Applications, energy consumption, and measurement

Conference Paper · July 2015

DOI: 10.1109/DT.2015.7222967

5 authors, including:

Ah-Lian Kor
Leeds Beckett University
131 PUBLICATIONS **707** CITATIONS
SEE PROFILE

Colin Pattinson
Leeds Beckett University
97 PUBLICATIONS **767** CITATIONS
SEE PROFILE

Ibrahim Alsaleemi
Sultan Qaboos University
1 PUBLICATION **7** CITATIONS
SEE PROFILE

Applications, Energy Consumption, and Measurement

Ah-Lian Kor, Colin Pattinson, Ismail Imam, Ibrahim AlSaleemi, Oluwafemi Omotosho
School of Computing, Creative Technologies and Engineering,
Leeds Beckett University
{A.Kor, C.Pattinson}@leedsbeckett.ac.uk

Abstract:
The task of reducing the energy footprint of IT devices and software has been a challenge for Green IT research. Monitoring approaches have primarily focused on measuring the energy consumption of the hardware components of computing devices. The use of applications or software on our computer systems consumes energy and it also affects how various hardware components and system resources consume energy. Consequently, running web browsers applications will utilise considerable energy and battery consumption. In this research, we have run different types of experiments which involve the use of several measuring tools. Firsly, a joulemeter is used to monitor (and measure) the power consumed by the hardware and software while running web-based and stand-alone applications on several devices. Additionally, the tablet in-built battery status checker is used to measure the battery consumption when web-based applications are run on the device.

1 Introduction

Green computing technology focuses on the efficient use of computing resources. In computing devices such as laptops, smartphones, tablets, or other mobile devices, energy consumption is the top priority because they are run on battery, with limited lifespan, as their source of power (Banerjee et al. 2007). With the increasing complexity of IT equipment, the energy consumption rate of these devices system also increases (Silven and Jyrkka, 2007). Most portable mobile device users are conscious of the energy usage by these devices and consequently, they look for ways through which the lifespan of the battery can be extended to serve them longer (Rahmati et al. 2007).

Experiments relating to energy measurement could be at various levels: the hardware level; energy efficiency directive level (Simunic, et al. 2000); operating system (Sagahyroon, 2006); software application or data and user levels (Ravi, et al. 2008). Energy conservation is made possible through the use of different techniques which estimate or forecast energy consumption at the device and application level (Krintz, et al. 2004). The goal of green computing technology is to reduce carbon emission, maximize performance and prolong the lifespan of the computing resources.

1.1 Aim

The aim of this paper is to discuss the results for several investigations conducted on the energy (and battery) consumption for running web-based and standalone applications on Windows and IOS portable computing devices. The following objectives will help to achieve this aim:

- Research Objective 1: To conduct experiments on the measurement of energy consumed for running youtube videos in different web browsers (e.g. Google Chrome, Mozilla Firefox, etc…) on Windows (i.e. laptops), and IOS machines (i.e. tablet);
- Research Objective 2: To conduct experiments on the measurement of energy consumed for playing audio and video files on several media players for windows (on a laptop);
- Research Objective 3: To conduct analyses on data collected in Research Objectives 1 and 2.

This paper will be organised into the following sections: Introduction; Literature Review; Methodology; Results and Discussion; and Conclusion.

2 Literature Review

The Smart2020 report (The Climate Group and GeSI, 2008) predicts an increasing trend of BAU CO2 emissions for the ICT industry. The emissions growth rate for three ICT categories (end-user devices, telecommunication and networks, and data centers) is expected to decrease from 6.1% 3.8%. By 2020, the ICT industry's footprint is expected to rise to 1.3 $GtCO_2e$ (equivalent to 2.3% of global emissions by 2020). The PC (e.g. desktops, laptops, etc.) footprint (due to its embodied and usage emissions) is the highest (60%) followed by printers (18%), peripherals (13%), smartphones (10%), and tablets (1%). It is estimated that the footprint of end-user devices will grow at 2.3 percent per year to reach 0.67 $GtCO_2e$ in 2020 and thus, energy efficiency improvements in these devices and their proper usage are essential for reducing their overall footprint.

2.1 Energy Consumption of software

Green and sustainable software is a software product that has the smallest possible economic, societal, ecological impact as well as impact on human beings (Ahmed, et al., 2014). This has led to the introduction of various programmes and initiatives that encourages energy efficient software such as green software engineering and Eco-design software (Kaliterre, n.d.).

According to the Greenhouse Gas Protocol (2012), applications are executed with an OS. They affect the power consumption of a device due to data requests and processing. Managing energy requires accurate measurement of the energy available and consumed by a system. This involves monitoring or estimating the resource and energy consumption of hardware and software (Noureddine, et al., 2013). However, a device's power consumption is subjected to the type of application and the task being performed which is evident in our experimental results presented in Section 4 of this paper. In order to reduce the overall power consumption for a web-based or standalone task, it will be necessary to provide users with an insight of the power consumption of the different web-based browser applications (e.g. Google Chrome, Internet Explorer, Mozilla Firefox, Safari, etc…) and also the resource hungry nature of many applications such as movie player and games.

2.2 Energy Consumption of Media Players

Modern technologies incorporate a number of power management features to reduce power waste. Dynamic Voltage and Frequency Scaling (DVFS) can enable the CPU speed to be dynamically varied based on the workload which leads to a reduced power consumption during periods of low utilization (Liu, et al., 2008). The energy-aware dynamic voltage scaling technique has been used to reduce energy consumption in portable media players (Yang & Song, 2009). This scheme showed a relationship between frame size and decoding time. These two cited work merely discuss how energy consumption can be reduced using various techniques, but have not measured the actual amount of energy being consumed by the application. However, the energy consumption of Windows Media Player has been measured using the EEcoMark v2 tool (EecoMark, 2011) but the empirical details of the measurement have not been explicitly discussed. Media playback application power consumption has been analysed by Sabharwal (2011) using windows event tracing. Event tracing does not seem to be an appropriate method for measuring energy consumption because the process itself may have impact on the results. A comparative analysis of energy consumption of media players has been conducted by Techradar (2010). The energy consumption is monitored by playing a DVD on Windows Media Player (WMP) and VLC Media Player. Their research results show that the VLC Media Player is more energy efficient than Windows Media Player. However, the cited work has not mentioned which tool has been used for measurement and additionally, the experiment procedures have not been explitcitly discussed.

