Metrics of energy consumption in software systems: a systematic literature review

Metrics of energy consumption in software systems: a systematic literature review

S Ergasheva[1,2,] I Khomyakov[1,2], A Kruglov[1,2] , G Succi1[2]

1 Innopolis University, Innopolis, Russia
2This research project is carried out under the support of the Russian Science Foundation Grant No 19-19-00623.
E-mail: a.kruglov@innopolis.ru

Abstract. The current situation when using tight time frames and frequently changing requirements when creating software dictates the need to create a system for monitoring energy consumption at any stage of production of a software product. At the first stage, we need to evaluate the state-of-the-art on this topic. To this goal, we conducted a systematic literature review. During the review more than 500 studies were observed and 124 of them were selected for detailed analysis. Among these papers, 169 metrics were derived and assessed from the point of their applicability within invasive software development process analysis. The study demonstrates the relevance of the questions posed and shows the immaturity of the area. There is no evolutionary study and the possibility of assessment at any stage of the development of a software product. The data show the importance and relevance of technical work and the importance of its further development.

1. Introduction

Given the sharp growth of IT systems and their impact on worldwide energy consumption, energy efficiency is becoming a real concern. It is estimated that the energy consumption of the ICT sector will reach 433 GW in 2020, meaning more than 14.5% of worldwide power consumption [1]. Therefore, it is essential to have precise figures of the current energy consumption of computer and mobile devices and how much of this is due to the software running on them, to understand how to reduce their power consumption and design future energy efficient equipment.

In order to provide a novelty approach on the energy efficiency assessment of the software product, we decide to focus on the development process analysis and related software metrics. For the data collection process, we will use non-invasive tools for monitoring the process of software development. A set of metrics for measuring software (source code), and the development process can be adjusted [2]. The toolkit integrates with major software development environment and office applications, so developers are not distracted from the main workflow. This technology has been successfully tested for problem analysis methodologies pair programming on software quality [3].

The project aims at building and validating a quantitative framework to guide the development and the evolution of sustainable software systems using a variety of metrics collected throughout the life-cycle of software systems, optimizing the performances of the systems under a variety of relevant factors, including efficient use of resources [4].

At the first stage of the project the metrics which describe, interpret and measure the value of resources software components of adaptive systems throughout their life cycle (design, implementation, operation) should be defined. These metrics will correlate with specific monitored resources, architectural elements and the behaviour of the system as a whole. To define a proper

metrics, a systematic analysis of the state of the art should be performed, related to the main topics of the project: i.e., resource- and sustainable-wise metrics, pervasive and non-invasive data collection instruments, quality- oriented development support tools and methods, self-adaptation techniques and policies. As far as possible and depending on the characteristics of the available literature, this task will provide an organized body of empirical evidence with reference to the main themes of concern for the project.

For identification and analysis of relevant research literature, we formulated following 4 research questions (RQs):

- What are the existing studies on green metrics (energy consumption in mobile, embedded and cloud systems)?
- What is the classification of these metrics?
- What are descriptions and limitation of these metrics?
- What are the ways of collecting selected metrics?

2. Review protocol

We defined the search strategy based on the research questions and terms that helped us in studies selection to examine. The following approaches were used to build the search terms [5]:

- Derivation of major terms from the research questions.
- Identification of alternative spellings and synonyms for major terms.
- Identification of keywords in relevant papers or books.
- Usage of the Boolean OR to incorporate alternative spellings and synonyms.
- Usage of the Boolean AND to link the major terms.

For identification of primary research papers, the search string was generated based on the PICO (Population, Intervention, Comparison, Outcomes) approach. In the result of PICO strategy we defined following terms based on the research questions. Correlation of numerical quantities and search terms defined can be seen from the table 1.

