The Journal of Systems and Software 117 (2016) 185–198

Contents lists available at ScienceDirect

The Journal of Systems and Software

journal homepage: www.elsevier.com/locate/jss

Empirical evaluation of two best practices for energy-efficient software development

Giuseppe Procaccianti*, Héctor Fernández, Patricia Lago

VU University Amsterdam, De Boelelaan 1081a, 1081 HV, Amsterdam, The Netherlands

ARTICLE INFO

Article history:
Received 12 July 2015
Revised 16 December 2015
Accepted 23 February 2016
Available online 4 March 2016

Keywords:
Software engineering
Best practices
Energy efficiency

ABSTRACT

Background. Energy efficiency is an increasingly important property of software. A large number of empirical studies have been conducted on the topic. However, current state-of-the-Art does not provide empirically-validated guidelines for developing energy-efficient software.

Aim. This study aims at assessing the impact, in terms of energy savings, of best practices for achieving software energy efficiency, elicited from previous work. By doing so, it identifies which resources are affected by the practices and the possible trade-offs with energy consumption.

Method. We performed an empirical experiment in a controlled environment, where we applied two different Green Software practices to two software applications, namely query optimization in MySQL Server and usage of "sleep" instruction in the Apache web server. We then performed a comparison of the energy consumption at system-level and at resource-level, before and after applying the practice.

Results. Our results show that both practices are effective in improving software energy efficiency, reducing consumption up to 25%. We observe that after applying the practices, resource usage is more energy-proportional i.e., increasing CPU usage increases energy consumption in an almost linear way. We also provide our reflections on empirical experimentation in software energy efficiency.

Conclusions. Our contribution shows that significant improvements in software energy efficiency can be gained by applying best practices during design and development. Future work will be devoted to further validate best practices, and to improve their reusability.

1. Introduction

The energy impact of software has been recognized as significant with respect to the overall energy consumption of its execution environment (Capra et al., 2012b; Procaccianti et al., 2012). Many researchers have been working on sophisticated software power models (Sinha and Chandrakasan, 2000; Kansal and Zhao, 2008) able to estimate and predict the energy consumption of software applications through different parameters. In spite of this effort, no reusable information is available for practitioners and developers to create energy-efficient software applications. A step in this direction has been made by Larsson (2011) from Intel Corp., which provided a number of guidelines and best practices for creating energy-efficient software. However, little to no validation has been performed on those practices, and their effectiveness in terms of energy consumption has not been precisely quantified.

To understand how software can impact on energy consumption, consider the following example[1]: after launch, the popular Youtube video of the "Gangnam Style" song reached a record amount of visualizations during the first year after its publication–roughly 1.7 billion. The amount of energy used by Google to transfer 1 MB across the Internet (as reported by the company on their website[2]) is 0.01 kWh (a rough average), and displaying it uses 0.002 kWh (depending on the destination device). Hence, the energy needed to stream and display the "Gangnam Style" video is 0.19 kWh. Multiplying this amount of energy by the 1.7 billion visualizations gives 312 GWh of total energy consumption, which is roughly the yearly energy demand of a city of 22.000 inhabitants (as an example, the city of Isernia, Italy, consumed 340 GWh of electricity in 2013 (TERNA, 2013)).

However, this impressive amount of energy might hide huge wastes. A complex software architecture lies behind modern web

* Corresponding author. Tel.: +31 644549209.
E-mail addresses: g.procaccianti@vu.nl (G. Procaccianti), hector.fernandez@vu.nl (H. Fernández), p.lago@vu.nl (P. Lago).

[1] https://www.2degreesnetwork.com/groups/energy-carbon-management/resources/gangam-style-it-sustainable/.
[2] http://www.google.com/green/bigpicture/.

http://dx.doi.org/10.1016/j.jss.2016.02.035
0164-1212/

applications and services (e.g., webservers, database servers, middleware) and countless instances are executed every second in physical and virtual environments. Even a tiny optimization on a single software application, on such a massive scale, could potentially lead to significant energy savings. For this reason, software architects and developers need to think about energy efficiency and a solid knowledge base is needed to provide guidance in building energy-efficient software.

The aim of this work is assessing the impact of two best practices for energy-efficient software development on energy consumption. We want to answer the following research questions: 1. what is the energy impact of each practice, 2. what are the main factors causing such impact (i.e., system resources).