2.3 Metrics, Measure and Tools for Energy Consumption

2.3.1 Metrics

Generally, software does not directly consume energy. However, running the software involves the hardware which consumes energy. Therefore, the resource usage metric such as the CPU usage, memory and disk usage are used as the measuring criteria (Mahmoud & Ahmad, 2013). It is important to analyse the energy efficiency of software by observing the amount of resource utilized versus the useful work performed. To measure the power consumption of a system, the power consumed by individual PC components must be measured. Therefore a system wide resource utilization monitoring technique at the user level seems to be more appropriate.

2.3.2 Measure and Tools

There is a wide range of methods for measuring the energy consumption of a computer system. Generally, the measurement of energy consumption is grouped into three categories hardware, software and power models (Noureddine, et al., 2013). Measuring energy consumption of hardware using devices such as *wattsup*[1] and the method described by McIntire and colleagues (2007) yields a precise value. *PowerScope* is a tool that uses a multi-meter to measure the energy consumption of applications (Flinn & satyanarayanan, 1999). This method is more precise because it can determine the energy consumption of a specific process and even procedures within the process. However, these methods have some limitations. It can only monitor hardware devices, not flexible, requires additional hardware and the value may fluctuate due to electro-mechanical issues. It is also difficult to upgrade to a more newer and precise monitoring without replacing the entire hardware.

Power models are used to calculate the energy consumption of hardware and software. Kansal and Zhao (2008) use a generic automated tool to profile the energy usage of various resources components used by an application. This method is either too generic or coarse-grained and it is platform dependent (Seo, et al., 2007). The model proposed by Lewis and colleagues (2012) is an integrated model for the calculation of a system's energy consumption.

More promising approaches are software energy measurement using energy application profiler (Noureddine, et al., 2013). In their contribution, Varrol and Heiser (2010) use *Openmoko Neo Freerunner* to decompose the energy consumption of each resource of a system. *PowerAPI* is an Application Programming Interface (API) used for monitoring the real time energy consumption of applications at the granularity of system process (Bourdon, et al., 2013). *PowerAPI* can also be used to estimate the energy consumption of a running process for hardware resources e.g. CPU or for hard disk or for both and many more other resources (Noureddine, et al., 2013). Energy consumption estimation in PowerAPI distinguishes the energy consumption for hardware resources and software blocks of codes.

pTop is a process-level power profiling tool which provides information on the power consumption of the running processes in joules (Do, et al., 2009). It gives the power consumption values for the CPU, computer memory, hard disk and the network interface for each process. The energy consumed by an application is the sum of energy consumed by individual resources in addition to energy consumed by the interaction of these processes (Noureddine, et al., 2013). The windows version also uses the windows API to perform the same task.

Intel energy checker is a software development kit SDK with the capability to provide the Application Programming Interface (API) required to define, measure, and share energy efficiency data (Intel, 2010). The SDK is developed with the intention to facilitate energy efficiency analysis and

[1] http://research.microsoft.com/en-us/projects/joulemeter/quickstart.pdf

optimization in data centre and telecoms environment. It can, however, also be used on the client or mobile computing platforms to measure energy consumption (Intel, 2010). There is the functionality of importing and exporting counters from an application, which can measure the time spent for a particular process such as converting a file or reading a video (Noureddine, et al., 2013). However, this approach is limited in flexibility because it requires hardware power meter for power estimation.

Joulemeter is a software tool that can be used to estimate the power consumption of hardware resource and software applications on a computer (Microsoft Research, 2011). It can monitor resources such as the CPU utilization and screen brightness in order to compute these resources energy consumption. It has been used by Vonkoch and colleagues (2011) to measure and compare the power consumption of web browsers. The initial calibration of the *joulemeter* renders it inflexible. However, the power model is easy and straight forward to use. Additionally, it has various versions for different Operating Systems. It is an open-source tool which is available for free downloads from Microsoft. Joulemeter calculates the energy consumption of various components within a computing device: central processing unit (CPU); software application; the monitor etc. (Narayanan, 2005). The experimental technique is useful for obtaining real-time data and it is cheaper than purchasing an external hardware device because it does not require power supply to operate, there is no need for storage space and disposal after expiry of lifespan (which contribute to e-waste) (Widmer, et al 2005).

3 Methodology

There are three sets of physical experiments conducted to investigate the relationship between various applications and their energy (and battery) consumption. Their respective results are presented in: (i) Section 4.1 – web browser applications and their energy consumption (for a laptop); (ii) Section 4.2 – web browser applications and their battery consumption; (iii) Section 4.3 – media players for Windows and their energy consumption (for a laptop).

3.1. Experiment Setup

Equipment

Hardware devices

The hardware devices used for the 3 sets of experiments have been tabulated in Table 1.

Experiment Set	Device Type	Operating System	CPU	RAM	Screen Size	Experiment Results
1	Toshiba Protégé Z30-A-1D6	Windows 7 Enterprise (64 bit)	4th Gen Intel® Core i7-4600U @2.10GHz 2.70GHz	8.00GB	13.3”	Section 4.1
2	Toshiba satellite L50-1NL	Windows 8.1	4th Gen Intel® Core i3-4005U	4.00GB	15.6”	Section 4.2
3	Apple iPad Air2	IOS 8	triple-core A8X @1.5GHz	2.00GB	9.7”	Section 4.3

Table 1: Experiment hardware devices

Measuring Tool (Software)

Joulemeter 1.2[2] is used for experiment sets 1 and 2 while an inbuilt battery status checker is employed for experiment set 3. There are some specific guidelines which are adhered to when calibrating the Joulemeter. They are: (i) ensure that the laptop battery is fully charged (or more than 50%); (ii) automatic or manual calibration of the energy model. It is necessary to ensure that other applications or programs are not running while the automatic calibration is in progress. Detailed instructions on using the Joulemeter are found in this user manual (Microsoft Research, 2011).

3.2. Experimental Procedures

Experiment Set 1: To investigate the energy consumption of several web browser applications in Windows (the sample interface is shown in Figure 1)

Experimental Steps

i. Create a csv file for saving the real time power consumption data via Joulemeter (by clicking on the *browse button*);
ii. Click on the *start saving button*;
iii. Click on the *start button* to run the application in Google Chrome 1.3.27 (i.e a youtube video[3]);
iv. Click on the *stop saving button* to end the application;
v. Repeat the above steps for 9 times;
vi. Repeat all the above steps for each of the following web browser: Internet Explorer 9; Mozilla Firefox 27.0.1; and Safari 5.2.1.

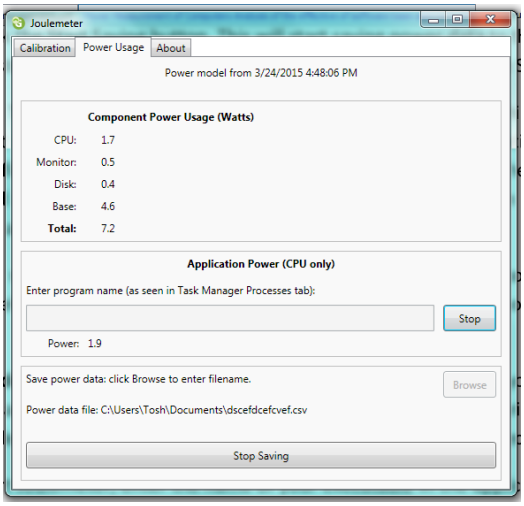

Figure 1: Sample interface for the Joulemeter Measuring Tool

The constants of this experiment are:

i. The wifi network used is eudroam;
ii. Constant environment (experiment is carried out within the same office throughout the entire experiment);
iii. The time for all the experiments is from 1200 -1600 for Day 1 and Day 2.

The limitations of the experiments are:

i. The Joulemeter only monitors the energy consumption of the client machine;
ii. Human inconsistency involved when clicking on the essential buttons (see % error due to in Table 6);
iii. Technical inconsistency which rendered several of the experiments as errors (Table 2).

[2] http://research.microsoft.com/en-us/downloads/fe9e10c5-5c5b-450c-a674-daf55565f794/
[3] https://www.youtube.com/watch?v=ZtZLFHZZlrM

Experiment Set 2: To investigate the energy consumption of several web browser applications in IOS

Experimental Steps

i. Record the battery status (in %) of the device manually;
ii. Run a 30 minutes youtube video in the following web browsers: Google Chrome, Safari, Opera Mini and Puffin;
iii. Record the battery status (in %) of the device manually immediately after (ii);
iv. Repeat all the above steps for three different times of the day: morning; noon; and night;

Constants in the experiments

i. The experiments are run in the same physical environment;
ii. The experiments are run in the same day;
iii. Volume of the device is set to *full* while the brightness, *auto*.

Critique of the experiments

i. The number of experiments for each web browser during a certain part of the day ought to be repeated at least 9 times in order to obtain an average of the readings;
ii. The tablet ought to be fully charged (i.e. 100%) before each experiment is conducted.

Experiment Set 3: To investigate the energy consumption of several media players for Windows

Experimental Steps

i. Create a csv file for saving the real time power consumption data via Joulemeter (by clicking on the *browse button*);
ii. Click on the *start saving button*;
iii. Click on the *start button* to run an audio file using KMPlayer in Windows;
iv. Click on the *stop saving button* to end the application;
v. Repeat the above (i-iv) for a video file;
vi. Repeat all the above (i-v) for the following media players: Windows Media Player (WMP) and VLC Media Player.

Constants in the experiments

i. The device's default settings are used except for the volume which is set to *100%* while the brightness of the monitor, *auto*.

Critique of the experiments

i. The number of experiments for each media player ought to be at least 4 times in order to obtain an average of the readings.

4. Results and Discussion

This section will discuss the results of the data analysis for the three sets of experiments discussed above. The Joulemeter monitored raw data is for the time stamp (in ms), power consumption (in Watts) for each component: CPU, monitor, disk, base and the application. The formula used to calculate the energy consumption by each component is: Energy (J) = Power (W) x Time (s). The results of the calculation for all the experiments runs are shown in Table 2. Note that the data is cleansed so as to omit records with application power consumption = 0W. If the number of remaining records > 50% of the raw data then the cleansed readings of the csv file will be included in the data analysis. However, if it is otherwise, then the experiment is considered an error (see Table 2).