Table 1. Search queries

Search query	Papers
Software (metrics OR measurements)(energy consumption OR energy efficiency OR power consumption)(mobile AND (devices OR phones OR development))	50
Software metrics for embedded systems in power consumption	11
Metrics of (power OR energy) consumption in (embedded OR cloud) systems	51
software metrics energy consumption mobile devices -wireless -radio	16
software metrics of energy consumption in smartphones	29
software metrics of energy consumption in mobile application development	23
software (metrics OR measurement) (energy consumption OR power management) (mobile devices OR mobile phones) -wireless -radio	21
software (metrics OR measurement) ((energy OR power) AND (consumption OR management OR reduction)) ((mobile AND (devices OR phones)) OR Smartphones) -wireless -radio	10
software (metrics OR measurement) (energy consumption OR power management) (mobile devices OR mobile phones) -wireless -radio	37
software (metrics OR measurement) ((energy OR power) AND (consumption OR management OR reduction)) ((mobile AND (devices OR phones)) OR Smartphones) -wireless -radio	16
software (metrics OR measurement) of (Energy OR Power) (consumption OR management OR Efficiency OR Reduction) in ((Mobile (device OR phone OR application development)) OR Laptop OR Smartphone) -wireless -radio	16
(software AND (metrics OR measurement) AND power AND (consumption OR management) AND ((mobile AND (device OR phone OR application development)) OR smartphone) NOT wireless NOT radio)	50
(software AND (metrics OR measurement) AND ((energy AND consumption) OR (power	92

AND management)) AND (embedded AND systems))	
Software (metrics OR measurement)(energy consumption OR power management) (cloud systems)	82

To execute the search on the research studies, we selected following free access literature libraries for automatized searches, namely:

- Google Scholar
- Scopus
- IEEE Xplore
- Web of Knowledge
- Digital library ACM
- Science Direct
- ResearchGate

Primary studies identification was performed based on the Snowballing instructions proposed in [5] which includes the steps:

1. Use the papers selected in automatic and manual searches as the initial set of selected studies;
2. Based on the selected studies, check references by looking at works of authors already included, since they obviously carry out relevant research in relation to their objectives;

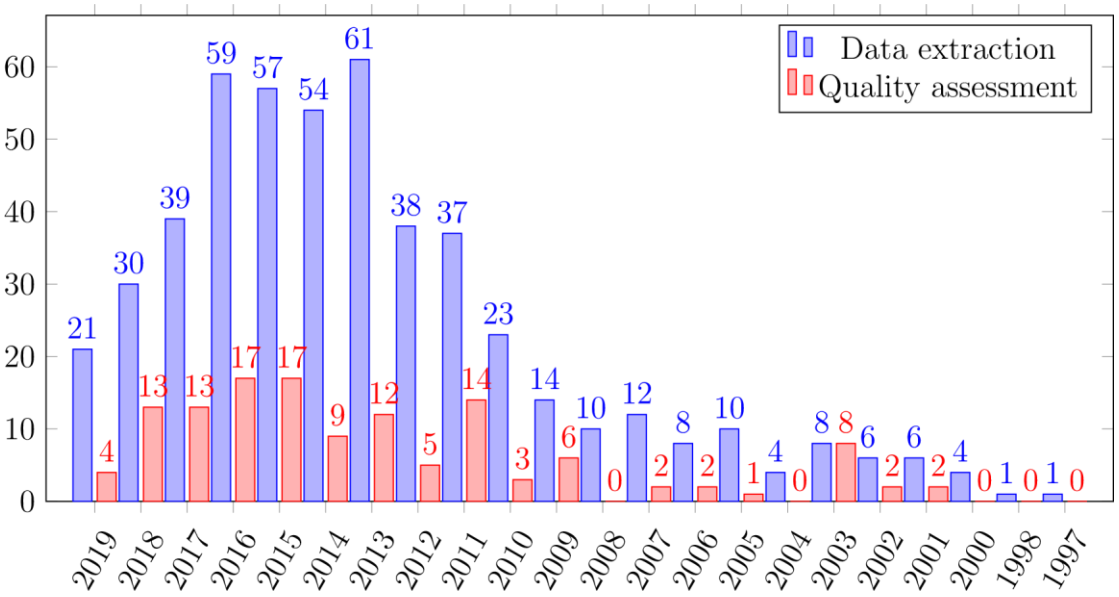

Figure 1. Papers related to the energy efficiency topic

3. Based on the set of documents found, check studies that cite the selected studies (forward snowballing).

This approach is conducted with the help of Google Scholar Database, since it captures more than individual databases. With the aim of selecting the most relevant articles for the study object, we included:

1. Articles that present Software measurements and discussion on metrics used for mobile, embedded and cloud systems;
2. Articles that contain related title, keywords and abstract to the research questions;
3. Major publications between 2000 and 2019.

During the research paper selection process, we applied exclusion criteria to eliminate duplication, not relevant and incomplete studies. If the study has been published in more than one conference or has more than one versions, then the most complete version was chosen. Furthermore, the studies that do not present the related information about the software measurements of mobile, embedded and cloud systems were not included. Besides, the patents and the studies in terms energy consumption of wireless network and radio connections were excluded from this SLR. According to the approach of

including the studies between 2001 and 2019, the representation of articles distribution by years is given in figure 1.

In the stage of quality assessment from the title, keywords and abstract, 75.4 % of primary studies were rejected, 23.6% of them were accepted and 1%was rejected for duplication. In the next stage of data collection, selected papers were completely reviewed and observed whether they do contribution in answering the research questions in the areas mobile, embedded and cloud systems. After the complete reading of the papers, they were classified into Tool, Process or Project metrics, Models and Methods, Empirical study and Best practises reviews. These classification of the papers gave complete understanding of which paper can answer which Research questions.

A Systematic Literature Review investigates metrics associated with power consumption of mobile, embedded and cloud systems. At first, the search keywords string was inserted. There were selected 504 primary studies from the digital library sources (see figure 1). Following this, the results were sorted with reading of title, keywords and abstract. We classified them in terms of suitability and relevance from 1 to 5 scale, where 124 studies were selected according to the high scale. Following this stage, complete reading of the articles was performed meaning that they answer the elaborated research questions. Besides, the studies that were not available and not applicable which discusses hardware metrics were derived separately for exclusion procedure. The ranking of selected studies according to the scale depicted in table 2.

Table 2. Experts ranking results

NotApl	1	2	3	4	5	NA
134	68	56	86	82	38	39

3. Results

In the previous section, the results of data extraction of primary studies were described. In this section the further analysis of the extracted data is discussed in order to obtain the answers to the research questions. By analyzing the existing software metrics of mobile, embedded and cloud systems towards the power consumption and their classifications, the distribution of selected studies in the areas of Cloud, Embedded consisted 23.3% and 27.5% respectively. While the relevant studies in Mobile Devices took 49.2% of overall selected studies in the research (see figure 2).

Besides, we want to find out descriptions and some limitations of the metrics encountered during the SLR. Then the types of tools for collecting these metrics will be analysed.

What are the existing studies on energy consumption in mobile, embedded and cloud systems?

The figure 1 shows the graphical distribution of the selected papers derived by years.

What is the classification of these metrics?

From the refined set of studies the exhausting set of metrics was derived. Overall number of energy efficiency metrics mentioned in literature is 169. In order to distinguish the focus of these metrics, their classification was performed. Based on the review of the studies, the metrics were classified into one of the following category:

- hard metrics, which are related to physical measurement of energy consumption by different components of a system
- code metrics, which are related to analysis of the software code itself
- run-time metrics, which are related to the dynamic analysis of the applications, e.g. analysis of the byte code
- indirect metrics, which are commonly related to the particular energy efficiency model
- process metrics, which are related to the analysis of the software development process
- others, which are focused on the specific parts of the systems' operation

The distribution of the metrics according to the given classification is given in figure 3.

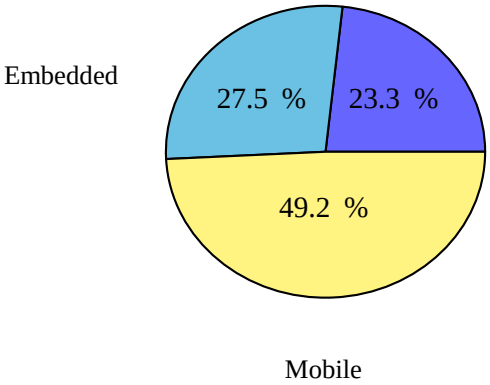

Figure 2. Papers' distribution by topic

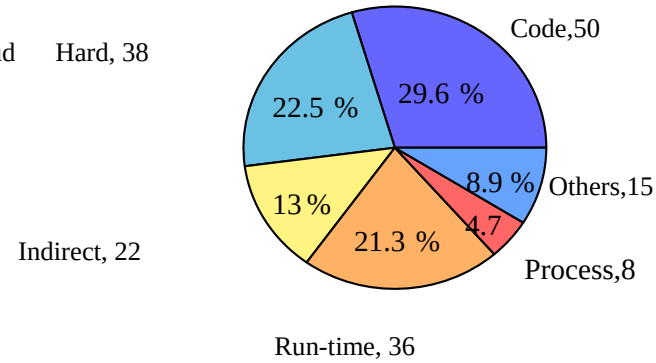

Figure 3. Analyzed metrics classification

What are descriptions and limitations of these metrics?

