Teaching how to engineer greener software

JJ Merelo-Guervós
jmerelo@ugr.es
Department of Computer Engineering, Automatics and Robotics and CITIC, University of Granada
Granada, Spain

ABSTRACT

Green computing is a general term that describes a host of techniques that try to minimize the carbon footprint of software applications. As such, it is not a single body of knowledge, but a series of best practices that help reduce energy consumption relying on the features of any of the different layers that are exercised by software applications. This represents a challenge at the time of designing a comprehensive syllabus that would help students develop the series of skills needed to identify energy bottlenecks and eliminate them. In this poster we will describe the different concepts involved, and how they will be delivered to guarantee the achievement of learning objectives.

CCS CONCEPTS

• **Software and its engineering → Agile software development**.

KEYWORDS

Software engineering, green computing, project-based learning

ACM Reference Format:
JJ Merelo-Guervós. 2024. Teaching how to engineer greener software. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 1 page. https://doi.org/10.1145/nnnnnnn.nnnnnnn

Green computing [2] deals, in general, with reducing the environmental impact of the creation and use of computing resources. From the software perspective, it proposes maximizing the amount of work done for every unit of energy spent. But in order to achieve that, how energy is spent across all the different computing layers need to be assessed, and understood.

This is why getting the student to achieve a certain amount of understanding of the different process involved, methodologies needed to carry out that assessment, and eventually design your code from the ground up or refactoring it to make it *greener* is a challenge.

And it is a challenge that has been recently acknowledged by the joint IEEE/ACM task force in [1]. Environmental concerns is one of the skills mentioned in the draft competencies in software engineering as part of the needed "behavioral attributes" as well as in the Master's degree in Information Systems (IS), as part of the IS Strategy and Governance competence; the computer engineering set of draft competencies includes it as part of the Systems Resource Management subject matter. This again shows the inter-disciplinary content that needs to be considered even if the focus is on software engineering.

The final objective would be to get the student to understand how choices of hardware and software platform from the ground up will affect the environmental impact of the workload that is going to be created or refactored and which best practices need to be involved to reduce that impact. The syllabus proposed would, then, would be as follows:

- **Understanding the hardware**: Computing units (CPU, GPU, memory) and their energy profiles. Energy-wise heterogeneous architectures. Energy consumption sensors and standard APIs to get measurements from them (e.g. RAPL). Interfaces to the computing power configuration and administration system (e.g. ACPI). This will help the student to understand how the workload exercises different parts of the hardware, and why it does so.
- **Understanding how the workload spends energy**: software profiling, methodologies and tools for energy profiling (PowerMeter, pinpoint, hardware meters). This will help the student identify bottlenecks from the point of view of performance as well as energy consumption; also find out how this consumption scales with workload size.
- **Refactor for reduction of energy footprint**: once the bottlenecks have been identified and specific benchmarks developed to measure the energy footprint, the student needs to work across the board to reduce the footprint: choosing the computing platform where possible, configuring the application to work on specific computing units, choosing the programming language toolchain that reduces energy consumption (such as the compiler or interpreter) or configuring it, change in data structures used to store and process data or leveraging of multi-threading or symmetric multiprocessing capabilities.

Since most of these items require hands-on experience, we have decided to use active learning methodologies, organizing the students in groups and making them develop a project from scratch where they will first measure energy consumption and then minimize it. The grade can be tied to the reduction achieved. This way, putting the best practices in green computing to use, the students will be able to learn the skills and commit them to muscle memory so that they can be deployed in the future when needed.

REFERENCES

[1] CC2020 Task Force. 2020. *Computing Curricula 2020: Paradigms for Global Computing Education.* Association for Computing Machinery, New York, NY, USA.
[2] Patrick Kurp. 2008. Green computing. *Commun. ACM* 51, 10 (2008), 11–13.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
Conference'17, July 2017, Washington, DC, USA
© 2024 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn