GREEN SOFTWARE DEVELOPMENT LIFE CYCLE : POWER CONSUMPTION METRICS FOR DEVELOPERS

P.S. Felix [1],M. Mohankumar[2]*

Abstract

Application Development, even with the most energy-efficient hardware, can invariably waste a great deal of energy. Programmers can avoid high energy consumption by regulating the amount of energy used during the [5] development process rather than wait till the deployment stage. However, this regulation is greatly stalled by an enormous conceptual gap in hardware application, development abstractions, and high-level programming languages. GSDLC entails analysing statistics of the program's energy consumption, right from its programming language and logic through its energy model. GSDLC aims to allot energy values to each level of the programming process, be it at the machine instruction level or source code level.

In development, programmers use tools for code analysis and debugging. The tools determine the performance and vulnerability of the developed software. As there is Power Management Unit (PMU), an integrated circuit that regulates the power functions of different digital devices, a Power Reading Unit (PRU) can be installed in PMU to measure the energy consumed during development. The generated reading can be used for monitoring and optimizing the energy consumption of a running application with different Diagnostic Metrics Tools. With the help of the collected readings, the energy efficiency of the software can be evaluated.

Keywords : Green Software Development Life Cycle, SDLC, Green Environment, software engineering, Measuring Power, Green IT, GREENSOFT model

I. INTRODUCTION

The advent of Green IT measures has elevated the standards of the Information and Communication Technology (ICT) requirements [6]. While the ICT requirements typically focus on the energy consumptions of only ICT devices, the Green IT measures that predominantly deal with computer hardware establish an understanding that "Evolutionary application software that enables energy-efficient system must be the most strategic aspect of eco-friendly software." The following research paper provides an insight to create and implement metrics and approaches in order to evaluate and regulate software energy consumption [17]. The system will support software developers to continually monitor power consumption during the development stages and take sufficient measures to make the software more energy-efficient. The derived process will systematically harmonize with GREENSOFT Model [9] (Fig 1), a GSDLC derived prototype for Greener Software.

II. GREEN SOFTWARE APPLICATION DEVELOPMENT LIFE CYCLE

Software Application Development Life Cycle (SADLC) illustrates the lifecycle of software platforms and their correlation with various phases of the effects of energy consumption. The model is loosely based on the differentiation effect of Berkhout et al., who distinguish between three stages of effects - first-order effects (products' direct impact, eg. energy consumption), second-order effects (impact from the usage, eg. software dematerialization), and third-order effects (eg. when a power-efficientsoftware increases total energy consumption).

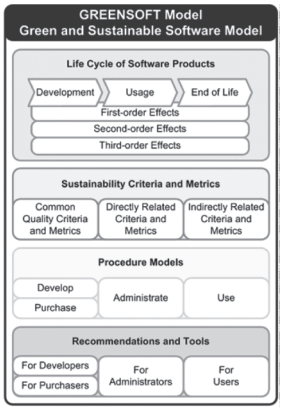

Figure 1. The GREENSOFT Model [9]

In Fig. 2, we can find the differences between the 3 stages of lifecycle – development, usage, and end of life.

[1,2]Department of Computer Science,
Karpagam Academy of Higher Education, Coimbatore, Tamil Nadu, India
* Corresponding Author

Naumann et al. developed the GREENSOFT Model to determine the aforementioned facets of Green and Sustainable Software. The prototype, relatively hypothetical, is developed with an aim to provide support to programmers, designers, DevOps team, deployment team, and software consumers in developing, preserving, and ecologically utilising software applications. The derived example offers a holistic lifecycle prototype for software applications, sustainability standard of measurement, guidelines for software applications, process models to stakeholders, and action-reasoning and recommendations, besides a set of tools that help participants to develop, implement, buy, and use software platforms in a greener and sustainable way.

