A Comparative Study on the Energy Consumption of PHP Single and Double Quotes

Peter Olawale Olaoluwa
School of Computing, Creative Technologies, and Engineering,
Leeds Beckett University,
Leeds, UK
p.olaoluwa3698@student.leedsbeckett.ac.uk

Ah-Lian Kor
School of Computing, Creative Technologies, and Engineering,
Leeds Beckett University,
Leeds, UK
A.Kor@leedsbeckett.ac.uk

Colin Pattinson
School of Computing, Creative Technologies, and Engineering,
Leeds Beckett University,
Leeds, UK
C.Pattinson@leedsbeckett.ac.uk

Abstract— This paper is an enhanced version of the paper presented at the SEEDS Conference (Olaoluwa, et. al, 2015). The increasing rate of carbon dioxide and other greenhouse gas emission resulting from the use of IT and other human activities to the atmosphere has become a major source of concern. It is imperative for the IT sector to ensure that its products are effective and energy efficient accompanied by mitigated negative impact on the environment. Reducing energy consumption of IT products is a key to contributing towards a greener environment. Another alternative is to produce energy efficient codes for software applications. In programming or scripting languages, an end result can be achieved in more than one way. For example, in PHP, a print command can be executed using a single quote and can also be achieved using a double quote. They have similar functions with similar quality of the intended outcomes. The aim of this research is conduct an investigation on the energy consumption of selected PHP scripts that perform similar functions: print single and double quote; echo single and double quote, etc... The Joulemeter energy measuring tool is used to measure the amount of energy consumed when run the various PHP scripts.

Keywords—energy, consumption, efficiency, software, application, rules, green, code, programming language, joulemeter

I. INTRODUCTION
According to GESI (2015), ICT is a “central pillar in the response to climate change”. Their Smarter2030 report has discussed findings which show that an ICT-enabled world of 2030 is cleaner, healthier and more prosperous. It has also identified possibilities of decoupling economic growth from resource depletion and emissions growth. This is through the use of “Green IT” with two-fold aim. They are: (i) “Greening IT” which involves producing energy efficient IT products with environmental sustainable designs, operations, use, infrastructure and systems to mitigate the environmental impact of ICT itself; (ii) “Greening by IT” which is harnessing IT (via ICT-enabled solutions) to mitigate the environmental impact of other sectors listed in Smarter2020 and Smarter2030 (GESI, 2012, 2015) and it involves the deployment of ICT to create energy-efficient and environmental sustainable operations, processes, and practices (Pattinson, et. al, 2015; The Climate Group, 2008). Several ways in which IT could help to mitigate its own environmental impact is through the development of energy efficient hardware, software, storage, networks, communications, infrastructure, etc... However, according to Murugesan and colleague (2012), an energy-unfriendly software component or program or script can render inefficient of all the energy management functionalities built into the hardware. Much research work has focused on energy management software and also green software to reduce the energy consumption of hardware. There is a need to conduct scientific experiments for measuring the actual energy consumed when running a piece of code or application. Therefore, this research aims to investigate the energy consumption when running some PHP: Hypertext Processor (PHP) commands which perform similar functions (e.g. print and echo; use of single and double quotes, etc efficient codes. The aim of this research is to use the Joulemeter to investigate the energy consumption of selected PHP commands. A set of research objectives to support the aim is as follows:

• Research Objective 1: To conduct a critical literature survey on energy efficiency of software;
• Research Objective 2: To write different PHP scripts with similar functions;
• Research Objective 3: To conduct quantitative experiments to measure the energy consumption of the PHP codes in Research Objective 2;
• Research Objective 4: To analyse results, discuss findings and make recommendations.

II. LITERATURE REVIEW
A. ICT Energy Consumption And Environmental Impact
According to the Climate Group (2008), two percent of the global carbon emission comes from the ICT sector (note: this includes PCs, Data Centers and other peripherals) while the remaining 98 percent comes from all other sectors such as health, transportation and the education. Additionally, the emission from the ICT sector is expected to rise by the year 2020 in a business as usual (BAU) scenario from the recorded 0.53 billion tonnes of 2002 to about 1.43 billion tonnes of global carbon emissions (ibid). The increasing carbon emission and other greenhouse gases will have a direct impact on the world climate and also the economy if there is no intervention (Murugesan, 2007)

To reiterate, a lot of research and effort has gone into the production of energy efficient hardware and other embedded
systems. However, there is also the need for similar focus on the energy consumption of application software developed using platforms such as PHP and Java (Capra, et al., 2011).