The goal of this SLR is to analyze existing metrics for energy efficiency assessment, which can be applied in the project. The goal of the project is to develop framework which will be focused on the analysis of the software metrics excluding direct measurement of hardware parameters, and be applicable to any mobile, embedded or cloud system. Thus, among the giving classification the only two groups suitable for software framework are code and process metrics. The description of the derived software and process metrics are given in table 3.

Table 3. Code metrics for energy efficiency

#	Metric	Description	Source
1	Control flow-graph	Graphical representation of control flow or computation during the execution of programs or application.	[6]
2	Sorting Algorithm Type	The approach of saving through software by choosing the appropriate sorting algorithm.	[7]
3	Performance tips	Strategies that can be used to conserve and extend the battery life.	[8]
4	Software energy metric	Software energy metric (SEM) which corresponds to the number of executed assembly instructions and the number of memory accesses to the date memory.	[9]
5	Nesting function	A function which is defined withing another function, the enclosing function. It was assigned for evaluation of software designs energy consumption of hierarchical measures(EIC, MAC).	[9]
6	Sequencing function	It is calculated based on the control flow-graph. $I(F;F_1;F_2;\ldots;F_n) = \sum I(F_i)$	[9]
7	Prime function	It is calculated based on the control flow-graph and defined as the number of basic mathematical operators (excluding division) plus the number of memory accesses in the statement.	[9]
8	Number of executed instructions	It corresponds to instruction execution within the processor and instructions fetches from the memory.	[9]
9	Number of Immediate sub-classes of a Class	It refers to the number of direct descendants of a class.	[10]
10	Coupling between Object	Number of classes coupled to a specific class.	[10]
11	Lack of Cohesion on Methods	Number of pairs of member functions with shared instance variables subtracted by the total pairs of member functions without shared instance variables.	[10] [11]
12	Efferent Couplings	Classes in the package that depend on the other packages.	[10][11]
13	Response For a Class	Number of methods of a specific class plus number of other class methods called by the methods of this class.	[10]
14	Abstractness	Ratio of the number of abstract classes in the analyzed package to the total number of classes in the analyzed package.	[10] [11]
15	Instability	Ratio of efferent coupling to total coupling.	[10] [11]
16	Attribute Hiding Factor	These metrics measure how properties like variables and methods are encapsulated in a class. A private property is completely hidden.	[10]

17	Coupling Factor	Number of actual method overrides divided by the maximum number of possible method overrides.	[10]
18	Weighted Methods per Class	Aggregate of complexities of methods in a class. When complexities are equal, number of methods defined in each class.	[10] [11]
19	Loops	For and While loops in the source code.	[12]
20	Code smell	Internal Setter, Leaking Thread, Member ignoring method and Slow Loop consume up till 87 times more than methods affected by other code smells.	[12]
21	Cyclomatic Complexity	It counts the number of flows through a piece of code.	[11]
22	LOC	Lines of code.	[13]
23	Invocation	The number of executions of a program or function.	[8]
24	Field access	Number of accesses to object fields.	[8]
25	Array length	Number of accesses to the array length operation.	[8]
26	Number of Static Attributes	The number of attributes that contain the same value for each object (all instances of the class).	[11]
27	Specialization Index	Indicate the average of the specialization index, defined as NORM * DIT / NOM	[11]
28	Normalized distance	This metric from Main Sequence calculate by $\mid A + I - 1 \mid$, this number should be small, close to zero for good packaging design.	[11]
29	Depth of Inheritance tree	(DIT) is the distance from class Object in the inheritance hierarchy.	[11]
30	Number of Packages	Number of packages in the code.	[11]
31	Number of overridden Methods	This metric shows the total number of methods in the selected scope that are overridden from an ancestor class	[11]
32	Number of Static Methods	Number of methods whose signatures differ is known as overloading. Single return value.	[11]
33	Nested Block Depth	This metric is the number of statement blocks that are nested due to the use of control structures (branches, loops)	[11]
34	Number of Parameters	Number of formal parameters being passed onto the function	[11]
35	Development productivity	How many outputs can be produced per unit time. It is measured as size divided by time.	[14]
36	Defect density	Number of defects found per unit size of product. It is measured as Defects divided by size.	[14]
37	Defect removal efficiency	The percentage of defects removed during the phase. Calculated as Defects removed within the phase divided by total defects.	[14]
38	Cost performance index	This metric indicates the degree to which actual time spent is meeting time commitments	[14]
39	Size estimation errors	Indicator to which degree the estimate matches the actual size: (Actual size - Planned size)/Planned size.	[14]
40	Life cycle Cost Metric	Relation between efforts needed for redesigning an application for energy consumption optimization vs. the potential energy savings	[15]
41	Quality of Service Metric	It includes the following parameters: Availability, Throughput, Response Time, Process Time/Job Duration, Reliability, Recoverability, Workload, Application Performance Metric	[15]

