Journal of Engineering and Technology

ISSN 2180-3811 eISSN 2289-814X https://jet.utem.edu.my/jet/index

GREEN CODING IN PROGRAMMING AND PRACTICES

S. J. Mahmudova*[1]

[1] Department of Software Engineering and Scientific-Theoretical Problems of Intelligent Software Systems, Institute of Information Technology, 9A, B. Vahabzade str., AZ1141 Baku, Azerbaijan.

corresponding: shafagat@gmail.com

Article history:
Received Date: 18 April 2024
Revised Date: 30 September 2024
Accepted Date: 28 November 2024

Keywords: Code Environmental Impact, Green Practice, Programming, Testing

Abstract— This work explores green coding in programming, its principles and use in industry. Green coding principles should not be considered contrary to existing practices. It analyses some of the studies conducted in this field. Best practices on green coding are defined. Software developers can support their sustainability efforts through a number of green coding tactics. These best practices range from minimizing artefacts to improving efficiency. The paper conducts research on the minimization of resources used in testing, testing and deploying updated code, using solar and sustainable energy, environmental impact on the IT sector, programming languages used in green coding, etc. The best practices

mentioned to make the program eco-friendly are reported to be followed.

I. Introduction

Green coding is a term in the field of software engineering to describe a set of methods used to create applications that consume as little energy as possible. Green coding is a relatively recent notion used to refer to programming code that is produced and written in a way that minimizes the software's energy consumption, thereby limiting its potential impact on the environment. Green coding principles should not be considered contrary to existing practices. They should be incorporated into software engineers' principles to consider when writing and projecting code to balance functionality and power consumption [1].

The President of the Republic of Azerbaijan has signed an order declaring 2024 as the "Year of Solidarity for the Green World" in the Republic of Azerbaijan [2]. The telecommunications and ICT industry is a major contributor to carbon emissions on a global scale. These emissions come from a variety of sources, including the energy required to power telecommunications companies' cellular networks, and emissions associated with the manufacturing and distribution of phones, the processing of other digital applications and technologies. These emissions will increase as the use of digital technologies rises.

Although green coding does not solve all the industrial problems (such as the distribution, use and life cycle of handsets), it can be an important lever to reduce energy consumption. Moreover, incorporating green coding principles can further strengthen the drive towards sustainability by ensuring the followings:

i. Software developers can support their sustainability efforts through several green coding tactics. These best practices range from minimizing artifacts to improving efficiency.

ii. Processing, storage, and other activities consume a lot of energy and generate a lot of heat waste. While data center operators may struggle with green initiatives at the physical hardware and facilities level, developers and testers can contribute with green coding [3].
iii. From reduced artifacts to solar-powered coding, it is important to learn green coding techniques for software programming bearing in mind sustainability. To engage teams and initiate change, it is necessary to take advantage of the various advantages of a continuous development approach.

The purpose of the article is to use the practices used in this field to create new software systems and to help create software systems using as little energy as possible by green coding. This research contribution are as follows:

i. This work explores green coding in programming, its principles and use in industry.
ii. Best practices on green coding are defined which range from minimizing artefacts to improving efficiency.
iii. The paper conducts research on the minimization of resources used in testing, testing and deploying updated code, using solar and sustainable energy environmental impact on the IT sector, programming languages used in green coding.
iv. The best practices mentioned to make the program eco-friendly are reported to be followed.

II. Literature Review

Green computing is not only principled and compulsory; it is also a gainful, reasonable and decisive solution. The amount of oil and gases in the environment causes the use of fauna as a power generator, where the environment is contaminated by manufacturing, local surplus, and the cars pose a huge threat to the health of human beings. In this circumstance there is a great necessity for a real tendency to live in a safer environment. The global use of ICT infrastructure

extremely increases power consumption and consequently, it is raising carbon emissions every minute. Thus, this power consumption by ICT should be minimized. The goal of Green IT is to reduce the consumption of power. However, this paradigm lacks models, descriptions, and some realizations in ICT, which is crucial for effective employment of Green Computing. Furthermore, scientists fail to offer any systematic algorithms for the integration of sustainability aspects in the design and development of software products. Sustainability and greener aspects of software are not resolved by the software development process models. [4] attempts to comprehend the sustainability aspects of software and offer some enhancements in the SDLC models to comprise the greener aspects of software development.

In order to lessen the software energy consumption while preserving ongoing functionalities for the software it is necessary to build sustainable and green software. In [5], firstly, the authors highlight several definitions of the terms sustainable and green in software development. It is noted that to build a software product, software engineering process should be followed. Henceforth, the authors define and describe sustainable and green criteria to be esteemed at each step of the process to launch a sustainable and green software engineering process. Moreover, the study emphasizes the estimation of software energy consumption. Many authors attempted to offer tools to estimate energy consumption due to software to minimize carbon footprint.