B. When Is Software Green or Energy Efficient?

According to Intel (nd; 2011b), an energy efficient software which is computationally efficient, helps save execution time and energy consumption. A list of computational efficient techniques listed by Intel is: efficient algorithms, multi-threading, and vectorization.

Green or energy efficient software or application will incur minimal direct or indirect negative impacts on the economy, society, humans or environment as a result of its development, deployment, or usage (Naumann, et al., 2011). According to them, a green or sustainable software product can be achieved when the developer is aware of the likely negative and also the positive impacts of the product when it is being deployed. These direct or indirect impacts of software are further explained in the book entitled “Green Information Technology: A Sustainable Approach” (Dastbaz, et al., 2015). Engel (2015) describes the direct impact of a piece of software as the effect it produces on the users, economy, society or the environment as a result of its use. This impact of software is much easily noticeable by the software users especially due to its effect on the battery life of laptops and other mobile devices. Additionally, the direct impact of software on the host mobile devices account for approximately between 25% and 40% of their total energy consumption.

There is a pressing need for software product designers and developers to optimize their products in order to reduce their negative impact on the environment. Intel (2011a) has provided energy efficient software guidelines which focus on how to optimize applications for energy efficiency. It is recommended that the application developer view power efficiency at a system-wide level because running background applications might affect or interact with the target application. Naumann, et al (2011) have developed the GREENSOFT Model which is a conceptual reference model for green and sustainable software to support software developers in the creation and maintenance of software in a sustainable manner. Ia-manee and colleague (2010) have experimented ways to reduce energy in C programs using code transformations (e.g. function, loop optimisation, control structure and operator) which focus on cohesion. Their findings suggest different types of source code yield different energy consumptions based on cohesion measures.

C. Green Software and Hardware Energy Consumption

To reiterate, the current focus of IT energy efficiency is on software. Intel (nd) stresses the important role in reducing the energy used on mobile devices and extending the battery life. Inevitably, as hardware becomes increasingly powerful, the influence of software behaviour on energy consumption grows significantly (Ardito, et al., 2015). Thus, software behaviour has a significant impact on a platform’s energy consumption and its battery life. It has significant influence on whether the energy saving features built onto the platform are effective (Mitra, et al., 2013). Undeniably, a single ill-behaving software application or service can eliminate all the benefits designed into the hardware and can have an adverse impact on platform power (Intel, 2011a, nd; Murugesan and Gangadharan, 2012). Firstly, it could hinder the effective working of the energy saving features of the hardware (Intel, nd). Secondly, all forms of hardware include significant software components (Bener, et al, 2014) and although software does not consume energy directly, it affects hardware utilisation which results in indirect energy consumption. This is echoed by Ferreira and colleagues (2013) who state that a piece of hardware dissipates energy in response to the instructions in a piece of software.

Ardito and colleagues (2015) view writing energy efficient software as one of the most challenging issues because it requires the reduction of the software effect on the energy consumption of the underlying hardware. Muttrejat and colleagues (2005) view that software energy estimation as a critical step in the design of energy efficient embedded systems. However, internal software energy measurement tools consume energy while external measurement tools will be rather complex because it will entail the use of sensors (Calandrini, et al, 2013). Seo and colleagues (2008) have defined and evaluated a framework for the estimation of energy consumption in pervasive Java-based software systems which are relevant for distributed, embedded, and pervasive applications. A unified view of the strategies, models and tools for the design, development, and deployment of greener software is necessary (ibid). In making software and applications energy-efficient, developers need to be ready to make certain trade-offs between the requirements of customers and the concern for sustainability (Berner, et al., 2014).

D. PHP Commands and Energy Consumption

PHP is an acronym that stands for PHP: Hypertext Processor. It is a recursive acronym that references itself, which means it is an acronym within an acronym. The first acronym, PHP stands for Personal Home Page (Beighley & Morrison, 2008). It is a server-side programming language. This means that its code is stored in PHP scripts that run on a web server. These scripts usually have a.php file extension. It is used for web development and also as a multipurpose programming language. PHP can be used for server-side scripting, for command line scripting and as a platform for writing desktop applications and programs on all major operating systems such as Linux, Microsoft Windows and the Mac operating Systems (The PHP Group, 2015).