What kind of tools are used for collecting these metrics? A number of tools for collecting energy consumption metrics were found from selected studies. Some of the found tools are used while collecting metrics of the product, process or both of them. Found tools and their descriptions are provided.

MEMT - It allows the development of plugins for automatic tuning where the design makes a reference to Periscope Tuning Framework, as described in the paper [16].

PETrA - Power Estimation Tool for Android is software-based tool developed to measure the power consumption of mobile apps at a method-level granularity [17].

PUPiL - a hybrid software/hardware power capping system. It is based on a novel decision framework. To ensure a power cap, PUPiL navigates nodes - choices about how to use a particular resource in a decision framework [18].
JADE - a system that adds sophisticated energy-aware computation offloading capabilities to Android apps. Jade monitors device and application status and automatically decides where code should be executed [19].
GEMMA - GUI Energy Multi-objective optiMization for Android apps. It generates color palettes using a multiobjective optimization technique, which produces color solutions optimizing energy consumption and contrast while using consistent colors with respect to the original color palette [20].
Green Advisor - a first of its kind tool that systematically records and analyzes an application's system calls to predict whether the energy-consumption profile of an application has changed [21].
ePRO-MP - profiles and optimizes both performance and energy consumption of multi-threaded applications running on top of Linux for ARM11 MPCore-based embedded systems. It does not require power consumption equipment and uses regression-based energy model [22].
LEMON - Lemon is a server/client based monitoring system for a very large scale infrastructure. On every monitored node, a monitoring agent is launched and this communicates using a push/pull protocol with sensors [23].
ANEPROF - Android systems's tool that can obtain function-level power distribution and distinguish the power consumption among threads, Java methods, and JVM services and also addresses how to correctly associate power logs with Android system's behaviours [24].

4. Discussion

This section represents various findings of our Systematic Literature Review(SLR) and draw some conclusions. The 504 papers in the areas like Mobile, Embedded and Cloud systems were under study consideration that investigate energy consumption. The Systematic Literature Review's aim was to identify Green metrics and their classification with descriptions and limitations, and the types of tools to collect these metrics. The results of the review shows that the studies that investigate Energy efficiency in Mobile systems constitutes more than Embedded and Cloud systems. In fact, that as young fields, energy efficiency surveys in Cloud and Embedded systems require more research studies where it became common in recent years to pay more attention to them.
From 124 studies 17 of them focused on experimenting existing best practises and conducting empirical studies for better understanding of existing techniques efficiency of energy metrics. In the paper [13], there were taken into consideration six popular mobile applications, where the authors observed whether the battery of the mobile device can last up to approximately an extra hour if the applications are developed with energy-aware practices. While in [25], there was conducted an empirical study of the effects of 18 code obfuscations on the amount of energy consumed by executing a total of 21 usage scenarios spread across 11 Android applications on four different mobile phone platforms.
Whereas, 54 studies such as [26] and others were concentrated on proposing new or improved Models and Frameworks for mapping software design to power consumption. Where the importance of energy efficiency was described in early analysis of software design and have significant impact on energy conservation achievement.
Furthermore, the tools to collect Green metrics were considered to capture types of Tools in the observed research papers. In the paper [27], the tool under consideration the dynamic energy consumption estimator showed high estimation accuracy and effectiveness as it selected the suitable energy consumption estimator for various mobile application programs. Moreover, the paper [21] considered precision, recall, specificity, and F-measure, proposing the GreenAdvisor tool that uses the application's system-call profile to warn developers about possible changes in the application's energy consumption profile by using Rule of Thumb. However, our study did not consider the complete understanding of Tools' functions and working conditions.
The main concentration during the SLR was to identify the metrics related to the software code metrics defined in the study [28] excluding hardware metrics. In the paper [29], methods affected by 4 code smell types, Internal Setter, Leaking Thread, Member ignoring method and Slow Loop consume up till 87 times more than methods affected by other code smells. Refactoring of these code smells