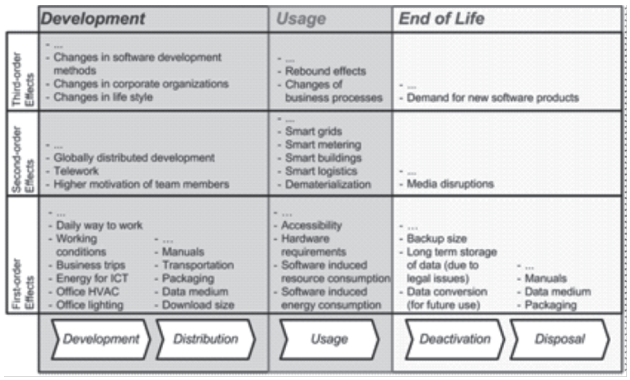

Figure 2: Product life cycle for software products [8]

III. ENERGY EFFICIENCY IN THE CONTEXT OF SDLC

Concerning [4] Fig. 3, the Requirement Stage explains the different needs and factors that arise during the software project or system development when energy efficiency is mandatory. The Design Stage recognizes best practices for solving recurring and common problems with the help of software design patterns [12] during the application development, which can greatly influence power consumption. The Implementation Stage binds efficient methods that developers and users can implement to improve power consumption during the software development process. The Verification Stage monitors the software's energy consumption with support from tools and metrics [18] before the software installed.

The Maintenance Stage applies different code-refactoring tools, techniques, and practices on an installed software application to reduce [16] consumption of energy without changing its external behaviour.

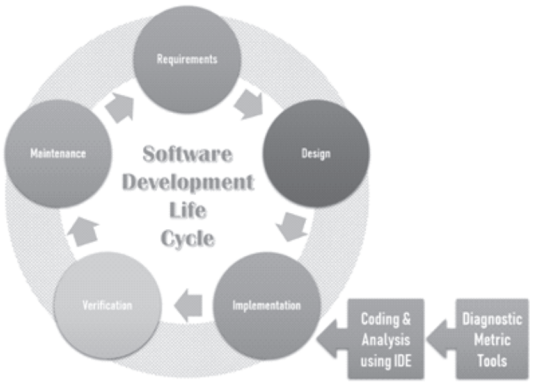

Figure 3: SDLC and Development Phase Metrics

IV. LITERATURE SURVEY

Green and Sustainable development study was initially discussed in the World Convention Environment and Development, 1987[8]. At the convention, Green Software was identified as the "process of development that fulfils the requirements of the present without compromising the capability of the generation ahead to meet their requirements." Besides posing as a research model, the convention discussed development criteria for software solution providers across the world. The initial stages of research concentrated on reducing raw material or waste to promote secure and ecological software. However, the concept of Green Software became increasingly complex at each stage.

Green IT [21] covers the hardware sustainability aspects of DataCenters, and many publications have published several articles highlighting them. Code of Conduct for DataCenter efficiency is one of them. The Quality aspect points out that commoditization has pressurized the software development industry to develop products of a high standard and offer services in greater volume with a limited budget and time.

Software testing is an essential part of quality aspect, and when it comes to that, it is observed that there is a huge efficiency gap between the codes delivered by experienced developers and those developed by automatic interpreters. Furthermore, it is also noted that software engineers play a significant role in depleting the power consumption of applications as opposed to automated solutions. These findings outline the importance of Green quality classifications using Green metrics.

In the proposed software security considerations between 1980 and 2010, the developers primarily focused on software requirements and development processes. This included secure development factors, such as secure advances in software engineering and system computations, piracy prevention architecture, design, development of secure systems, and system verifications. Between 1990 and 2010, developers leveraged technology for design safety and reliability, testing at different levels of development, software safety analysis and hazard analysis, certifications and standard resources, and other safety considerations.

Software safety was an emergent topic [7] in software sustainability considerations in 2010. It surfaced when there were interactions between system components and the environment. Supporting sustainability requires identifying the key challenges (e.g. stakeholders) and success factors [13] for all projects.

Commendable efforts have been put forward to fight pollution problems. It is more prominent in the European Union (EU). The EU Cap limits the total amount of emission in each country, which can be exchanged in the form of emission permits. While each installation can be accredited, it must be within the Cap. If the installation is deemed to exceed the cap or emit more gas, the company has to obtain a credit against a substantial payment. This way, manufacturers will be able to monitor carbon footprint, environmental footprint and hazardous waste more effectively.

