainability in the software engineering community, what are the motivations behind the current state of teaching, and what can be done to improve it. To this end, they report the findings from a targeted survey of 33 academics on the presence of green and sustainable software engineering in higher education. The major findings from the collected data suggest that sustainability is under-represented in the curricula, while the current focus of teaching is on energy efficiency delivered through a fact-based approach. The reasons vary from lack of awareness, teaching material and suitable technologies, to the high effort required to teach sustainability. Finally, they provided recommendations for educators willing to teach sustainability in software engineering that can help to suit millennial students needs.

D. Gil, J. L. Fernández-Alemán, and J. Trujillo (2018) [7] reported in their article that unfortunately, sustainability is an issue very poorly used when developing software and hardware systems. Lately, and in order to contribute to the earth sustainability, a new concept emerged named Green software which is computer software that can be developed and used efficiently and effectively with minimal or no impact to the environment. Currently, new teaching methods based on students' learning process are being developed in the European Higher Education Area. Most of them are oriented to promote students' interest in the course's contents and offer personalized feedback. Online judging is a promising method for encouraging students' participation in the e-learning process, although it still has to be researched and developed to be widely used and in a more efficient way. The great amount of data available in an online judging tool provides the possibility of exploring some of the most indicative attributes (e.g., running time, memory) for learning programming concepts, techniques and languages. So far, the most applied methods for automatically gathering information from the judging systems are based on statistical methods and, although providing reasonable correlations, these methods have not been proven to provide enough information for predicting grades when dealing with a huge amount of data. Therefore, the great novelty of this paper is to develop a data mining approach to predict program correctness as well as the grades of the students' practices. For this purpose, powerful data mining technologies taken from the artificial intelligence domain have been used. In particular, in this study, they have used logistic regression, decision trees, artificial neural network and support vector machines; which have been properly identified as the most suitable ones for predicting activities in the e-learning domains. The results have achieved an accuracy of around 74%, both in the prediction of the program correctness as well as in the practice grades' prediction. Another relevant issue provided in this paper is a comparison among these four techniques to obtain the best accuracy in predicting grades based on the availability of data as well as their taxonomy. The Decision Trees classifier has obtained the best confusion matrix, and time and memory efficiency were identified as the most important predictor variables. In view of these results, they concluded that the development of green software leads programmers to implement correct software.

M. V. Palacin-Silva, A. Seffah, J. Porras (2018) [8] stated that sustainable development (SD) has become a millennium challenge for humanity. It has boosted the integration of sustainable, sound practices across different fields (e.g. e-participation, smart transportation, sustainable agriculture). Information and Communication Technologies (ICT) have strongly supported social transitions in becoming more sustainable and participative over the last two decades. Integrating multi-disciplinary sustainability concepts into the higher education of computer scientists is important in that this ensures that future ICT endeavors will take sustainability concerns into account. This article describes four capstone projects developed by students from the Erasmus Mundus Master Course in Pervasive Computing and Communications for Sustainable Development (PERCCOM), who were enrolled alongside regular students in a traditional software engineering course held at the Lappeenranta University of Technology. The coursework was part of a research project called Green.Citizen@ICT, which aims at investigating the use of ICT and software services for infusing sustainable habits in citizens through the development of applications for SD. This study demonstrates how a sustainable development focus can be integrated into a traditional software engineering course. The goal of this article was to enhance understanding of the integration of sustainability into software engineering education, by providing a detailed example of a master course in which this took place. This course supported the development of ICT competences for building cleaner, greener, and more resource- and energy-efficient cyber-physical systems, while addressing the social and environmental dimensions of sustainable development.

Y. Chamekha and M. A. Hammamib (2020) [9], According to their findings, virtual reality is taking centre stage in education. For students to obtain relevant skills, a virtual world must be introduced so that they can achieve the practical aspect required in the workplace environment. Memorizing data bores pupils; thus, the requirement for virtual reality to assist students in gaining competence. The purpose of this study is to analyze the role of virtual reality in current education. The theoretical method and quantitative study were utilized to understand the effects of virtual reality in the education sector. The study's findings indicate that virtual reality can help children with special education requirements learn new skills. Students' self-esteem improves when they use virtual reality.