reduced energy consumption in all situations. While in [30], they explored the memory usage and the bytecode executed during an operation. They found that choosing the wrong Collections type, as indicated by their profiles, can cost even 300% more energy than the most efficient choice.

5. Conclusion

Systematic Literature Review conducted within this project clearly shows the urgency of the software energy efficiency issue and its proper and operative analysis in particular. In general, all studies could be devoted into the real measurement and model-based measurement, as it was proposed in [24]. Direct measurements involve usage of third-party hardware tools to read actual energy consumption by different modules and components of the software or entire hardware-software systems.

Metrics, related to this area of energy efficiency analysis were separated into hard metrics group. These metrics are not in scope of the given project, as far as our goal is to develop framework for invasive analysis of the software development process. Thus, the product and process metric, related to the software quality analysis should be the main focus of further research. Based on the analysis of these metrics (table 3) it could be concluded that the group of process metrics has not properly observed yet. One of the reason for it is the absence of the tool for sufficient analysis of the development process. Usually, such process involves participation of the developer [31], and we faced with the problem of subjective measurement and time costs increasing due to the task switching. Developing of the framework which provides an invasive measurement of the development process could result in bunch of metrics showing development effort patterns. The combination of the developers' behavior patterns and code metrics could result in highly precise model for the in situ analysis of the developing product. Thus, the development of framework for invasive process measurement as well as developing of the model for mapping process and code metrics to the actual energy efficiency of the software product will be the subject of the further research.

6. References

[1] Vereecken W, Heddeghem W, Deruyck M, Puype B, Lannoo B, Joseph W, Colle D, Martens L and Demeester P 2011 Power consumption in telecommunication networks: Overview and reduction strategies *IEEE Communications Magazine* **49** 62–9

[2] Pedrycz W, Succi G, Sillitti A and Iljazi J 2015 Data description: A general framework of information granules *Knowledge-Based Systems* **80** 98–108

[3] Coman I D, Sillitti A and Succi G 2008 Investigating the usefulness of pair-programming in a mature agile team *Lecture notes in business information processing* (Springer Berlin Heidelberg) pp 127–36

[4] Corral L, Sillitti A and Succi G 2015 Software assurance practices for mobile applications - A survey of the state of the art *Computing* **97** 1001–22

[5] Kitchenham B, Brereton O P, Budgen D, Turner M, Bailey J and Linkman S 2009 Systematic literature reviews in software engineering a systematic literature review *Information and Software Technology* **51** 7–15

[6] Hao S, Li D, Halfond W G J and Govindan R 2013 Estimating mobile application energy consumption using program analysis *Proceedings of the 2013 international conference on software engineering* ICSE '13 (Piscataway, NJ, USA: IEEE Press) pp 92–101

[7] Verma M and Chowdhary K R 2018 Analysis of energy consumption of sorting algorithms on smartphones *SSRN Electronic Journal*

[8] Li D and Halfond W G J 2014 An investigation into energy-saving programming practices for android smartphone app development *Proceedings of the 3rd international workshop on green and sustainable software - GREENS 2014* (ACM Press)

[9] Chatzigeorgiou A and Stephanides G 2002 *Software Quality Journal* **10** 355–71

[10] Bangash A A, Sahar H and Beg M O 2017 A methodology for relating software structure with energy consumption *2017 IEEE 17th international working conference on source code analysis and manipulation (SCAM)* (IEEE)