Similarly, software development industries must also use this approach to reduce energy consumption and overall environmental impact in data centers.

Efficient software energy consumption has become an important global research subject in 2020. For instance, Europe 2020 strategy, proposed by the European commission, targets three key areas that include:
1. 20% increase in power efficiency
2. 20% reduction in greenhouse gas emissions
3. Boost share of renewable energy up to 20%.

Developed and developing countries must focus on the aforementioned key areas for sustainable software growth. Design and development of software play a crucial role while creating the third-order effect or the rebound effect. Software engineers must aim at increasing processing power along with storage capacity at a proposed rate.

With reference to the EU information society strategies, the ICT (Information and Communication Technology) [7] application is one of its strongest pillars that aims to raise environmental care, offer eco-friendly development, and boost the quality of life.

Surprisingly, despite the stringent measures, one-third of the European organizations do not follow Green IT approaches, while approximately 20% of the firms implement energy consumption regulatory measures regularly. The key reason is lack of official legislation that administers Green IT regulations in those countries.

The paper studied the consumption of energy by various Management Information System (MIS), Customer Relation Management System (CRM), Enterprise Resource Planning System (ERP), and Database Management System. The results are given below:

(i) MIS application layer evidently impacts consumption of energy by up to 70%, besides the infrastructural layer
(ii) MIS applications fulfil the common prerequisites to observe various levels of consumption of energy. The variance measures approximately 145%.
(iii) Certain circumstances that show improvement in time performance do not boost energy efficiency.

V. DEVELOPMENT: MONITORING ENERGY EFFICIENCY WITH METRICS

The best process to promote understanding among developers and users about Green and Sustainable Software is by providing comparable results. Unlike other references that only talk about various software development life-cycle approaches in theory [2], we strive to deliver platforms that can be incorporated in the already established development processes enabling designers to create energy-efficient software solutions every time. According to Anwar et al., existing Green Software engineering research and development do not have support tools. They further state that the biggest research gap is in the lack of energy consumption evaluation tools that are required during project development to help produce eco-friendly software products, e.g. “power-awareness testing approaches”. At present,

developers use static research and energy profiling to deal with energy problems at the time of developing the software product. Mahmoud et al. also mention how software application development consumes energy and resource with a theoretical approach, and by taking into account a comprehensive view. On the other hand, Shenoy et al. provide general guidelines and recommendations with regard to sustainable and green software development.

However, we deliver a more practical and measurable approach and application, as opposed to both their theories.

Mahaux et al. and Becker et al., took a step back to define the software sustainability objective by emphasizing software engineering and sustainable application software designing.

However, we propose continuous incorporation of energy efficiency methods in the development phase to monitor the effect of software applications on the ecosystem and to ensure Greener IT requirements in development.

These tools can either be implemented at the development stage or used as a stand-alone measurement platform [14]. Anwar et al. say that it is likely to retrieve actionable findings to "make significant trade-offs between power usage efficiency and other quality parameters of development lifecycle".

Eventually, continuous assessment and knowledge of the energy consumption by software will guide developers to gain a natural understanding of their software products' energy values. This will further help programmers manage environmental issues with regard to Green IT more efficiently. Additionally, measuring energy efficiency during the project development phase as a continuous task, coupled with effective software testing methodology and CI/CD (Continuous Iteration & Continuous Deployment) will improve software developers' capabilities in producing energy-efficient products.

That being said, existing working structures can effectively adapt our approach to efficient energy measurements to fulfil the needs of practitioners.

The measurement method is effectively one of the very few possible tools in the Green Software engineering segment, and therefore, comprises part four of our [11] GREENSOFT Model (Fig. 1). Moreover, calculating the energy consumption during software development could also become an important point of reach for different stakeholders, including users, programmers, buyers, marketers, developers, etc.

Fig. 4 below shows a usage graph over a period – derived from the data collected from the hardware [1], such as Memory Disk, CPU, Network, and GPU – which will help users to decipher the energy consumption of the hardware components.