4.1 Web Browser Applications for Windows

Browser	Experiment No	Hardware Energy Consumption (J)					Application (J)	Total Energy Consumption (J)	Total Time (s)	Aggregated Average Energy Consumption (J) for t=1s
		CPU (J)	Monitor (J)	Disk (J)	Base (J)	Total Hardware				
Google Chrome 1.3.27	1	111.168	154.130	0.000	378.319	643.616	106.652	750.268	140.606	
	2	101.205	157.087	0.000	385.576	643.868	96.657	740.524	142.806	
	3	103.981	155.673	0.000	382.107	641.761	98.958	740.719	141.521	
	4	102.365	152.213	0.000	373.613	628.190	98.770	726.960	138.375	
	5	109.315	156.681	0.000	384.580	650.576	105.205	755.781	142.437	
	6	135.710	155.001	0.000	380.457	671.167	126.990	798.157	140.910	
	7**	Error								
	8	126.437	155.297	0.000	381.183	662.917	119.757	782.674	141.179	
	9*	76.769	97.491	0.000	239.296	413.556	74.544	488.100	88.628	
	10	166.749	155.394	0.000	381.421	703.563	157.886	861.449	144.637	5.442
Internet Explorer 9	1	45.764	154.217	0.000	378.532	578.512	38.276	616.788	140.197	
	2	49.357	162.142	0.000	397.985	609.484	43.736	653.220	147.402	
	3	50.285	152.635	0.000	374.649	577.569	44.348	621.917	138.759	
	4	58.794	156.955	0.000	385.252	601.001	50.227	651.228	142.686	
	5	51.618	156.134	0.000	383.238	590.990	46.801	637.791	141.940	
	6**	Error								
	7**	Error								
	8*	32.292	116.732	0.000	286.524	435.548	17.344	452.892	106.120	
	9	47.084	153.871	0.000	377.684	578.640	34.351	612.990	139.883	
	10	57.187	153.783	0.000	377.468	588.438	49.972	638.411	139.803	4.454
Mozilla Firefox 27.0.1	1	128.149	157.489	0.000	386.564	672.203	118.922	791.125	143.172	
	2	82.580	154.375	0.000	378.921	615.876	75.664	691.540	140.341	
	3	97.265	155.539	0.000	381.777	634.581	88.251	722.832	141.399	
	4	73.910	155.802	0.000	382.423	612.134	66.883	679.017	141.638	
	5	87.325	159.672	0.000	391.921	638.918	78.392	717.310	145.156	
	6	128.675	153.585	0.000	376.982	659.243	115.772	775.015	139.623	
	7	87.172	154.356	0.000	378.875	620.403	80.608	701.011	140.324	
	8	86.112	155.170	0.000	380.873	622.155	78.340	700.495	141.064	
	9	98.018	155.813	0.000	382.450	636.281	89.956	726.237	141.648	
	10	90.892	154.529	0.000	379.299	624.720	83.158	707.878	140.481	5.098
Safari 5.7.1	1	102.231	154.069	0.000	389.845	646.145	98.505	744.650	144.387	
	2	88.365	151.310	0.000	383.786	623.461	83.946	707.407	142.143	
	3	98.013	150.006	0.000	381.194	629.213	92.471	721.684	141.183	
	4	92.004	149.867	0.000	380.047	621.917	69.480	691.397	140.758	
	5	93.047	146.667	0.000	371.272	610.985	80.209	691.195	137.508	
	6	154.067	149.436	0.000	380.489	683.992	146.294	830.286	140.922	
	7	95.686	148.821	0.000	378.729	623.236	90.298	713.533	141.378	
	8	104.739	150.848	0.000	383.824	639.410	98.896	738.307	142.157	
	9	97.910	150.878	0.000	383.846	632.633	92.643	725.276	142.165	
	10	94.616	147.126	0.000	372.743	614.485	80.421	694.906	141.267	5.134

Note: * - proportion of the remaining cleansed data > 50% of the raw data and ** is for otherwise and thus considered an error.
White coloured records – results of experiments for Day 1; Blue coloured records – results of experiments for Day 2.

Table 2: The hardware and application energy consumption for running several web browser applications

Table 3 depicts the aggregated data for all the experiments conducted for each web browser: Google Chrome, Internet Explorer, Mozilla Firefox, and Safari. However, in order to provide a fair comparison among the web browsers, the time for running the application will have to be set to 1s (i.e. t = 1s) Consequently, the corresponding energy consumption for each component will have to be normalised for t = 1s (see Table 4). Based on the results shown in Table 4, it seems that Internet Explorer 9 consumes the least energy on laptops, followed by Mozilla Firefox and Safari while Google Chrome seems to be the highest energy consumer. These results are consistent with experiments

conducted by the Center for Sustainable Energy Systems at Fraunhofer USA[4], which compare the energy consumption of Internet Explorer 10, Mozilla Firefox and Google Chrome on laptops and desktops. Their results reveal that Google Chrome consumes the highest amount of energy followed by Mozilla Firefox. The conclusion drawn by them is that Internet Explorer seems to be the most energy efficient web browser.

Browser	Aggregated Average Hardware Energy Consumption (J)					Aggregated Application (J)	Aggregated Total Time (s)	Aggregated Average Energy Consumption for t = 1s (J)
	CPU (J)	Monitor (J)	Disk (J)	Base (J)	Total Hardware			
Google Chrome 1.3.27.5	1033.70	1338.97	0.00	3286.55	5659.21	985.42	1221.10	5.442
Internet Explorer 9	392.38	1206.47	0.00	2961.33	4560.18	325.05	1096.79	4.454
Mozilla Firefox 27.0.1	960.10	1556.33	0.00	3820.08	6336.51	875.95	1414.85	5.098
Safari 5.7.1	1020.68	1499.03	0.00	3805.77	6325.48	933.16	1413.87	5.134

Table 3: Aggregated hardware and application energy consumption for running several web browser applications

Aggregated Energy Consumption for t = 1s	Aggregated CPU (J)	Aggregated Monitor (J)	Aggregated Disk (J)	Aggregated Base (J)	Aggregated Hardware Energy (J)	Aggregated Application (J)	Aggregated Total Energy Consumption (J)	Time (t=1s)
Google Chrome 1.3.27.5	0.85	1.10	0.00	2.69	4.63	0.81	5.442	1.00
Internet Explorer 9	0.36	1.10	0.00	2.70	4.16	0.30	4.454	1.00
Mozilla Firefox 27.0.1	0.68	1.10	0.00	2.70	4.48	0.62	5.098	1.00
Safari 5.7.1	0.72	1.06	0.00	2.69	4.47	0.66	5.134	1.00

Table 4: Normalised hardware and application energy consumption for running several web browser applications (for t = 1s)

Figure 2 is drawn based on the normalised values in Table 4. The normalised aggregated monitor and base energy consumption values seem to be similar for all the web browsers. The CPU energy consumed by Google Chrome seems to be the highest while Internet Explorer seems to be the lowest. On the other hand, the CPU energy consumption for Mozilla Firefox and Safari seems to be similar.