Researchers estimate the energy efficiency of IT as one of the hottest topics in the last few years. Some hardware manufacturers and designers try to solve the problem, but lately, industry and academia shifted their focus to the role of software for IT sustainability. The most challenging issue in this field is energy-efficient software writing, since it entails not only an alteration of attitude for software developers and

designers but also models and tools to evaluate and reduce the effect of software on the energy consumption. In this regard, [6] presents a conceptual framework to provide a merging view of the strategies, models, and tools available so far for designing and developing greener software.

v. Sustainable development is affected by ICT due to its rising demands for energy and resources required when manufacturing hardware and software products. Many Green ICT/IT attempts so far intended to solve the effects of hardware on the environment, however, it is insufficient at the background of the effects of building software products. Well-organized software will indirectly consume less energy by using up less hardware equipment to run. The contributions of the authors in [7] include building a two-level green software model that covers the sustainable life cycle of a software product and the software tools promoting green and environmentally sustainable software. They offer a new green software engineering process in the first level that is a hybrid process between sequential, iterative, and agile development processes. Each stage of the software process is further explored to generate a green and sustainable stage. They also offer green guidelines or green processes for each software stage in the engineering process. The authors add the requirements stage and the testing stage to the software life cycle. They also include a complete list of metrics to estimate the greenness of each stage in the first level. In the second level, they explain how software itself can be applied as a tool to aid in green computing by monitoring resources in an energy efficient manner. In conclusion, they present and describe relationships between the two levels in the proposed model to make the software engineering process

and product green and sustainable.

III. Green Coding Best Practices

Figure 1 shows the best practices of green coding.

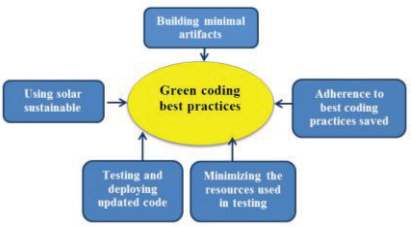

Figure 1: Best practices of Green Coding

A. Construction of minimal artifacts

Anyone who has used a package manager like Npm or HomeBrew knows that installing a large package often requires several smaller packages too. In addition, these little packages have dependencies, and the number of artifacts involved should be limited as much as possible.

B. Operating Systems

Operating systems vary greatly in requirements. For example, Linux Mint, a popular Linux distribution based on Ubuntu and Debian, recommends 100 GB of disk space for smooth operation. Alpine Linux, on the other hand, is a 5 MB image built around musl libc and BusyBox for Docker and Kubernetes container environments. IT organizations can reduce disk, memory and processing requirements by considering software for a specific purpose. To this end, they need to compile a list of dependencies. It is only required to compile the binaries with the necessary dependencies. Cloud-based and virtualized deployments should be considered because multiple virtual servers can run on shared physical hardware.

C. Adherence to Best Coding Practices Saved

How much processing power should programmers use for software? Programmers build software to take advantage of available processing power. This approach estimates how to enable software to run faster when it is released to market or launched into production. Joel Spolsky, the founder of Trello, Stack Overflow, etc., proposed

the idea of bloat ware to be good back in 2001. Computer science covers topics like memory consumption and Big O notation. Big O notation is a way to calculate the amount of processing power used by an algorithm. Big O optimization aims to make the demand grow more slowly as the number of elements in the list or leaves in the tree structure increases. It may be time to take another look at Big O notation. Other green coding techniques may include using lower-resolution photos on the web and moving searches from databases to within-memory caches. Open-source NoSQL databases such as MongoDB, Couchbase, and Redis retrieve and store aggregated data with less processing power than other relational databases.

D. Minimizing Resources in Testing

Although choosing SaaS development and testing tools can be more efficient than installing them to run on servers, cloud applications can still suffer from completion. Modern DevSecOps tools often create full test environments and perform all automated checks on each task. They can also run security scans, code limits, and sophisticated analyzers and use them in the cloud. When a team merges code, they do it all over again. Some systems run with a delay and restart automatically. Monitoring tools to keep track of everything can cause processing fatigue and increased network frequency. For example, let's imagine a scenario where the team activates the observation system for all tests. Each time network traffic occurs, the monitoring system sends a message to the server about what data and where is this data going to, doubling the test traffic. The energy consumed is mostly wasted. In the best case, test servers are slow for small files. In the worst case, production servers are also affected, and outages occur.