[11] Keong C K, Wei K T, Ghani A A A and Sharif K Y 2015 Toward using software metrics as indicator to measure power consumption of mobile application: A case study *2015 9th malaysian software engineering conference (MySEC)* (IEEE)
[12] Vasile C V, Pattinson C and Kor A-L 2018 Mobile phones and energy consumption *Green IT engineering: Social, business and industrial applications* (Springer International Publishing) pp 243–71
[13] Cruz L and Abreu R 2017 Performance-based guidelines for energy efficient mobile applications *2017 IEEE/ACM 4th international conference on mobile software engineering and systems (MOBILESoft)* (IEEE)
[14] Gwak T-H and Jang Y-J 2006 An empirical study on sw metrics for embedded systems
[15] Kipp A, Jiang T and Fugini M 2011 Green metrics for energy-aware IT systems *2011 international conference on complex, intelligent, and software intensive systems* (IEEE)
[16] Liu W, Zhou J, Gong B, Dai H and Guo M 2018 Improving the energy efficiency of data-intensive applications running on clusters *International Journal of Parallel, Emergent and Distributed Systems* 1–14
[17] Nucci D D, Palomba F, Prota A, Panichella A, Zaidman A and Lucia A D 2017 Software-based energy profiling of android apps: Simple, efficient and reliable? *2017 IEEE 24th international conference on software analysis, evolution and reengineering (SANER)* (IEEE)
[18] Zhang H and Hoffmann H 2016 Maximizing performance under a power cap: A comparison of hardware, software, and hybrid techniques *SIGPLAN Not.* **51** 545–59
[19] Qian H and Andresen D 2015 Reducing mobile device energy consumption with computation offloading *2015 IEEE/ACIS 16th international conference on software engineering, artificial intelligence, networking and parallel/distributed computing (SNPD)* (IEEE)
[20] Linares-Vásquez M, Bavota G, Cárdenas C E B, Oliveto R, Penta M D and Poshyvanyk D 2015 Optimizing energy consumption of GUIs in android apps: A multi-objective approach *Proceedings of the 2015 10th joint meeting on foundations of software engineering - ESEC/FSE 2015* (ACM Press)
[21] Aggarwal K, Hindle A and Stroulia E 2015 GreenAdvisor: A tool for analyzing the impact of software evolution on energy consumption *2015 IEEE international conference on software maintenance and evolution (ICSME)* (IEEE)
[22] Wonil C, Hyunhee K, Wook S, Jiseok S and Jihong K 2009 EPRO-mp: A tool for profiling and optimizing energy and performance of mobile multiprocessor applications *Scientific Programming* **17** 285–94
[23] Marian B, Ivan F, Nicholas H, Hector L T, Daniel L, Miroslav S and Denis W 2011 LEMON - LHC era monitoring for large-scale infrastructures *Journal of Physics: Conference Series* **331** 052025
[24] Chung Y-F, Lin C-Y and King C-T 2011 ANEPROF: Energy profiling for android java virtual machine and applications *2011 IEEE 17th international conference on parallel and distributed systems* (IEEE)
[25] Sahin C, Wan M, Tornquist P, McKenna R, Pearson Z, Halfond W G J and Clause J 2016 How does code obfuscation impact energy usage? *Journal of Software: Evolution and Process* **28** 565–88
[26] Mendonça J, Andrade E and Lima R 2019 Assessing mobile applications performance and energy consumption through experiments and stochastic models *Computing*
[27] Lim K-H and Lee B-D 2014 History-based dynamic estimation of energy consumption for mobile applications *16th international conference on advanced communication technology* (Global IT Research Institute (GIRI))
[28] Gurbuz H G and Tekinerdogan B 2016 Software metrics for green parallel computing of big data systems *2016 IEEE international congress on big data (BigData congress)* (IEEE)
[29] Palomba F, Nucci D D, Panichella A, Zaidman A and Lucia A D 2019 On the impact of code smells on the energy consumption of mobile applications *Information and Software Technology* **105** 43–55

[30] Hasan S, King Z, Hafiz M, Sayagh M, Adams B and Hindle A 2016 Energy profiles of java collections classes *Proceedings of the 38th international conference on software engineering - ICSE 16* (ACM Press)
[31] Anon 2009 *The personal software process sm (psp sm) body of knowledge (version 2.0)* (Carnegie Mellon University)

Acknowledgements
This research project is carried out under the support of the Russian Science Foundation Grant № 19-19-00623.