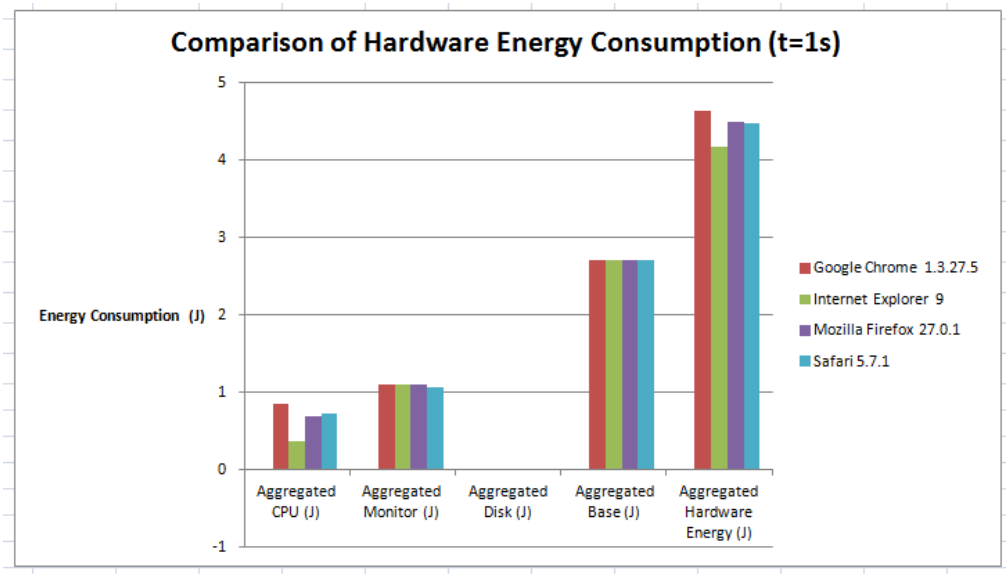

Figure 2: A comparison of the normalised hardware energy consumption for the web browsers

[4] http://news.thewindowsclub.com/internet-explorer-10-energy-efficient-browser-62912/

Figure 3 depicts the energy consumption by the hardware and application for each web browser. It can be seen that the energy consumed by the application is very much less compared to the energy consumed by the hardware that runs the application. In order to reduce the overall energy consumption, a further investigation on the interface as well as processes that occur between the software and hardware will have to be conducted and optimized. The energy consumption patterns for the various web browsers are consistent with that for the hardware.

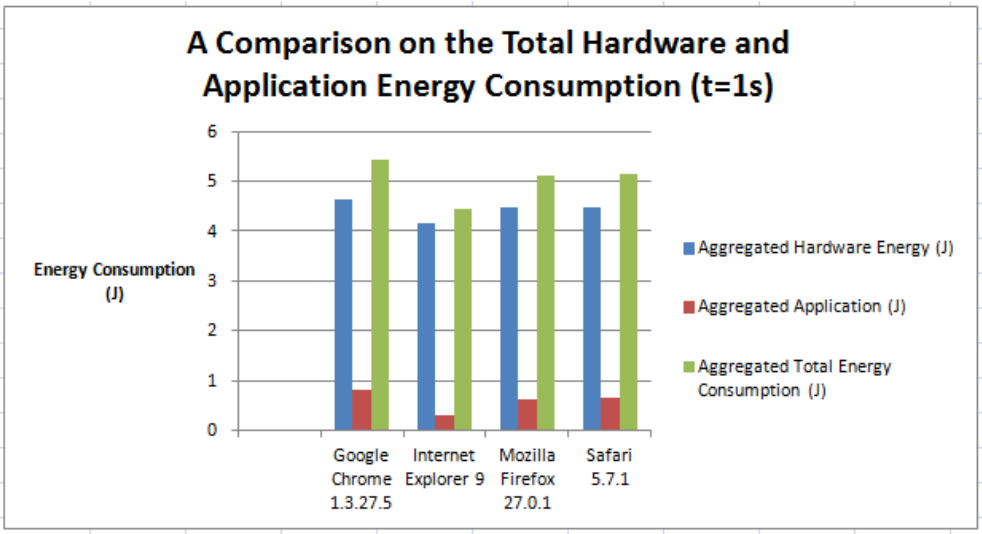

Figure 3: A comparison of the normalised hardware and application energy consumption for the web browsers

Further analyses have been conducted to investigate the ratio of energy consumption between the various web browsers and Internet Explorer (used as a base because it is the lowest energy consumer). Results in Table 5 reveal that the factor for application energy consumption by Google Chrome is almost 3 times that of the Internet Explorer while it is a factor of 2 for Mozilla Firefox and Safari. However, the corresponding factors for the hardware ares lower compared to the factor for the application. Additionally, running a youtube application on Google Chrome (in a laptop) seems to consume approximately 22% more energy than Internet Explorer (note: this is consistent with the 18% finding provided by Center for Sustainable Energy Systems at Fraunhofer USA)[5]. However, the increase for Mozilla Firefox and Safari seems to be in the region of 14%-15%.

Aggregated Energy Consumption for t = 1s	Aggregated CPU (J)	Ratio Comparison to IE
Google Chrome 1.3.27.5	0.85	2.37
Internet Explorer 9	0.36	1.00
Mozilla Firefox 27.0.1	0.68	1.90
Safari 5.7.1	0.72	2.02

Aggregated Energy Consumption for t = 1s	Aggregated Application (J)	Ratio Comparison to IE
Google Chrome 1.3.27.5	0.81	2.72
Internet Explorer 9	0.30	1.00
Mozilla Firefox 27.0.1	0.62	2.09
Safari 5.7.1	0.66	2.23

Aggregated Energy Consumption for t = 1s	Aggregated Hardware Energy (J)	Ratio Comparison to IE
Google Chrome 1.3.27.5	4.63	1.11
Internet Explorer 9	4.16	1.00
Mozilla Firefox 27.0.1	4.48	1.08
Safari 5.7.1	4.47	1.08

Aggregated Energy Consumption for t = 1s	Aggregated Total Energy Consumption	Ratio Comparison to IE
Google Chrome 1.3.27.5	5.44	1.22
Internet Explorer 9	4.45	1.00
Mozilla Firefox 27.0.1	5.10	1.14
Safari 5.7.1	5.13	1.15

Table 5: Energy consumption ratio for the various web browsers

[5] http://news.thewindowsclub.com/internet-explorer-10-energy-efficient-browser-62912/

As previously mentioned in Section 3, an experimental error may arise due to human error. The expected total uninterrupted application running time is 2 minutes and 14 seconds (134 seconds). The aggregated expected total application time has been calculated by taking into consideration the proportion of the cleansed data and also the number experiments that have been rendered as errors. The percentages of experimental error for the 4 web browsers are shown in Table 6 and it seems that the experimental error for the Google Chrome is the lowest while the rest is approximately 5 times of Google Chrome. In order to re-affirm the validity of the findings previously discussed, it will be essential to in crease the number of experiments for each browser and measures taken to reduce human inconsistency.

