This article has been accepted for publication in IEEE Software but has not yet been fully edited.
Some content may change prior to final publication.

What do programmers know about the energy consumption of software?

Candy Pang, Abram Hindle, Bram Adams, Ahmed E. Hassan

Abstract

Traditionally, programmers have received a wide range of training on programming languages and methodologies, but rarely about software energy consumption. Yet, the popularity of mobile devices and cloud computing require increased awareness about software energy consumption. On a mobile device, computation is often limited by the battery life. Under the demands of cloud computing, data centers struggle to reduce energy consumption through virtualization and data center infrastructure management (DCIM) systems. Efficient energy consumption of software is increasingly becoming an important non-functional requirement for programmers. However, are programmers knowledgeable enough about software energy consumption? Do programmers base their implementation decision on popular beliefs? In this article, we survey over 100 programmers for their knowledge of software energy consumption. We find that programmers have limited knowledge about energy efficiency, lack knowledge about the best practices to reduce energy consumption of software, and are often unsure about how software consumes energy. Education about the importance of energy effective software will benefit the programmers. Our results highlight the need for training about energy consumption and efficiency.

Keywords: *software engineering, software energy consumption, energy efficiency, software power consumption, power usage, practitioners*

I. Introduction

With the popularity of mobile computing and the advent of large scale cloud deployments, the non-functional requirement of minimizing software energy consumption becomes a concern for software systems. For mobile devices, energy consumption affects battery life and limits device usage. For data centers, energy consumption limits the number of machines that can be run and cooled. Today, for every $1.00 spent on new hardware, an additional $0.50 is spent on energy and cooling, more than twice the amount five years ago. Data centers at their energy and cooling thresholds are unable to support new server deployments, a fact that severely limits the expansion of IT resources [6]. Energy consumption is sometimes called power consumption by consumers, as energy and power are related [5]. Energy is usually measured in Joules (J). Power, measured in watts (W), is the rate of energy conversion, transfer or consumption. Electricity providers charge users by their energy consumption measured in Kilowatt-hours (kWh) [14].

We argue that the demand for energy-efficient computing is not reflected in programmers' education, training, or knowledge. The training of programmers often focuses on methodologies such as object-oriented programming, and non-functional requirements such as performance. Performance optimization is often considered as a substitution of energy optimization, since a faster system likely consumes less energy. Although this is a step in the right direction, it is not at all sufficient and sometimes even incorrect. For instance, parallel processing might improve performance by reducing calculation time. However, saving and restoring execution context, scheduling threads and losing locality might end up consuming more resources than sequential processing [4].

PeerJ PrePrints

PeerJ PrePrints | https://dx.doi.org/10.7287/peerj.preprints.886v2 | CC-BY 4.0 Open Access | rec: 27 Jul 2015, publ: 27 Jul 2015
Digital Object Indentifier 10.1109/MS.2015.83 0740-7459/$26.00 2015 IEEE

This article has been accepted for publication in IEEE Software but has not yet been fully edited.
Some content may change prior to final publication.

We want to understand programmers' level of knowledge and awareness regarding software energy efficiency and consumption. Hence, we qualitatively and quantitatively survey programmers about their corresponding knowledge. This survey follows up on the results of an initial analysis on StackOverflow's [1] energy-related questions. The initial analysis shows that programmers have many energy-related questions, but rarely get appropriate advice [11]. To gain more tangible evidence and concrete insight into this paradox, we survey programmers to understand their level of knowledge regarding software energy consumption and efficiency. In particular, this paper addresses the following research questions:

RQ1 Are programmers aware of software energy consumption?

RQ2 What do programmers know about reducing the energy consumption of software?

RQ3 What is the level of knowledge that programmers possess about energy consumption?

RQ4 What do programmers think causes spikes in software energy consumption?

Our survey shows that programmers have low awareness about energy efficiency in the software context, and lack knowledge about reducing the energy consumption of software. Our results highlight the need for training about energy consumption and efficiency. We hope that our work would inspire the creation of energy consumption aware tools and promote the urgent need for education on software energy efficiency. Full survey details, data and analysis can be found online [10]
`http://webdocs.cs.ualberta.ca/~hindle1/2014/green-programmers/`.

The remainder of the paper details the setup of our survey, and discusses the results of our survey with respect to the aforementioned research questions.

II. Survey setup

An anonymous online survey [10] was created with 13 questions regarding programmers' demographics and their knowledge about software energy consumption, using the consumer-friendly term 'power consumption'. The survey also invited respondents to participate in an optional follow-up interview.

The survey had four phases:

1. Respondent demographics (Q1-Q3): Categorize survey respondents by experience, skill level and programming language.
2. Quantitative questions (Q4-Q11): Quantify knowledge through questionnaires.
3. Qualitative questions (Q10b, Q12, Q13): Quantify knowledge through open-ended questions.
4. Qualitative follow-up interviews: Interview respondents through open ended questions.

The first three questions are categorization questions, which classify respondents into different categories.

1. How many years of programming experience do you have?
2. How would you rank your programming skill?
   (Choose between: beginner, intermediate and advanced.)
3. What is your most proficient programming language?

The next two questions are skill-and-knowledge questions, to evaluate respondents' current knowledge in software energy consumption.

4. On a desktop computer, please rank the software power consumption level of the following components (CPU, Hard Disk Drive, Memory, Network, and Screen & GPU).