E. Testing and Deploying Updated Code

Monolithic systems can require an expensive regression testing process on each build.

Even if fully automated, the build and automated verification process would be largely redundant, repeating what was done before for each build. Micro services offer the ability to modify, test, deploy, and monitor only a portion of the code. Other modern approaches include languages such as PHP, where code can be deployed one web page at a time. There are also tools that get languages like PHP and compile them, resulting in smaller files that run with less CPU and memory usage.

F. Solar and Sustainable Energy

While green coding focuses primarily on software design and operation, developers and testers can make a personal difference to the environment. It's no secret that data processing centers require a lot of energy to operate. According to experts' estimates, the energy used by 7.2 million data processing centers around the world consumes 1% of the world's total electricity [8]. Therefore, data center providers began to spend time and money to improve the efficiency of their sites. Google has set a goal to make 2030 the year when all its data centers run on carbon-free energy. Amazon Web Services aims to run on 100% renewable energy by 2025 [9]. However, the problem is that the number of data centers will increase as all sectors and even countries gradually go digital. While it's good to commit to powering databases with renewable energy, the goal should be using less energy first.

Green coding means using as few processor instructions and minimum memory as possible. Green coding creates algorithms that consume minimal energy. This requires software engineers to pay more attention to the code they write. When working with green coding, analysts consider two types of adjustments.

i. Structural considerations: They refer to the power consumption of code blocks. A block is a unit of code.
ii. Behavioral considerations: They refer to power consumption at user side.

IV. Research Methodology

A. Green Color During Coding

If we go back to the software development processes adopted 20 years ago, in most cases the programming can be immediately green. Lengths and dimensions dictate how long a piece of code can be. It is almost always possible to reduce the sizes of videos, images or text files. A simple compression tool can help reduce file size, speed up navigation, improve user experience, and reduce power consumption.

Website developers should know when to use high-quality media files and consider smaller files that can achieve the same goals. Having fewer large files also reduces download time. It helps businesses in their SEO (Search Engine Optimization) efforts and achieves sustainability. For example, in an application with 500,000 users, by reducing the resolution of images, application developers can save more than two days of application runtime per year.

V. Results and Discussion

A. Environmental Impact on IT Sector

Digitization has resulted in a significant increase in greenhouse gas emissions, which contribute to climate change and global warming. According to the UN report, the technology industry currently accounts for 2-3% of global emissions, and digitization is expected to increase rapidly. The technology industry's contribution to greenhouse gas emissions will grow if not addressed, and enterprise technology alone is responsible for 350-400 megatons of carbon dioxide emissions. In 2021, the UK government launched a Net Zero Strategy to decarbonize all sectors of the economy by 2050. This strategy is important for the technology industry as it is one of the fastest growing and most energy intensive sectors of the UK economy.

Software development has changed people's lives almost significantly. This importance will surely increase in the future. Many programs are large and complex. If so, what is their impact on the environment?

Both computer software and hardware contribute to environmental degradation. Software affects hardware performance. Accordingly, the software decides how much power the hardware will consume. In this way, the software directly affects the computer hardware, which further affects the power consumption. Inefficient programs like code will consume more energy. This makes it a part of the challenges it faces in achieving environmental sustainability.

A study conducted at the University of Cambridge found out that the Bitcoin network requires about 115 terawatt-hours to operate. This amount is equal to twice the energy consumption of all of Switzerland [6]. Figure 2 illustrates the percentage of energy consumption of various software components [6].

In addition, software development itself can be very energy intensive. For example, IT professionals trained an artificial intelligence model to distinguish flowers using a database of light colors. The model achieved 96.17% accuracy while consuming 964 joules of energy. But to increase the accuracy to 1.74%, the consumption of the model is 2815 joules. This means that more computer power and energy must be spent on the problem to get better results than necessary. Researchers have dubbed this wasteful practice "Red AI." Red AI additionally tracks accuracy [10].

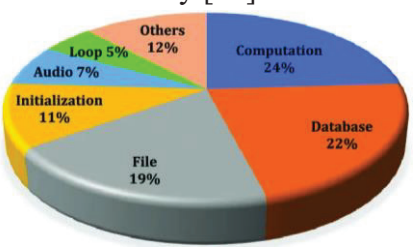

Figure 2: The percentage of energy consumption of various software components

Red AI makes it even more important to create codes and software to optimize power consumption. While software and mobile apps are often ignored when talking about sustainability, it's time to pay more attention to green coding practices. Businesses and industries in the IT sector should adopt green coding and make it

mandatory as part of their efforts to achieve sustainability [11].