	Time in Joulemeter (s)	Expected Total application time (s)	Experimental error (%)
Google Chrome 1.3.27.5	1221.10	1206.00	1.25
Internet Explorer 9*	1096.79	1035.84	5.88
Mozilla Firefox 27.0.1	1414.85	1340.00	5.59
Safari 5.7.1	1413.87	1340.00	5.51

Table 6: Experimental Error

4.2 Web Browser Applications for IOS

Table 7 depicts the results of running experiments on an Apple ipad Air2 with and IOS operating system. A youtube video has been chosen for this set of experiments and the running time is 30 minutes. Additionally, the experiments are run forthree different times of the day: morning; noon; and night.

Watching YouTube video at (noon)

Browser	Battery % (Before)	Video Duration	Battery % (After)	Battery Consumption %	Brightness	Volume	Date
Chrome	100%	30 (min)	97%	3%	Auto	Full	05/05/2015
Safari	96%	30 (min)	87%	9%	Auto	Full	05/05/2015
Opera mini	87%	30 (min)	78%	9%	Auto	Full	05/05/2015
Puffin	77%	30 (min)	66%	11%	Auto	Full	05/05/2015

Watching YouTube video at (night)

Browser	Battery % (Before)	Video Duration	Battery % (After)	Battery Consumption %	Brightness	Volume	Date
Chrome	66%	30 (min)	62%	4%	Auto	Full	05/05/2015
Safari	62%	30 (min)	58%	4%	Auto	Full	05/05/2015
Opera mini	58%	30 (min)	54%	4%	Auto	Full	05/05/2015
Puffin	54%	30 (min)	48%	6%	Auto	Full	05/05/2015

Watching YouTube video at (morning)

Browser	Battery % (Before)	Video Duration	Battery % (After)	Battery Consumption %	Brightness	Volume	Date
Chrome	47%	30 (min)	40%	7%	Auto	Full	06/05/2015
Safari	39%	30 (min)	33%	6%	Auto	Full	06/05/2015
Opera mini	33%	30 (min)	28%	5%	Auto	Full	06/05/2015
Puffin	28%	30 (min)	18%	10%	Auto	Full	06/05/2015

Table 7: Results of battery consumption for the various web browsers on an IOS device

Firstly, it is noted that the battery consumption by the Puffin web browser is consistently the highest at any part of the day. However, the findings for the rest of the three web browsers (in Table 7) are rather inconclusive. Consequently, the average battery consumption for each web browser has been calculated and shown in Table 8. Once again, the average value for Puffin seems to be the highest while it is the lowest for Google Chrome. However, this finding does not seem to be aligned to the findings in the previous section where the energy consumption for Google Chrome is higher than

Safari. In summary, in order to yield more valid results, the following measures (which have been previously mentioned) will have to be taken: conducted repeated experiments for each web browser (at least 9 times in order to obtain the avaerage battery consumption value); use a fully charged battery for each experiment.

Browser	Battery Consumption (%)			Average (%)	Ratio Compared to Chrome
	Morning (%)	Noon (%)	Night (%)		
Chrome	7.00	3.00	4.00	4.67	1.00
Safari	6.00	9.00	4.00	6.33	1.36
Opera mini	5.00	9.00	4.00	6.00	1.29
Puffin	10.00	11.00	6.00	9.00	1.93

Table 8: Average battery consumption for the various web browsers

4.3 Media Players for Windows

To reiterate, the Joulemeter monitored raw data is for the time stamp (in ms), power consumption (in Watts) for each component: CPU, monitor, disk, base and the application. The formula used to calculate the energy consumption by each component is: Energy (J) = Power (W) x Time (s). The results of the calculation for all the experiments runs are shown in Table 9. Normalised data for t = 1s is depicted in Table 10 in order to provide a fair comparison between the various media players.

Audio	Hardware Energy Consumption (J)					Application (J)	Total Energy Consumption (J)	Time (s)
	CPU (J)	Monitor (J)	Disk (J)	Base (J)	Total Hardware Energy Cnsumption (J)			
KMPlayer	1312.99	4593.31	8.38	33228.59	39143.26	433.60	39576.86	2215.24
VLC	1694.42	2870.96	11.36	37110.92	41687.66	1612.72	43300.38	2474.06
WMP	1323.99	15806.66	10.34	37046.85	54187.84	1257.62	55445.45	2469.79
Video								
KMPlayer	2387.37	29492.64	41.48	55298.70	87220.19	2305.68	89525.88	3686.58
VLC	2347.85	29278.87	8.01	54897.88	86532.61	2264.76	88797.37	3659.86
WMP	1906.01	26106.89	161.64	48950.41	77124.95	1124.46	78249.40	3263.36

Table 9: Total hardware and application energy consumption for running several media players in Windows

Audio (t=1s)	CPU (J/s)	Monitor (J/s)	Disk (J/s)	Base (J/s)	Total Hardware Energy Cnsumption (J/s)	Application (J/s)	Total Energy Consumption for t=1s (J)
KMPlayer	0.59	2.07	0.00	15.00	17.67	0.20	17.87
VLC	0.68	1.16	0.00	15.00	16.85	0.65	17.50
WMP	0.54	6.40	0.00	15.00	21.94	0.51	22.45
Video (t=1s)	CPU (J/s)	Monitor (J/s)	Disk (J/s)	Base (J/s)	Total Hardware Energy Cnsumption (J/s)	Application (J/s)	Total Energy Consumption for t=1s (J)
KMPlayer	0.65	8.00	0.01	15.00	23.66	0.63	24.28
VLC	0.64	8.00	0.00	15.00	23.64	0.62	24.26
WMP	0.58	8.00	0.05	15.00	23.63	0.34	23.98

Table 10: Normalised hardware and application energy consumption for running several media players in Windows (for t=1s)

The results revealed in Table 10 suggest that the energy consumption for playing audio and video files seem to be similar for the KMPlayer and VLC Media Player. However, the energy consumption for running the audio file by the Windows Media Player seems to be the highest. However, it is the contrary for running the video file. It could be noted that the audio finding is consistent with Techradar's (2010) finding which shows that the VLC player is more energy efficient than the Windows Media Player. In order to produce more valid results, it is necessary to repeat the entire set of experiments for at least 9 times in order to obtain a more reliable average value. Figures 4 and 5 are plotted based on values in Table 10.