Likewise, when the PMU provides the additional functionality of reading the current energy consumption will help in adding that feature to the Operating System itself. This will also help developers and stakeholders solve the problem of comparing the existing software's energy consumption. Additionally, the process will help in the Black Box Testing of Energy efficiency and White Box testing of Energy efficiency.

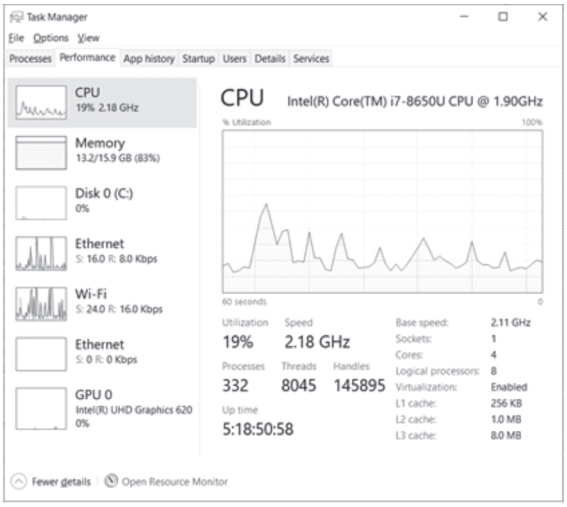

Figure 4. Task Manager of Windows 10

VI. SOURCE CODE ANALYSIS

Source code analysis identifies troubleshoots and vulnerabilities [3] in a computer program before it enters the deployment stage. The following segment discusses different works on dynamic source code analysis that categorizes energy-related bugs and hotspots by evaluating computer programs in real-time. Here, a Diagnostic Metrics Tool is used. The tool, which can be found in IDEs, analyses the code to provide source code analysis [19] guidelines ahead of its execution. Fig. 5 reveals that the Diagnostic

Metrics Tool displays the CPU usage graph and Memory Usage graph. This can indicate hardware utilization during the operation stage. Likewise, when the PRU generates power consumption reading, the diagnostic tool can project it via a graphical representation in real-time. So, when the developer reviews the code, he will be able to identify the detailed functionality with regard to the energy efficiency aspect.

VII. DEVELOPMENT POINT OF VIEW

The programming approach differs from developer to developer – experienced to fresh [7]. While experienced developers write the code in an optimized manner, the new developers create their style of coding, which might not be the right coding approach. Therefore, we have to consider a unified solution based on different coding styles. The procedure is established in three facets - system, function, and time.

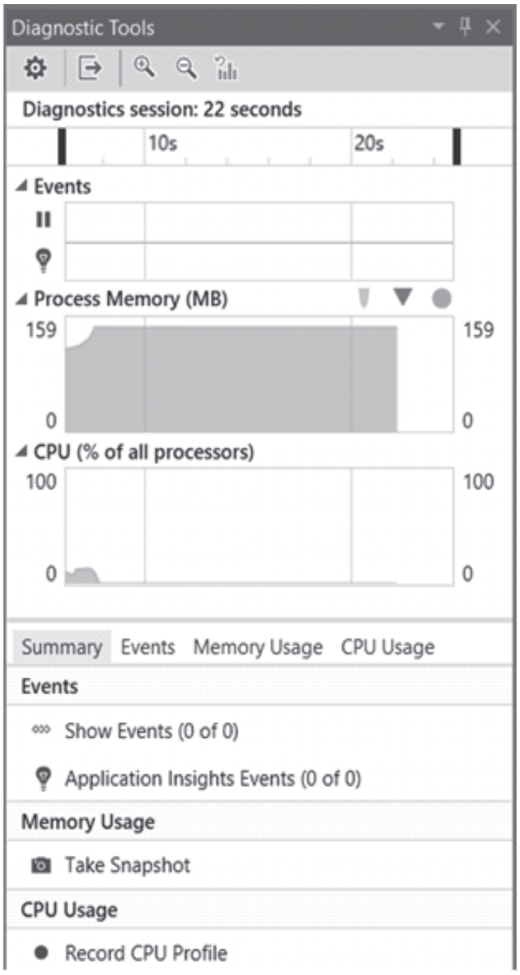

Figure 5. Diagnostic Metrics Tool of Visual Studio

The system refers to a software development provider. The function denotes the reduced ecological impact with proper economical balance, while the time horizon is derived from the size of the company and the median period of the projects. During the development stage, the provider has to remain focused on power efficiency, consumption, and development of the program. The advent of modern digital paradigms and increased usage of electronic devices have heavily impacted the total energy consumption [20].