C. Calero, J. Mancebo , F. Garc ıa, M. A. Moraga, J. Alberto G. Berna, J. L. Fernandez-Aleman, and A. Toval (2020) [10], Green and Sustainable Software, they believe, has arisen as a new and active sector in the software world. They feel that after several years of research and work, it is now required to gain a general snapshot of how research in this area is evolving. To accomplish this, they used the 5Ws (why, when, who, where, and what), a technique for obtaining the entire story about a subject. They conducted a study using 542 papers pertaining to Green and Sustainable Software research that were found using SCOPUS. As a result of the findings, they argue that it is critical to define crucial parts of research in order for researchers to be completely aware of the state of the research on Green and Sustainable Software (why); The study draws on papers published between 2000 and the beginning of November 2018 (when); the most prolific authors are primarily from Europe, though the United States is the most active country, Green and Sustainable Software being a highly interactive field with a large number of multinational publications (who).

A. Iatsyshyn, A. Iatsyshyn, V. Artemchuk, I. Kameneva, V. Kovach, and O. Popov(2020)[11], According to their findings, in order to satisfy Ukraine's promises to the world community on climate change prevention, signed documents on renewable energy development, modernization of fuel and energy sector firms, and waste management must be followed. As a result, developing software to tackle problems of visual analysis of environmental state dynamics of territorial systems and determining the stability of particular territories is an urgent challenge. The authors offer new forms of monitoring data presentation of technological genic loadings and dangers that depict environmental situation changes in the space of informative aspects. It is critical to strengthen the abilities of professionals, particularly those in ministries, businesses, and organizations with decision-making authority, in order to lessen the harmful influence on the environment. It is proposed to improve the qualification of such management specialists in the following areas:

conducting training seminars at ministries, institutions, and departments interested in the implementation of developed systems; scientific and methodological support and advisory assistance in the software implementation process; development and improvement of educational and methodological support for postgraduate students and advanced trainees.

J. Saraiva, Z. Zong, and Rui Pereira (2021) [12], It has been stated that the software engineering community has only lately begun to perform research on building energy efficient software, also known as green software. When compared to previous research in the computer hardware community, this pales in comparison. While research into green software is expanding rapidly, several recent surveys with software developers demonstrate that they still lack strategies, expertise, and resources to design greener software. Indeed, all of these research show that green software should be included in a current computer science curriculum. In this study, they report survey data from both researchers and educators on green software education. These results confirm the present lack of courses and training materials for teaching green software in higher education. Furthermore, we emphasize three major pedagogical problems in implementing green software. Furthermore, they identify three major pedagogical obstacles in integrating green software into computer science curricula and examine known solutions to these major challenges. They are adamant that “green thinking” and the widespread use of green software in computer science curricula may tremendously help our environment, society, and students in an era when software is ubiquitous and evolves at an unparalleled rate.

A. Q. Mohabuth (2022) [13], According to reports, universities today have a large infrastructure of computing resources, including data centres that consume a lot of energy. The increasing demand for computing resources facilitates the delivery of greater services and performance at all levels in universities, but has sadly had a severe impact on the environment. Computing resources are estimated to contribute 2% of the carbon footprint, and there is a strong need for universities to transition to green computing to reduce their environmental impact. This study referred to the example of the University of Mauritius, which is not unique to the problem and has similar problems to other universities in terms of implementing green practices on its campus. Students’ and faculty members’ objective and perceived knowledge were evaluated. First, students’ and faculty members’ objective and perceived knowledge were assessed. The influencing elements and green practices used by students and faculty members were explored. The study was directed by both a quantitative and qualitative methodology. Questionnaires were utilized to collect data on the level of green computing knowledge and practices among University students and personnel from various Faculties. The research was triangulated using qualitative approaches, in which students and staff were interviewed to build up additional difficulties with green computing, as well as validate and confirm some of the facts acquired during the quantitative stage. There were semi-structured and e-focus group interviews. The findings led to the construction of an adequate framework that allowed for green computing knowledge and its assessment, facilitating in the background energy efficiency, cloud computing, reuse, recycling, and disposal. An app that matched the findings' specifications was created. It had features that would promote the implementation of green computing practices across campus. The app was assessed and determined to be useful for universities in making green practices a duty.

R. Mehra, V.S. Sharma, V. Kaulgud, S. Podder, and A.P. Burden (2022)[14], According to studies, as sustainability becomes increasingly important in business, green and energy-efficient solutions are more important than ever. While it is becoming increasingly clear that software and the software industry are significant and quickly evolving contributors to carbon emissions, there are few methods for creating practical awareness about this within the software development lifecycle (SDLC). Can software teams understand how green their projects are? They present an industrial perspective on why this is a difficult and important topic that must be addressed. They also explain a method for quickly determining the “greenness” of a software project based on the decisions made across several SDLC dimensions, as well as show the early promising comments this method has gotten. With their article, they addressed the obstacles, possible impact, and ongoing research on gaining a holistic perspective on a software project's adoption of green decisions/practices.