In PHP, some of these commands such as the print and echo have similar functions. However, these commands can be written in more than one way, i.e. whatever that needs to be displayed could either be enclosed within single or double quotes. Every software or program code or script written consumes some measure of energy (San Murugesan & Gangadharan, 2012), and it is recommended that one way to make software and applications energy-efficient and environmentally friendly is through their algorithms and data structures. It is suggested that as long as requirements permit and as long as it is effective to get the job done, less complex algorithm should be chosen because they are more energy efficient.
Efficient. Highly recursive algorithms should also be minimized as they can be energy inefficient (Intel Corporation, 2008).

E. Metrics, Measurement and Tools for Energy Monitoring
Measurement of energy consumption of personal computers was first performed in detail and published in conjunction with the United States Department of Energy late in the 1980s (Harris, et al., 1988). This was followed by total power estimates used to measure the energy consumption of office IT equipment as described by Koomey and colleagues (1996). The first IT energy specification by Energy Star began with personal computers early in the 1990s. Subsequently, the amount of energy used by computers and other IT systems has been a priority (Johnson & Zoi, 1992). The discussion of energy measurement and monitoring tools is necessary because it is relevant for achieving higher energy efficiency with the use of IT equipment and its subsystems, and lowering Greenhouse gas (GHG) emissions is the goal of green computing (Murugesan & Gangadharan, 2012). Additionally, there is a popular adage which says “You cannot manage what you cannot measure”. A set of tools to evaluate and measure the energy consumption of software has been listed by Murugesan and his colleague. Capra and colleagues (2011, 2012) have developed an approach to measure energy efficiency in software applications, present a method to develop and apply metrics and measurement methods to measure the energy consumption that is directly related to the software. GreenTracker (Amsel, et. al., 2011) measures the energy consumption of software in order to raise awareness about the environmental impact of software usage. The Joulemeter by Microsoft (Microsoft Research, 2015) is a tool for measuring energy consumption in virtual machines, servers, desktops, laptops and software applications individually running on the computer. On the other hand, Kaliterre has created the Greenspector (Philippot, et al, 2014) which is a tool that aims at measuring the computer resource consumption during the loading of a website in a browser. However, JouleSort (Rivoire, et. al., 2007) is a balanced energy-efficiency benchmark for the evaluating the energy efficiency of a wide range of computer systems from clusters to handheldds. PowerPack (Ge, et. al., 2010) provides an energy profile and analysis of high-performance systems and applications.

There are several tools available for measuring power consumption externally. Such tools include wattmeter, multimeter, power meter and the oscilloscope. These tools record the voltage and current used for the activity and records the power consumption in watts and the energy consumption in watt hours (GHG Protocol ICT Stakeholder Advisory Group, 2012). Other energy measurement tools include the Kill A Watt (P3 International Corporation, 2015) which allows the calculation of electrical energy expenses at intervals and also help to monitor energy usage.

F. Why Joulemeter?
To reiterate, the Joulemeter is chosen for this research because it has the capacity to measure the energy consumption of software applications resident on a computer and other IT systems (Microsoft Research, 2015). The Joulemeter is a great tool for measuring and monitoring power usage of IT equipment and is particularly useful for web developers who wish to optimize their software and related services by using energy and power consumption measurement to their advantage.

With a very user-friendly dashboard, the Joulemeter can be used to view power consumption of the computer and also track power usage of specific applications (Microsoft Research, 2011). This provides the opportunity to merely focus on the measuring process of target applications (e.g. the launch of created PHP webpages in Mozilla Firefox) followed by the appropriate measurement of energy consumption. It is very easy to install, calibrate, use and monitor.

III. METHODOLOGY

A quantitative experiment method is employed for this research. As said earlier, the Joulemeter tool is used for this experiment. It is a software energy monitoring tool that provides the opportunity to monitor the total power utilization of the computer as well as individual power usage of key components of the computer such as the CPU power, Disk power, Monitor power and the Base or Idle power (Microsoft Research, 2011).