VIII. DEVELOPMENT

Coding or development is the most important stage of the SDLC [15]. The coding style generally differs from programmer to programmer. But if a company wants to receive 3 Stars for its development methodology, then it has to write optimized code using the Green and Sustainability approach. 3 Stars are provided to companies that maintain clean coding. Irrespective of the position or experience, every programmer has to follow the derived code at par with GSDLC. The approach matches a defined process of the CMM level, in which the new coders also follow the sustainability software development procedure by default

IX. CONCLUSION AND FUTURE WORK

In this time and era, where global warming and pollution are at their peak, the only way out is environment preservation using Green approaches. Global researches are continually progressing to reduce power consumption in various industrial sectors. While many challenging areas have surfaced, a few areas are still kept hidden from the world. These areas require more stringent measures to improve efficiency. And Software development is one of them. Since software development is not a physical component, it becomes tougher to achieve sustainability in this industry.

The study reveals how we can spread awareness of Green IT and the GREENSOFT Model among the IT companies and software developers. Earlier research suggests that energy efficiency should be performed at the development and testing stage in a way that is similar to performance and vulnerability testing. However, if we compare the iteration cost to implement energy-efficient application between the deployment stage and development stage, the cost doubles in the former. Even if a power consumption tracking hardware [10] is used to test as Black Box testing, it would not reduce the cost.

At the development stage of an SDLC, developers use IDEs to create code. IDEs have advanced features and tools, such as Diagnostic Metric Tools and IntelliSense that help programmers monitor code in real-time and make informed decisions to establish suitable solutions to improve software sustainability at the initial stage rather than perform it as a stand-alone activity during the testing and delivery stage.

Diagnostic Metric Tools takes data from [22] CPU, Disk, Network, GPU, and memory hardware. However, without the PRU installed in it, one cannot gauge the code's energy consumption. The article helps researchers develop a cost-effective PRU (Power Reading Unit) to be installed in the PMU (Power Management Unit) so that it is available in all computers by default without any additional cost to the company.

Future work will be to identify the pros and cons of PRU and its benefits and advantages during the SDLC process.

REFERENCES

[1] Mohankumar Muthu; K. Banuroopa; S. Arunadevi, "Green and Sustainability in Software Development Lifecycle Process" Electromechanical Devices and Machines, IntechOpen, Nov 2019

[2] Eva Kern; Achim Guldner; Stefan Naumann, "Including Software Aspects in Green IT: How to Create Awareness for Green Software Issues" Green IT Engineering: Social, Business and Industrial Applications, Studies in Systems, Decision and Control 171, Springer Nature Switzerland AG 2019, Jan 2019

[3] Georgiou, Stefanos&Rizou, Stamatia&Spinellis, Diomidis. (2019). Software Development Lifecycle for Energy Efficiency: Techniques and Tools. ACM Computing Surveys. 52. 1-33. 10.1145/3337773.

[4] Naumann, Stefan & Kern, Eva & Dick, Markus. (2014). Classifying Green Software Engineering - The GREENSOFT Model. Softwaretechnik-Trends. 33. 18-19. 10.1007/s40568-013-0027-z.