The use of the application code will have an almost immediate effect on the application's energy consumption. But there are also steps software companies can take at the training level to green their coding practices. For example, some companies now offer green coding training at all levels. They even make it part of the recruitment and hiring process. Companies should be required to make green coding a profession-wide practice to reduce their environmental impact. By now, pressure is mounting on data center providers to become more resilient. But software engineers must also know and acknowledge their role in reducing coding power consumption.

Green coding should become a necessity, not an option. To create a more sustainable world, software developers and engineers must embrace and recognize the potential of green coding. Adapting code to the environment can help create an environment that benefits current and future generations. Green coding is a relatively new tendency that tries to minimize the energy intensity involved in processing lines of code. As the telecommunications industry works toward net zero goals, green coding can play an important role.

B. Green Coding Application in Telecommunication Industry

The telecommunication and ICT industry is a major contributor to carbon emissions on a global scale. These emissions come from a variety of sources, including the energy required to power telecommunications companies' cellular networks, the manufacturing and distribution of phones, and emissions associated with the processing of other digital applications and technologies. These CO_2 emissions will continue to increase as the use of digital technologies increases. For example, the proportion of greenhouse gas emissions globally attributed to digital

technologies has increased from 2.5% to 1.2% since 2013, and energy demand from data centers has quadrupled over the same period.

Telecommunication operators and other ICT companies have made ambitious commitments towards their net-zero agenda, particularly in Europe, where a significant number of telecommunication companies have announced a 2030 target. However, achieving this goal will not be possible without a concerted effort in all areas including energy-efficient hardware and software. There are many levers the telecommunications industry can and should pull to meet net zero emissions, including using greater automation to reduce the need for (carbon-intensive) human intervention during site visits or greater use of renewable energy sources. However, it is also important to explore the ways to reduce the amount of energy required for energy operations first. This is where green coding comes in. Software programs and applications are instructed by the lines of code written by developers. As each line of code is parsed, the device processing that code causes carbon emissions. The more codes are processed, the higher the emission level. About 90% of software development uses open-source code that is not adapted for specific uses and applications. Therefore, it contains redundant code sections, using more processing power and causing unnecessary carbon emissions.

Green coding principles can be adopted to promote thin coding practices where only a minimal amount of processing is required to deliver the same application or output. Although green coding does not solve all the industry's problems (such as the distribution, use and end of life cycle of handsets), it can be an important lever to reduce energy consumption. In addition, incorporating green coding principles can further drive the move towards greater sustainability by ensuring the green coding training should be offered to new and existing IT engineers within the business. This can raise awareness of the

importance of this practice and encourage continued adoption of these principles. It should ensure that the number of excessive lines of code is minimized, as each lead to carbon emissions, and even minimal reductions will have a large overall impact when implemented globally. Then, the software engineers should be encouraged to use green coding within the organization, thereby encouraging innovation and skill development. A mindset and culture of efficiency across the business should be embedded, making it part of the organization's DNA to integrate sustainability by design across business units. A successful sustainability strategy recognizes its multifaceted nature as sustainability is not just the responsibility of one team [12]. Green coding is a programming approach that aims to create environmentally friendly software. Thus, what's behind it? With the rise of digitalization, software plays a major role in almost every business in sight. However, energy-intensive software plays a significant role in greenhouse gas emissions. One way to combat this problem is through green coding. Green coding is not only about creating energy-efficient software, but also about sustainable software development. This could include using renewable energy to power data centers or cloud systems. Green coding affects every part of the software development process. This includes both software architecture and core design. Green code programs should avoid unoptimized mixed code, so that insufficiently optimized code will execute commands that are not necessary for the program logic. This causes excessive resource usage and energy depletion.

When talking about green coding, the basic principles of green coding should not be ignored. They form the basis of sustainable software. There are four main areas of green coding. Each of these is imperative to the software development process and using them can potentially reduce the amount of energy and resources.

i. **Green architecture:** Software architecture is the first pillar. This is the basic architecture of the application. It can be optimized when it comes to energy usage. For example, the code can be adapted. This means that the equipment is not used as often, which reduces the power it needs. Having a system that automatically shuts down software when not in use is also part of a green architecture.
ii. **Green logic:** Another key factor in green coding is the program logic. Basically, this is the optimization of the program code so that the work of programs does not slow down due to unnecessary use of commands. When it comes to developing efficient and environmentally friendly code, other factors also play a big role, such as choosing environmentally friendly file types or efficient data structures.
iii. **Green method:** This pillar of green coding has little to do with the software code itself, instead the software development process is focused on. Flexible software development models are particularly useful for making code and applications more energy efficient. Gradual development and testing used in agile programming ensure early identification of parts of the program that are not energy efficient. Consequently, they can be modified to make the final product as efficient as possible.
iv. **Green platform:** Green coding is not just software related as this also includes hardware. There are several techniques that can be used to create environmentally friendly systems. Server load is a particularly important factor. Servers running on less load will consume more energy than necessary. Cloud computing can help in this regard, as resources can be flexibly scaled and tailored to individual needs. Ideally, the cloud computing server can also be powered by renewable energy.