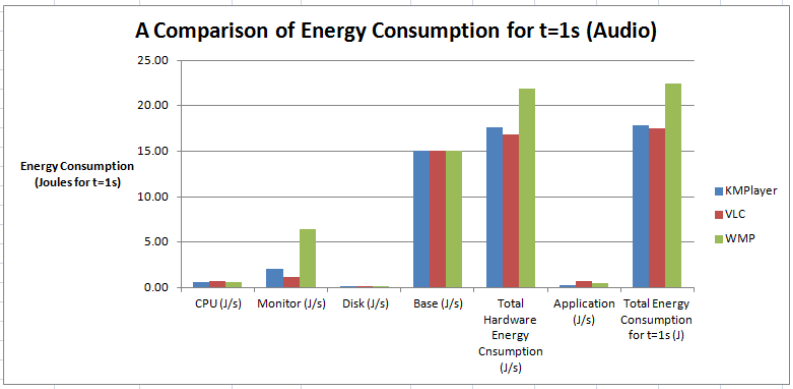

Figure 4: A comparison of energy consumption for playing audio files by several media players in Windows

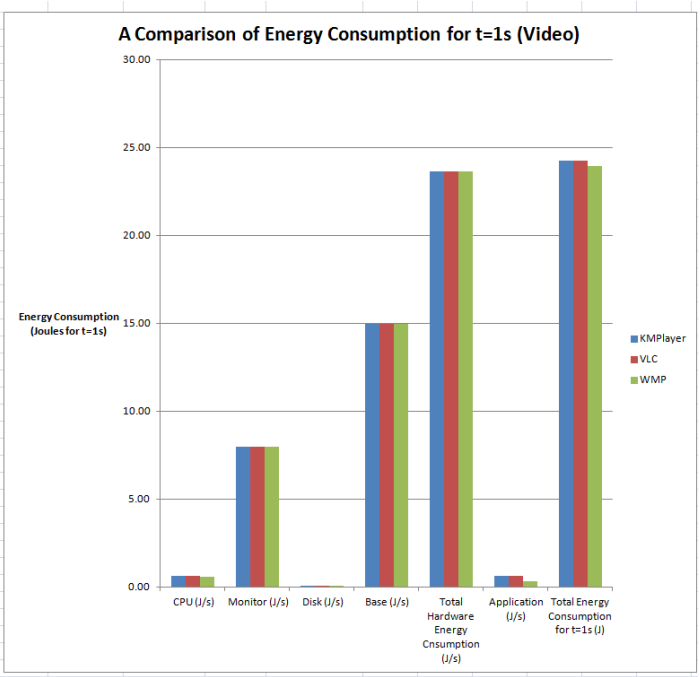

Figure 5: A comparison of energy consumption for playing video files by several media players in Windows

From Figures 4 and 5, it shows that the energy consumption for the disk, CPU and application is very low compared to the monitor and base. The total hardware energy consumed by KMPlayer and VLC for running the audio file seems to be similar while WMP consumes the highest amount of energy. On the other hand, the total hardware energy for running a video file by the three players seems to be similar. The only difference is in the application energy consumption where WMP seems to have consumed the least amount of energy. Further experiments will be necessary to verify this finding.

Conclusion

In this integrated research, we have demonstrated the different tools that could be used to measure the power and battery consumption of various web browser and stand alone applications. The Joulemeter has been employed for the measurement of power consumption by the hardware and software in laptops with Windows operating system while the inbuilt battery status checker has been used to measure the battery consumption in an Apple ipad Air2 with IOS operating system. Some of the results in the experiments (particularly Section 4.1) conducted confirm the findings in existing research. However, further experiments are necessary to verify the findings in Section 4.2 and 4.3 by taking into consideration the experiment critique that have been discussed. This could be completed with additional use of other measure tools that have been discussed in Section 2, and also external measurement devices (e.g. multi-meter, etc…) which would yield more holistic experimental results.

References

Ahmed, F., Mahmood , H. & Aslam, A. (2014). *Green Computing and Software Defects in Open Source Software: An Empirical Study.* Lahore, IEEE.

Banerjee, N., Rahmati, A., Corner, M., Rollins, S. and Zhong, L. (2007). Users and batteries: Interactions and adaptive management in mobile systems**. In proceedings of the 9th International Conference on ubiquitous computing, Zurich, Switzerland**. Url: http://dl.acm.org.ezproxy.leedsbeckett.ac.uk/citation.cfm?id=1771592.1771605&coll=DL&dl=ACM&CFID=646855466&CFTOKEN=15339289, accessed date: 3 March 2015.

Bourdon, A., Noureddine, A., Rouvoy, R. & Seinturier, L. (2013). *PowerAPI: A Software Library to Monitor the Energy Consumed at the Process-Level.* Url: http://ercim-news.ercim.eu/en92/special/powerapi-a-software-library-to-monitor-the-energy-consumed-at-the-process-level, [Accessed 16 April 2015].

Carroll, A. & Heiser, G. (2010). *An analysis of power consumption in a Smartphone.* Berkerly, USENIX.

Do, T., Rawshdeh, S. & Shi , W. (2009). *pTop: A process-level Power Profiling Tool.* Big Sky, SIGOPS.