[5] Kerstin Eder and John P. Gallagher (March 22nd 2017). Energy-Aware Software Engineering, ICT - Energy Concepts for Energy Efficiency and Sustainability, GiorgosFagas, Luca Gammaitoni, John P. Gallagher and Douglas J. Paul, IntechOpen, DOI: 10.5772/65985

[6] T. Johann, M. Dick, S. Naumann and E. Kern, "How to measure energy-efficiency of software: Metrics and measurement results," 2012 First International Workshop on Green and Sustainable Software (GREENS), Zurich, 2012, pp. 51-54., IEEE, Jun 2012

[7] M. Mohankumar, Dr.M.Anand Kumar, "A Green IT Star Model Approach For Software Development Life Cycle" International Journal of Advanced Technology in Engineering and Science Volume No 03, Special Issue No. 01, Mar 2015

[8] Stefan Naumann, Markus Dick, Eva Kern, Timo Johann, The GREENSOFT Model: A reference model forgreen and sustainable software and its engineering, Sustainable Computing: Informatics and Systems,Volume 1, Issue 4, December 2011, Pages 294-304, ISSN 2210-5379

[9] M.Mohankumar; M.Anand Kumar, "Green based Software Development Life Cycle Model for Software Engineering" Indian Journal of Science and Technology, Vol 9(32), ISSN (Online) : 0974-5645, Aug 2016

[10] T. Debbarma and K. Chandrasekaran, "Green measurement metrics towards a sustainable software: A systematic literature review," 2016 International Conference on Recent Advances and Innovations in Engineering (ICRAIE), Jaipur, 2016, pp. 1-7, doi: 10.1109/ICRAIE.2016.7939521.

[11] M. Dick, J. Drangmeister, E. Kern and S. Naumann, "Green software engineering with agile methods," 2013 2nd International Workshop on Green and Sustainable Software (GREENS), San Francisco, CA, 2013, pp. 78-85., IEEE, Sep 2013

[12] M.Mohankumar; Dr.M.Anand Kumar, "Empirical Study on Green and Sustainable Software Engineering" Advances in Software Engineering and Systems ISBN: 978-1-61804-277-4, Nov 2017

[13] M. Salam and S. U. Khan, "Developing green and sustainable software: Success factors for vendors," 2016 7th IEEE International Conference on Software Engineering and

Service Science (ICSESS), Beijing, 2016, pp. 1059-1062., IEEE, Mar 2017

[14] A. Hindle, "Green Software Engineering: The Curse of Methodology," 2016 IEEE 23rd International Conference on Software Analysis, Evolution, and Reengineering (SANER), Suita, 2016, pp. 46-55., May 2016

[15] S. K. Sharma, P. K. Gupta and R. Malekian, "Energy efficient software development life cycle - An approach towards smart computing," 2015 IEEE International Conference on Computer Graphics, Vision and Information Security (CGVIS), Bhubaneswar, 2015, pp. 1-5, doi: 10.1109/CGVIS.2015.7449881.

[16] Pinto, Gustavo & Castor, Fernando. (2017). Energy efficiency: A new concern for application software developers. Communications of the ACM. 60. 68-75. 10.1145/3154384.

[17] Sarker K U; Deraman A; Hasan R, "Green soft computing and standardization" J. Fundam. Appl. Sci., 2018, 10(6S), 462-470., Mar 2018

[18] Alvi, Hamza & Sahar, Hareem & Bangash, Abdul & Beg, Mirza. (2017). EnSights: A tool for energy aware software development. 1-6. 10.1109/ICET.2017.8281713.

[19] M. Rashid, L. Ardito and M. Torchiano, "Energy Consumption Analysis of Algorithms Implementations," 2015 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM), Beijing, 2015, pp. 1-4, Nov 2015

[20] A. A. Bangash, H. Sahar and M. O. Beg, "A Methodology for Relating Software Structure with Energy Consumption," 2017 IEEE 17th International Working Conference on Source Code Analysis and Manipulation (SCAM), Shanghai, 2017, pp. 111-120., Nov 2017

[21] Erdélyi, Krisztina. (2013). Special factors of development of green software supporting eco sustainability. SISY 2013 - IEEE 11th International Symposium on Intelligent Systems and Informatics, Proceedings. 337-340. 10.1109/SISY.2013.6662597.

[22] M. U. Farooq, S. U. Rehman Khan and M. O. Beg, "MELTA: A Method Level Energy Estimation Technique for Android Development," 2019 International Conference on Innovative Computing (ICIC), Lahore, Pakistan, 2019, pp. 1-10., Nov 2019