Our work follows the guidelines for empirical experimentation in software engineering provided by Wohlin et al. (2012) and Basili et al. (1986). For the purpose of this experimentation, we selected two best practices for energy-efficient software development. We elicited those practices inspired by academic literature and industry (Simunic et al., 2001; Saxe, 2010; Jones, 2007; Steigerwald et al., 2007; Baek and Chilimbi, 2010; Larsson, 2011; Gude, 2010) and collected them in a wiki[3] to share them with academics and practitioners. Such practices were chosen because they are applicable to widely-used and well-known open source software applications (specifically, we selected the Apache WebServer and the MySQL Database Server) that will serve as test cases. These applications were executed in a controlled environment (the Software Energy Footprint Lab, SEFLab (Ferreira et al., 2013)). During the experiment, we gathered two types of data: power consumption (both of the execution environment as a whole and of the single hardware components) and usage ratio of the different system resources. Then, we performed hypothesis testing on the data to answer our research questions. Besides assessing the energy impact of each practice, we elicited, for each test case implementation, the factors that we identified as most relevant for energy consumption purposes. From this comparison, we extracted meaningful information to further define the complex relationship between software and energy.

This paper is organized as follows: Section 2 presents our vision for research in software energy efficiency, along with an overview of previous empirical studies on the energy consumption of software applications; Section 3 introduces the aim of our contribution and the research questions that drive our research. In Section 4 we present our study design, in terms of subjects, objects, dependent/independent variables and instrumentation. In Section 5 we describe how the experiment was executed. In Section 6 we present our experimental results for each practice and hypothesis testing. In Section 7 we answer our research questions and discuss the implications of our findings. In Section 8 we discuss the validity aspects and possible threats arising from our experiment design and execution. In Section 9 we draw conclusions and outline our future research efforts.

2. Background

2.1. Research vision

The present contribution falls in the scope of *Green Software* research. *Green software* can be characterized in two ways: (1) the software code being "green", i.e., with a reduced environmental impact, typically in terms of energy consumption, independently of its purpose (as in *green IT*, or *greening of IT*) or (2) the software purpose being "green", improving the environmental impact of the supported processes in their usage context (also denoted as greening by IT (Lago et al., 2015)). In particular, our contribution is focused on the energy efficiency of software applications, hence we refer to Green Software as in (1). For this reason, in the remainder of this paper the terms "Green Software" and "Energy-efficient software" will be used interchangeably.

Modern technologies (e.g., distributed systems, mobile computation, virtualization and cloud computing) make the relationship between hardware and software more complex, especially in terms of energy use. For this reason, we consider energy efficiency an *emergent* property of software systems in use: the complexity of hardware-software interactions and multiple software layers create an environment that we are currently unable to deterministically describe. Consequently, our research makes use of an *inductive* approach, i.e., we build knowledge on software energy efficiency by gathering and analyzing empirical evidence (Basili, 1993).

Such evidence, obviously, needs to be contextualized in order to provide a consistent body of knowledge. Specifically, the hardware execution environment plays a very important role. For example, as we will show in our analysis of related work in the next section, performing experimentation on mobile devices, servers, or embedded systems results in a variety of different insights—which can often be contradictory. This is why, aside from simply assessing the impact of best practices, in our research we also try to explain the mechanisms behind such impact, by measuring and analyzing context-specific data. In this paper, we select a small number of best practices and applications as subjects—a decision that clearly poses an issue of generalization to our results (see Section 8)—but we focus on precision of measurements and rigorous data analysis techniques with the goal of increasing the internal validity of our experimentation.

2.2. Related work

A number of empirical experiments on software energy consumption have been conducted as Green Software became a popular research topic. In this section, we present those which are more related to our contribution, ordered by publication date, and summarize their findings. In Table 1 and 2 we give a more structured overview: we list the *purpose* of the study, the experimental *context* (e.g., on-line vs. off-line (Wohlin et al., 2012)), the *subjects* selected for the study and the *testbed* on which the energy measurements were performed (where applicable). As criteria for selection, we focused on the viewpoint of developers: hence, we selected studies analyzing the impact of programming techniques or practices on energy consumption, as well as studies that try to empirically characterize energy-intensive code elements.

1. Capra et al. (2012a) analyze the impact of application development environments over the energy efficiency of software applications. They propose a measure of the impact of application environments on the development process, called *framework entropy*, and evaluate it over a set of 63 open source applications. Hereby we list the main findings of this work:
   - *Finding 1.* A high framework entropy is beneficial for the energy efficiency of small and medium applications.
   - *Finding 2.* A high framework entropy is detrimental for the energy efficiency of large applications.
   - *Finding 3.* Different functional types of applications have different energy efficiency levels.
   - *Finding 4.* ERPs, text, image editors and games are less energy efficient than FTP clients and servers, and calendars.
2. Sahin et al. (2012) investigate the energy impact of using software design patterns. They consider a set of 15 design patters and evaluate the energy consumption of a "proxy" application developed on purpose for the study. The application

[3] https://wiki.cs.vu.nl/green_software/index.php/Best_practices_for_energy_efficient_software, last visited on December 5th, 2015.