A. Experimental Design
A total of ten web pages are created and each web page only contains one PHP command from the following list:
- Echo Single Quote
- Echo Double Quote
- Print Single Quote
- Print Double Quote
- Concatenate Single Quote
- Concatenate Double Quote
- Include Single Quote
- Include Double Quote
- Switch Statement Single Quote
- Switch Statement Double Quote
- Print, Concatenate
- Include
- Switch
- Statement

Each web page contains 250 lines of codes for consistency purposes (note: this is an improvement of the experimental procedures described in Olaoluwa, et. al, 2015). Each set of commands will produce similar outputs. For example, the echo single and double quote have the same outputs. The same goes for the Print, Concatenate, Include and the Switch Statement. For easy access and in order to reduce the time spent on navigation from one webpage to the other, all created pages are put together in an index page. These web pages are launched on Mozilla Firefox browser and a Joulemeter experiment is set up to capture the estimated energy consumption of the web page on the Firefox browser. The corresponding result is exported to a .csv file format and analyzed using Microsoft Excel.
B. Joulemeter and Calibration

The Joulemeter software is downloaded directly from the Microsoft website http://research.microsoft.com/en-us/projects/joulemeter/ and the downloaded Joulemeter setup file is installed according to system specification on the hard drive of the computer to be used for the experiment. The system specification is as follows:

- **Model:** HP Pavilion 15
- **Operating System:** Windows 8.1, 64 bit
- **Processor type:** Intel core i3, 1.80 GHz processor speed
- **Storage:** 500GB
- **RAM:** 4GB

In contrast to the desktop, the laptop does not require any external power metering device such as the Watts Up Pro power meter (Microsoft Research, 2011), and that makes the installation straightforward.

Calibrating Joulemeter (see Figure 1) requires getting the computer’s power model (Microsoft Research, 2011). The Joulemeter calibration setup for the laptop is done while running on battery power as specified by the user’s manual. The calibration is done manually according to system specification because the tool does not support automatic calibration in Windows 8.1. This manual calibration however, is carried out according to the recommendations in the Joulemeter user’s manual. All open programs are closed and all USB devices unplugged before the calibration exercise (Microsoft Research, 2011).

C. Data Collection

In preparing for data collection, the Joulemeter is set to target the Firefox (version 37.0.2) browser to capture the energy usage of each page when launched. This is achieved by typing the word “firefox” in the section for “Application Power (CPU only)” in the Power Usage tab as seen in the computer’s task manager. This is a recommended procedure for using the Joulemeter to capture the power impact of a software program (Microsoft Research, 2011). Doing this enabled the measuring tool to capture only the estimated power usage of the target application on the Central Processing Unit, rather than capturing the power consumption of all running applications. The location to save the .csv file and the name of the file is set as appropriate. On the Joulemeter, the reading for the application power is initiated by clicking on the “Start saving” button clicked and the page is launched to begin data capture.

IV. RESULTS AND DISCUSSION

It is observed from the generated Comma Separated Values (csv) file that the total usage of power by the computer (Total Power) for the process is the sum of CPU Power, Monitor Power, Disk Power and the Base Power. The Application Power is recorded separately by the Joulemeter. In Table 1, we have the following formulae:

\[
\text{Hardware Total (W)} = \text{CPU (W)} + \text{Monitor (W)} + \text{Disk (W)} + \text{Base (W)}
\]

\[
\text{Total Power Consumption (W)} = \text{Hardware Total (W)} + \text{Power Consumption of Application (W)}
\]

A. Normalisation of Data

In Table 1, \( n \) represents the number of lines of codes in the created web page (note: \( n=250 \)). The collected raw data undergo a series of normalisation in order to provide a fair comparison among the various PHP commands as well as the single and double quotes. Firstly, the power consumption of the hardware, application and the total power consumption are computed accordingly for \( t = 1 \text{s} \) (see Table 1). The values in Table 2 are calculated by using the formula, Energy (J) = Power (W) x Time (s) where the time, \( t = 1 \text{s} \).

Table 3 shows the second step of normalization where energy consumption is computed for each line of code (i.e. \( n = 1 \)). Consequently, this yields the metric, Joule per line. The goal of this normalisation is to provide a fair comparison for all the parameters set for the experiments. The graphs in Figures 2-4 are plotted based on the values in the normalized values for energy consumption in Table 3 (i.e. \( t = 1 \text{s} \), and \( n = 1 \)).
Figure 2 depicts the hardware energy consumption per line of code for the following PHP commands: echo, print, concatenate, include and switch with single and double quotes. The graph shows that the double quote consumes more hardware energy than single quote for all the investigated PHP commands except for the print command. All the PHP commands seem to consume almost the same amount of hardware energy.