According to the above review of literature, many previous studies emphasize the importance of sustainability in software development and encourage students of computer science - the future software engineers - to develop green software projects during their graduation, as many previous studies have also demonstrated that the industry faces challenges in terms of expertise, support, and knowledge. Ignorance results from a lack of methods for identifying and assessing environmental sustainability risks in software development, such as energy utilization. Furthermore, it is critical to analyse students' motivations, attitudes, and discretionary actions in order to understand how green software development implementation can be aided, maintained, and contributed as institutional content to the software engineering course.

The purpose of this research is to uncover methods and techniques for achieving environmental sustainability in software development, as well as the hurdles that impede computer science students from applying existing sustainable methods and tools.

3.1 MAJOR CHALLENGES IN ADPOTION OF GREEN SOFTWARE DEVELOPMENT & RECOMMENDATION FOR IMPROVEMENT

Software sustainability is and will be a must, not an option. To contribute to the building of a more sustainable environment, computer science students must adopt green IT practices and understand the fundamentals of green software. By successfully greening your software systems, they can take advantage of new opportunities, strengthen themselves to gain a competitive advantage in getting placed in good IT firms, and contribute to the establishment of a sustainable environment that benefits both current and future generations.

Due to the constantly expanding demand for energy efficient computing, traditional computer science programmes no longer suffice to teach our students only performance-oriented programming talents and mentality. It is critical to encourage pupils to "think green" and write more eco-friendly code. Unfortunately, the existing Computer Science curricula of technical education do not place enough focus on green software. There are a number of key problems and issues that hinder the general adoption of green software, according to the literature research [4, 5, 6, 7, 8, 11, 12, 14]. This section highlights the primary obstacles and issues, as well as some current remedies and proposals.

A number of studies in the literature review [1, 2, 3, 10,13] discovered that we need to inform, educate, and expand students' awareness about the possibilities of green software development. To advance and promote green software, technical institution students must be educated and gain awareness in a variety of areas, including environmental impact assessment, standards and laws, and applying software for environmental sustainability. The Green Software Foundation's Software Carbon Intensity definition, for example, is a step in the right direction. Now that the time has come, the student should be able to instrument, monitor, and measure the energy consumption of software systems, as well as become more educated about how programming strategies effect energy utilization.

To survive in the software industry and help nature maintain its greener view, students should be able to identify and address new research problems in green computing, comprehend the impact of various programming approaches on software energy efficiency, and understand the state-of-the-art research and best practices of industry in green computing. Another issue raised in several studies [6, 14] is that it is impossible to teach students how to design green software if they cannot measure the energy consumption of software. The findings of the literature review [1, 4, 6] revealed that the emphasis on environmental sustainability in software development is currently insufficient. Eco-friendly methods include reducing CPU consumption and turning off idle apps. The findings also reveal constraints in terms of accountability, demands, and knowledge. Setting priorities and goals for stakeholders and organizations. As a result, user-friendly tools capable of providing exact and detailed power measurements are critical for educating about green computing and green software. The lack of such tools and low-cost infrastructure that can enable lab experimentation in green software classes is one of the key challenges stopping educators from integrating green computing/software themes into the Computer science curriculum. As power utilization grows in importance, most vendors now provide power monitoring APIs, such as Intel's Running Average Power Limit (RAPL) [12, 14] and Nvidia's Management Library (NVML) [5]. These APIs enable programs running on desktop workstations or servers to monitor the real-time power consumption of the CPU, DRAM, and GPU. Because power data may only be accessible via machine specific registers (MSRs), some students may be unable to log in to a Linux OS unless they have special permission (especially undergraduate students). The Green Code cloud programming site [1, 2] provides an efficient solution by allowing users to submit code from anywhere at any time using a web browser, analyze their energy efficiency, and share their energy-efficient programs with the world. It provides free access to a "virtual lab" on the cloud for teachers and students, significantly reducing the time and money required to set up comparable power measurement equipment on their own.