C. Energy Efficient Programming Languages

The most energy-efficient programming language is C. C can be used in an effective device. This enables the programmer to use memory efficient strategies and free up memory. Object-oriented C++ based on C is a good option for saving energy resources. Along with C, Rust is another language that is particularly suitable for green coding. The relatively new language controls memory management to ensure efficient access. This language helps to avoid unnecessary and energy consuming memory operations. The wide range of parallelization available in Rust helps it to be energy efficient, providing optimized use of hardware resources. Rust is a multi-paradigm designed general-purpose programming language that combines functional and procedural programming paradigms with a token-based object system.

Despite being one of the most popular programming languages, Python has a particularly low energy efficiency rating. This is partly because Python has an interpreter, not a compiler. This means that there is more than one compilation process. However, dynamic language type JavaScript is not very efficient and less suitable for green coding.

VI. Conclusion

In conclusion, adopting the best practices is crucial for environment friendly program. By creating intelligent software architecture, minimizing unnecessary code, and designing programs with automated shutdown features, developers can significantly improve efficiency. Selecting energy-efficient programming languages and conducting continuous testing throughout the development stages ensures the optimization of resources. Furthermore, reducing data usage and optimizing network operations, such as through caching, contribute to sustainable software practices. These best practices support and make the program eco-friendly.

VII. References

[1] G. Donnelly (20220, "Green coding: What is it and why is it important to the telecoms industry?" *STL Partners*, pp. 1-4, [Online] Available: https://stlpartners.com/wp-content/documents/Articles/Sustainability/2022-08-19%20Green%20coding.pdf

[2] Decree of the President of the Republic of Azerbaijan, "On declaring 2024 as the Year of Solidarity for the Green World in the Republic of Azerbaijan," Dec. 25, 2023.

[3] M. Heusser (2023), "6 green coding best practices and how to get started," *TechTarget*. [Online] Available: https://www.techtarget.com/searchsoftwarequality/tip/Green-coding-best-practices-and-how-to-get-started

[4] H. Acar (2017), *Software development methodology in a Green IT environment*, Université de Lyon [Online]. Available: https://tel.archives-ouvertes.fr/tel-01724069

[5] S. Agarwal, A. Nath, and D. Chowdhury, "Sustainable Approaches and Good Practices in Green Software Engineering," *International Journal of Research and Reviews in Computer Science (IJRRCS)*, vol. 3, no. 1, pp. 1425-1428, 2012.

[6] M. Torchiano and A. Vetrò, "Understanding Green Software Development: A Conceptual Framework," *IT Professional*, vol. 17, no. 1, pp. 44-50, 2015. DOI: 10.1109/MITP.2015.16.

[7] S. S. Mahmoud and I. Ahmad, "A Green Model for Sustainable Software Engineering," *International Journal of Software Engineering and Its Applications*, vol. 7, no. 4, pp. 55-74, 2013.

[8] J. Smith (2023), "The Environmental Impact of Digitalization: What's Your Take on Sustainable Technology?" *FDM Group*. [Online] Available: https://www.fdmgroup.com/news-insights/environmental-impact-of-digitalisation/

[9] P. K. D. Pramanik, N. Sinhababu, B. Mukherjee, and P. Choudhury, "Power Consumption Analysis, Measurement, Management, and Issues: A State-of-the-Art Review of Smartphone Battery and Energy Usage," *IEEE Access*, vol. 7, pp. 182113-182172, 2019.

[10] E. Greenfield (2023), "Green coding: coding for sustainable development," *Sigmaearth*. [Online] Available: https://sigmaearth.com/ru/green-coding/

[11] H. Kim, J. Ben-Othman, and L. Mokdad, "Intelligent Terrestrial and Non-Terrestrial Vehicular Networks with Green AI and Red AI Perspectives," *Sensors*, vol. 23, no. 2, pp. 806, 2023. DOI: 10.3390/s23020806.

[12] "What is green coding and how is it done?" *Digital Guide IONOS*. (2023) [Online] Available: https://www.ionos.com/digitalguide/websites/web-development/green-coding/