In Figure 3, it shows that the normalized application energy consumption for the double quotes generally exceed that of the single quotes. Just as in the hardware energy consumption, the application energy consumption for the concatenate double quote commands seems to be the highest. This is followed by the include command with double quote and the range for the rest of the values is less than 0.05 J/line. The graph for the average energy consumption for the hardware and application (Joule/line) is depicted in Figure 4. Once again, the average energy consumption for the double quotes is slightly higher than the single quotes. The concatenate and include double quotes commands have the highest normalized total energy consumption for the hardware and application while the range for the rest is less than 0.05 J/line.
Table 4 provides an insight into the % gain in the average energy consumption (in Joule/line) of single quotes compared to double quotes for the corresponding PHP commands. The gain seems to be highest for the concatenate command, while the print command seems to be a negative gain. The range for the values in Table 4 is 10.642%.

**B. Aggregation of Data**

Table 5 shows the aggregated normalised energy consumption for all the single and double quotes in Table 3. The graphs in Figures 5-7 are plotted based on these values. Figure 5 reveals that the aggregated energy consumption for the monitor and base are similar. This is because the outputs for the single and corresponding double quotes for the same PHP command are the same. The aggregated CPU energy consumption for the double quotes is higher than that of the single quotes, thus contributing to a consistently higher aggregated hardware energy. This result is also consistent with the aggregated application energy consumption which is depicted in Figure 6, and Figure 7. In the latter, it shows that the aggregated energy consumption of the application for the double quote is higher than that of the single quote.

**C. Two tailed t-test for two samples (n = 1 line of code)**

The data analysis (in Section IV (A and B)) shows that the energy consumption for launching PHP web pages that contain single quotes is lower than that with double quotes. Additionally, Table 4 depicts the positive % gain in energy consumption for single quotes compared to double quotes. However, further statistical analysis is necessary to determine if their energy consumption is significantly different. Consequently, a two-tailed t-test (for two samples) is used to test whether there is any significant difference between the energy consumption for single and double quotes at confidence level ($\alpha$): 0.05.
Table 6 provides information on the total energy consumption (Joule/line) for aggregated PHP commands with single and double quotes. Tables 7 to 9 reveal the results of a two-tailed t-test for two samples with unequal variance at confidence level, $\alpha = 0.05$. The tests reveal that there are no significant differences between the aggregated energy consumption (hardware alone, application alone, hardware and application) for single and double codes. The confidence level, $\alpha$ is changed to 0.10 and the differences remain insignificant. The conclusion that could be drawn here is that though the hardware and application energy consumption for single quotes seem to be lower than double quotes, there is no significant difference in the gain. Thus, further experiments and tests will be necessary to confirm this finding.

V. CONCLUSIONS

In this research, a series of data normalization (i.e. t=1s, and n=1 line of code) is necessary in order to provide a fair comparison between the different PHP commands (i.e. echo, print, concatenate, include, and switch) with single and double quotes. In summary, the single quote seems to consume less energy than the double quote in PHP though the t-statistical tests conducted on their differences yield an insignificant outcome and the findings in this paper confirm our previous findings (Olaoluwa et al., 2015). Further rigorous experiments will be necessary to confirm these results. Viewing the fact that green computing is aimed at design, production, usage and disposal of computers and its other subsystems in a way that causes little or no damage to the environment (San Murugesan & Gangadharan, 2012), it is imperative to consider all possible ways of reducing energy consumed as a result of IT or computing related. The compared PHP commands can be used interchangeably, therefore, developers need to opt for the more energy-efficient ones as much as possible. The following issues ought to be addressed in order to further enhance the experimental procedures: (i) it is necessary for the n value (i.e. no of lines of codes) for all the PHP commands with single and double quotes be set as a parameter for the experiments; (ii) conduct repeated experiments for each command (note: the number of repeated runs for each set of experiments in this research is 19 and this could be systematically increased).

Two tailed t-test for two samples with unequal variance (n = 1 line of code)

<table>
<thead>
<tr>
<th></th>
<th>Ni</th>
<th>df</th>
<th>Mean</th>
<th>SD</th>
<th>t-value</th>
<th>t-critical value</th>
<th>significance</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Single Quote</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Single Quote</td>
<td>5</td>
<td>8</td>
<td>0.0710</td>
<td>0.0002</td>
<td>1.017</td>
<td>2.0505</td>
<td>not significant</td>
</tr>
<tr>
<td>Double Quote</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 7: A two sample t-test with unequal variance at confidence level $\alpha = 0.05$ for aggregated hardware energy consumption

VI. REFERENCES