The major factor that affects the student to write green code is their lack of skill about the coding techniques. Therefore, from the very beginning of software development, students should consider and take steps to reduce the negative effects of their work on the environment. The general guidelines listed below will assist students in developing green software are listed below:

- Pay attention to and manage power-hungry features and frequent usage scenarios.
- Limit the data usage by compressing and aggregate the data, use smaller sizes for media and images where possible, minimize data interchange, and manage the lifecycle of stored data. Adopt an effective cache policy.
- Eliminate or redesign unnecessary features. This increases software maintenance and energy efficiency.
- Find and delete energy-wasting loops that can't complete their specified task, such as probing an unavailable server.
- Adjust the behavior of the program to the power mode of the device or other operating circumstances.
- Keep an eye on the application's real-time energy usage to spot any modules that can be adjusted to use less energy and emit fewer emissions.

- Tools such as dynamic code analysis should be used to check real-time power consumption during development. Understanding the differences between design options and actual energy profiles will rely greatly on the data collected. Students now have tools and information to help them manage their energy consumption. For example, Intel's Software Development Assistant enables developers to collect energy measurements from the system while running various workloads within their application to verify its efficiency.

The programming language student pick can even have an impact on the software's energy efficiency. Many studies in past supported that changing language preference according to the work or task have better result in maintaining sustainability. They kept track of each device's speed and memory utilization as well as its electrical usage. They came to the conclusion that there are many aspects to take into account and that no one language is the best under all applicable criteria, as was noted in a brief article.

More user-friendly power measuring tools and supporting lab equipment are required to better facilitate the incorporation of green modules into software engineering. Having a long-term plan to finance and support the development of such infrastructure and services, as well as making them freely accessible to researchers, educators, and students in the green computing community, is advantageous and urgent.

Together, the software industry, software developers, engineers, experts, educators, researchers, and users may make a significant impact on the environment and contribute to its sustainability for the benefit of present-day and future generations.

4 CONCLUSION AND ENGAGEMENT

Finally, in this rapidly changing era of AI, supercomputing, cloud computing, big data, IoT, and edge computing, we are confident that "green thinking" and broad use of green software in software engineering will significantly benefit our planet, society, and students. Unfortunately, the existing software engineering process lacks educational tools, new education, and creative thinking in green computing and green software design. In this paper, we provide survey data on green software education, emphasize main barriers in teaching green software development, and offer some solutions to major issues that may benefit over the traditional software development process.

Green software integration, on the other hand, necessitates a continual community effort. This necessitates collaboration among researchers, industry leaders, and educators, as well as ongoing guidance on various courses to incorporate cutting-edge understanding and best practices of green software design. We sincerely ask all industry and academic partners to help us boost the attraction of green software to student's at all academic levels and speed the creation of a community that values software energy efficiency and fosters green software design.

REFERENCES

[1] M. Mohankumar, and M. Anandkumar, "Empirical Study on Green and Sustainable Software", Advances in Software Engineering and Systems, ISBN: 978-1-61804-277-4.

[2] S.Tiwari, "Need of Green Computing Measures for Indian IT Industry", Journal of Energy Technologies and Policy, ISSN 2225-0573, Vol.1, No.4, 2011.

[3] R. Abdullah, S. Abdullah, and J. Din, "A Systematic Literature Review Of Green Software Development In Collaborative Knowledge Management Environment", International Journal of Advanced Computer Technology (IJACT), ISSN:2319-7900, 2015.

[4] P. Lago, "Challenges and Opportunities for Sustainable Software", IEEE 5th International Workshop on Product Line Approaches in Software Engineering, 2015

[5] M. Patel, "Green ICT: A Study of Awareness, Attitude and Adoption among IT/Computer Engineering Students of LDRP-ITR, Gandhinagar", 11th International CALIBER-2017.

[6] D. Torre, G. Procaccianti, D. Fucci, S. Lutovac, and G. Scanniello, "On the Presence of Green and Sustainable Software Engineering in Higher Education Curricula", In Proceedings of the 1st International Workshop on Software Engineering Curricula for Millennials (Buenos Aires, Argentina) (SECM '17), IEEE Press, pp. 54–60, 2017.

https://doi.org/10.1109/SECM.2017.42017

[7] D. Gil, J. L. Fernández-Alemán, and J. Trujillo, "The Effect of Green Software: A Study of Impact Factors on the Correctness of Software", Sustainability 2018, 10, 3471; doi:10.3390/su10103471.

[8] M. V. Palacin-Silva, A. Seffah, J. Porras, "Infusing sustainability into software engineering education: Lessons learned from capstone projects", Journal of Cleaner Production, Vol. 172, pp. 4338-4347, 20 January 2018.