EEcoMark, 2011. *An overview of BAPCo EEcoMark v2,* s.l.: BAPCo.

Flinn, J. & satyanarayanan, M. (1999). *PowerScope: A tool for profiling the Energy Usage of Mobile Applications.* New Orleans, IEEE, pp. 2-10.

Greenhouse Protocol. (2012). GHG Protocol Product Life Cycle Accounting and Reporting Standard ICT Sector Guidance, Chapter 7: 7 Guide for assessing GHG emissions related to energy used by software. Url: http://www.ghgprotocol.org/files/ghgp/Chapter_7_GHGP-ICT%20Software%20v2-2%2010MAR2012.pdf , accessed date: 30th June, 2015.

Intel. (2010). *Intel Developer Zone,* url: https://software.intel.com/en-us/articles/intel-energy-checker-sdk/. Date accessed: 23 April 2015].

Kaliterre. (n.d.). Green IT, url: http://www.kaliterre.fr/, accessed dat: 30th June, 2015.

Kansal, A. & Zhao, F. (2008). Fine-grained energy profiling for power-aware application design. *ACM SIGMETRICS Performance Evaluation Review,* 36(2), pp. 26-31.

Krintz, C., Wen, Y. and Wolski, R. (2004) Application level prediction of battery dissipation**. International Symposium on low power electronics and design,** url: http://ieeexplore.ieee.org.ezproxy.leedsbeckett.ac.uk/stamp/stamp.jsp?tp=&arnumber=1349340, access date: 3 March 2015.

Lewis, A. W., Tzeng, N.-F. & Ghosh , S. (2012). Runtime energy consumption estimation for server workloads based on chaotic time-series approximation. *ACM Transaction on Architecture and Code Optimization*, September.

Liu, X., Shenoy, . P. & Corner, M. D. (2008). Chameleon: Application-Level Power Management. *IEEE Transactions on Mobile Computing,* 7(8), pp. 995-1010.

Mahmoud, S. S. & Ahmad, I. (2013). A Green Model for Sustainable Software Engineering. *Internatioanl Journal of Software Engineering and its Applications,* 7(4), pp. 55-74.

McIntire, D., Stathopoulos, T. & Kaiser , W. (2007). *ETOP: Sernsor network application energy profiling on the LEAP2 platform.* New York, ACM IPSN '07 Proceeding of the 6th international conference on Information processing in sensor network .

Microsoft Research. (2011). Joulemeter: The User Manual, url: http://research.microsoft.com/en-us/downloads/fe9e10c5-5c5b-450c-a674-daf55565f794/usermanual.pdf , accessed date: 30th June, 2015.

Narayanan, D. (2005) Software Power Measurement. **Microsoft Research,** url: http://research.microsoft.com/apps/pubs/default.aspx?id=70166, accessed date: 15 March 2015.

Noureddine, A., Rouvoy, . R. & Seinturier, L. (2013). A review of energy measurement approaches. *ACM SIGOPS Operating System Review,* 47(3), pp. 42-49.

Rahmati, A., Qian, A. and Zhong, L. (2007). **Understanding human batteries interaction on mobile phones.** url: www.ruf.rice.edu/~mobile/publications/rahmati07mobilehci.pdf , accessed date: 3 March 2015.

Ravi, N., Scott, J. and Iftode, L. (2008). Context aware battery management for mobile phones: In proceeding of International Conference on persuasive computing and communications (PERCOM). **IEEE Xplore Digital Library** [Online], (nd) March, pp. 224-233. Url: http://ieeexplore.ieee.org.ezproxy.leedsbeckett.ac.uk/stamp/stamp.jsp?tp=&arnumber=4517397 , access date: 4 March 2015.

Sabharwal, M. R. (2011). *Software power optimization: analysis and optimization for energy-efficient software.* New Jersey, IEEE/ACM Proceedings of the 17th IEEE?ACM international symposium on Low-power electronics and design .

Sagahyroon, A. (2006) Power consumption in handheld computers: In proceeding o8f Asia Pacific Conference on Circuit and Systems (APCCAS). **IEEE Xplore Digital Library** [Online], (nd) December, pp. 1721-1724. Url: <http://ieeexplore.ieee.org.ezproxy.leedsbeckett.ac.uk/xpl/articleDetails.jsp?arnumber=4145743 , access date: 4 March 2015.

Seo, C., Malek, S. & Medvidovic, N. (2007). *An Energy Consumption Framework for Distributed Java-Based Systems.* New York, ACM, pp. 421-424.

Silven, O. and Jyrkka, K. (2007). Observations on the power efficiency trend in mobile communication devices. **EURASIP Journal on embedded systems** [Online], (nd) March, url: http://jes.eurasipjournals.com/content/2007/1/056976/abstract, access date: 2 March, 2015.

Simunic, T., Benini, L., De Micheli, G. and Hans, M. (2000) source code optimization and profiling of energy consumption in embedded systems: In proceeding international symposium on system synthesis (ISSS 2000). **IEEE Xplore Digital Library** [Online], (nd) September, pp. 193-198. url: http://ieeexplore.ieee.org.ezproxy.leedsbeckett.ac.uk/stamp/stamp.jsp?tp=&arnumber=874049 , access date: 4 March 2015.

Techradar. (2010). *Techradar.com.* Url: http://www.techradar.com/news/computing/pc/the-ultimate-guide-to-reducing-your-pc-s-power-consumption-661978/3, access date: 17 April 2015.

The Climate Group and GeSI. (2008). SMART 2020: Enabling the low carbon economy in the information age, url: http://www.smart2020.org/_assets/files/02_smart2020Report.pdf, accessed date: 30th June, 2015.

Varrol, C. and Heiser, G. (2010). An analysis of power consumption in a smartphone, Proceeding USENIXATC'10 Proceedings of the 2010 USENIX conference on USENIX annual technical conference.

Yang, A. & Song, M. (2009). *Aggressive Dynamic Voltage Scaling for Energy-Aware video playback based on decoding time estimation.* New York, ACM EMSOFT '09 Proceeding of the seventh ACM international conference on Embedded software.