[9] Y. Chamekha, and M.A. Hammamib, "Impact of Virtual Reality on Modern Education", International Journal of Sciences: Basic and Applied Research (IJSBAR),Vol. 50, No 2, pp 1-8, 2020.

[10] C. Calero, J. Mancebo , F. Garc ıa, M. A. Moraga, J. Alberto G. Berna, J. L. Fernandez-Aleman, and A. Toval, "5Ws of Green and Sustainable Software", Tsinghua Science and Technology ISSN1007-0214 08/11, V ol . 2 5, No. 3, pp. 401–414, june 2020 , doi: 1 0. 2 6 5 9 9/t st. 2 0 1 9. 9 0 1 0 0 0 6.

[11] A. Iatsyshyn, A. Iatsyshyn, V. Artemchuk, I. Kameneva, V. Kovach, and O. Popov(2020), "Software tools for tasks of sustainable development of environmental problems: peculiarities of programming and implementation in the specialists' preparation", The International Conference on Sustainable Futures: Environmental, Technological, Social and Economic Matters (ICSF 2020), E3S Web Conf, Vol. 166, 2020.

[12] J. Saraiva, Z. Zong, and Rui Pereira, "Bringing Green Software to Computer Science Curriculum: Perspectives from Researchers and Educators", ITICSE 2021, June 26–July 1, 2021, Virtual Event, Germany.

[13] A.Q. Mohabuth, "A framework for the implementation of green computing in Universities", 5th International Conference on Energy Conservation and Efficiency,IEEE, 2022.

[14] R. Mehra, V.S. Sharma, V. Kaulgud, S. Podder, and A.P. Burden, "Towards a Green Quotient for Software Projects", Conference'17, July 2017, Washington, DC, USA.

[15] Podder, S., A. Burden, S. Kumar Singh, and R. Maruca: "How Green Is Your Software?". Harvard Business Review, September 2020, https://hbr.org/2020/09/how-green-is-your-software . Accessed: 1. Aug. 2021.

[16] G. Pinto and F. Castor, "Energy efficiency: a new concern forapplication software developers", *Commun.*, Vol 12 , pp. 68-75, 2017.

[17] S. Hasan, Z. King, M. Hafiz, M. Sayagh, B. Adams, and A. Hindle, "Energy profiles of java collections classes", In Proceedings of the 38th International Conference on Software Engineering, pp 225–236, 2016.

[18] M. Linares-Vásquez, G. Bavota, C. Bernal-Cárdenas, R. Oliveto, M. Di Penta, and D. Poshyvanyk, "Mining energy-greedy API usage patterns in Android apps: an empirical study", In Proceedings of the 11th Working Conference on Mining Software Repositories, pp. 2–11, 2014.

[19] Infotech, "2021 Tech Trends Report – Trend 6: Digital Sustainability, Technical report",2021.

URL: https://www.infotech.com/research/2021-tech-trends-report-trend6-digital-sustainability

[20] R. Pereira, M. Couto, J. Saraiva, J. Cunha, and J. P. Fernandes, "The Influence of the Java Collection Framework on Overall Energy Consumption", In Proceedings of the 5th International Workshop on Green and Sustainable Software (GREENS '16)", pp. 15–21, 2016.

[21] R. Pereira, P. Simão, J. Cunha, and J. Saraiva, "jStanley: Placing a Green Thumb on Java Collections", In Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering (Montpellier, France) (ASE 2018), pp. 856–859, 2018, New York, NY, USA. https://doi.org/10.1145/3238147.3240473

[22] R. Pereira, M. Couto, F. Ribeiro, R. Rua, J. Cunha, J. P. Fernandes, and J. Saraiva, "Energy Efficiency Across Programming Languages: How Do Energy, Time, and Memory Relate?", In Proceedings of the 10th ACM SIGPLAN International Conference on Software Language Engineering (Vancouver, BC, Canada) (SLE 2017), ACM, pp 256-267, 2017, New York, NY, USA. https://doi.org/10.1145/3136014.3136031.

[23] M. Couto, J. Saraiva, and J. Paulo Fernande, "Energy Refactorings for Android in the Large and in the Wild", IEEE 27th International Conference on Software Analysis, Evolution and Reengineering (SANER), pp. 217–228, 2020.

[24] L. Cruz and Rui Abreu, "Performance-based Guidelines for Energy Efficient Mobile Applications", In Proceedings of the 4th International Conference on Mobile Software Engineering and Systems (Buenos Aires, Argentina) (MOBILESoft '17). IEEE Press, pp. 46-57, 2017, Piscataway, NJ, USA,. https://doi.org/10.1109/MOBILESoft. 2017.19

[25] D. Li and W. G. J. Halfond, "An Investigation into Energy-saving Programming Practices for Android Smartphone App Development", In Proceedings of the 3rd International Workshop on Green and Sustainable Software (Hyderabad, India) (GREENS 2014), pp. 46-53, 2014, New York, NY, USA

https: //doi.org/10.1145/2593743.2593750.

[26] S. Maleki, C. Fu, A. Banotra, and Z. Zong, Understanding the Impact of Object Oriented Programming and Design Patterns on Energy Efficiency", In 2017 International Workshop on Sustainability in Multi-Many-Core Systems in conjunction with International Green and Sustainable Computing Conference. IEEE, 2017.

[27] R. Morales, R. Saborido, F. Khomh, F. Chicano, and G. Antoniol, "EARMO: An Energy-Aware Refactoring Approach for Mobile Apps", IEEE Transactions on Software Engineering 44, Vol. 12, pp. 1176–1206, 2018.

[28] S. Abdulsalam, Z. Zong, Q. Gu, and M. Qiu, "Using the Greenup, Powerup, and Speedup metrics to evaluate software energy efficiency", In Proceedings of the 6th International Green and Sustainable Computing Conference, IEEE, pp. 1–8, 2015.

[29] B. Ford and Z. Zong, " PortAuthority: Integrating Energy Efficiency Analysis into Cross-Platform Development Cycles via Dynamic Program Analysis", Journal of Sustainable Computing, 2021.

[30] B. Everman and Z. Zong, "GreenWeb:Hosting High-Load Websites Using Low-Power Servers", In 2018 International Green and Sustainable Computing Conference (IGSC'18), 2018.

[31] B. Everman, N. Rajendrana, X. Li, and Z. Zong, "Improving the Cost Efficiency of Large-scale Cloud Systems Running Hybrid Workloads - A Case Study of Alibaba Cluster Traces", Journal of Sustainable Computing, 2021.

[32] Z. Zong, R. Ge, and Q. Gu, "Marcher: A Heterogeneous System Supporting Energy-Aware High Performance Computing and Big Data Analytics", Journal of Big Data Research, Vol. 8, pp. 27-38, 2017.

[33] K. Gai, M. Qiu, H. Zhao, L. Tao, and Z. Zong, "Dynamic energy-aware cloudlet-based mobile cloud computing model for green computing, Journal of Network and Computer Applications, Vol. 59, pp. 46-54, 2016.

[34] R. Rua, M. Couto, and J.Saraiva, "GreenSource: A Large-Scale Collection of Android Code, Tests and Energy Metrics", In 2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR), pp. 176–180, 2019.

[35] M. Qiu, Z. Ming, J. Li, K. Gai, and Z. Zong, "PhaseChange Memory Optimization for Green Cloud with Genetic Algorithm", IEEE Trans. Comput., Vol. 64,pp. 3528 – 3540, 2015.

[36] D. Mahajan, C. Blakeney, and Z. Zong, "Improving the Energy Efficiency of Relational and NoSQL Databases via Query Optimizations", Journal of Sustainable Computing, Vol. 22, pp. 120-133, 2019.

[37] I. Manotas, C. Bird, R. Zhang, D. Shepherd, C. Jaspan, C. Sadowski, L. Pollock, and J. Clause, "An Empirical Study of Practitioners' Perspectives on Green Software Engineering", In Proceedings of the 38th International Conference on Software Engineering (Austin, Texas) (ICSE '16),pp. 237–248, Association for Computing Machinery, New York, NY, USA,

https://doi.org/10.1145/2884781.2884810

[38] C. Pang, A. Hindle, B. Adams, and A. E. Hassan, "What Do Programmers Know about Software Energy Consumption?", IEEE Software 33, Vol. 3, pp. 83-89, 2016.

[39] G. Pinto, F. Castor, and Y. D. Liu, "Mining questions about software energy consumption", In Proceedings of the 11th Working Conference on Mining Software Repositories,pp. 22–31, 2014.

[40] Y. Cai, Integrating Sustainability into Undergraduate Computing Education", In Proceedings of the 41st ACM Technical Symposium on Computer Science Education (Milwaukee, Wisconsin, USA) (SIGCSE '10, Association for Computing Machinery, pp. 524–528, 2010, New York, NY, USA.

https://doi.org/10.1145/1734263.173443.