PeerJ PrePrints | https://dx.doi.org/10.7287/peerj.preprints.886v2 | CC-BY 4.0 Open Access | rec: 27 Jul 2015, publ: 27 Jul 2015
Digital Object Indentifier 10.1109/MS.2015.83 0740-7459/$26.00 2015 IEEE

This article has been accepted for publication in IEEE Software but has not yet been fully edited.
Some content may change prior to final publication.

5. On a mobile device, please rank the software power consumption level of the following components (CPU, Data Storage Device, Memory, Network, and Screen & GPU).

The next six yes-no questions gather information about respondents' experience in software energy consumption. If a respondent answers yes to question 10, additional space is provided for textual responses.

6. Do you take power consumption into account when developing software?
7. Is minimizing power consumption a requirement or concern of your software?
8. Have users complained about the power consumption of your software?
9. Have you modified your software to reduce power consumption?
10. Have you measured the power consumption of your software?
    If yes, how do you measure power consumption of your software?
11. Would power consumption be one of your decision factors when choosing a mobile development platform?

The last two are open-ended questions for respondents to freely express their knowledge and experience.

12. What software functions have higher power consumption?
13. How would you improve power efficiency of your software?

Survey invitations were posted in numerous programming related Reddit [12] sub-groups (subreddits) between August 20, 2013 and September 4, 2013 to invite programmers to participate. 122 responses were received and 4 follow-up interviews were conducted.

III. Analysis

The survey respondents self-identified as programmers. 37 out of 121 (31%) respondents use C or C++, while 84 out of 121 (69%) respondents use C#, Java, JavaScript, Perl, PHP, Python or Ruby. According to the TIOBE Index on November 2014 [13], these programming languages accounted for 54% of existing programs, and likely represent more than half of the software running in data centers. In addition, Java is the primary language for Android and BlackBerry, while C# is the primary language for Windows Phone.

RQ1 Programmers have limited software energy efficiency awareness

Our survey results show that energy efficiency is often not addressed nor requested by users. Only 22 out of 122 (18%) respondents claimed to take energy consumption into account when developing software. In terms of energy efficiency expectation, only 17 out of 122 (14%) respondents consider minimizing energy consumption a requirement. When asked whether they actively addressed energy consumption, 26 out of 122 (21%) respondents said that they actually modified software to reduce energy consumption. These survey results show that the majority of programmers are not asked to address software energy consumption and only few have experience in modifying software to improve its energy efficiency.

An interviewee indicated that clients "care first and foremost about speed of development, and secondly about reasonable quality and performance." This suggests that the lack of attention to energy consumption of software may be an issue of priorities.

With only 18% of the respondents taking into account energy consumption during software development, the survey shows that programmers are either not aware of, or not asked to address energy

PeerJ PrePrints | https://dx.doi.org/10.7287/peerj.preprints.886v2 | CC-BY 4.0 Open Access | rec: 27 Jul 2015, publ: 27 Jul 2015
Digital Object Indentifier 10.1109/MS.2015.83 0740-7459/$26.00 2015 IEEE

This article has been accepted for publication in IEEE Software but has not yet been fully edited.
Some content may change prior to final publication.

efficiency. An interviewee mentioned that "1W would be a lot of power for a mobile phone, [but] it's absolutely negligible in comparison to other household appliances." 1W consumed by a mobile device may be negligible on the personal level. However, on the global level, energy consumed by all mobile devices and data centers multiplies. In 2006, 6000 data centers in the USA reportedly consumed 61 billion kWh of energy costing $4.5 billion dollars [8].

Similarly, software users and clients are not aware about software energy consumption. When asked whether their users complain about the energy consumption of their software, only 4 out of 122 (3%) respondents answered yes. Our survey results confirmed Khalid et. al.'s [7] finding that mobile application users have low awareness about resource usage. Khalid's results show that resource-related complaints (application reviews), including energy consumption complaints, rank 12th out of 12 types of complaints in terms of their frequency. Although users do not complain about resource consumption frequently, Khalid's results also show that resource-related complaints have a negative impact on users, which means that, despite their low frequency, such complaints are highly troubling. If customers and clients are not asking for energy efficient software, then programmers are less likely to address the energy efficiency of their software. Hence appropriate public education and expanded awareness among clients and programmers about energy consumption of software is needed.

> Only 18% of the respondents take energy consumption into account during software development.

RQ2 Programmers lack of knowledge about reducing the energy consumption of software

For programmers to reduce the energy consumption of software, they have to start with measuring the energy consumption of their software. When asked whether they measure the energy consumption of their software, only 12 out of 122 (10%) respondents answer yes. 15 out of 122 (12%) respondents indicate that software energy consumption can be measured through power meter, battery, power supply, resource measurement, software tools and CPU time.

Our results show that programmers are lacking knowledge about how to accurately measure software energy consumption. Most of the suggested methods measure the overall hardware energy consumption, instead of the fine-grained energy consumption of the actual software. In addition, mobile device batteries are not accurate in reporting actual energy that is used [5]. Li et. al. [9] also find that programmers are using "typical practices in energy measurement studies [..., which] have limitations that could introduce inaccuracy".

Measuring the energy consumption of software is a challenging task. An interviewee identified that "one has to have a proper understanding of the entire system [to] make an informed [energy consumption] analysis." Programmers have to understand the interactions between high-level and low-level components to really analyze the root causes of software energy consumption. The survey and interviews show that it is a challenge for most of the programmers to measure and optimize the energy consumption of software even when tools are available.

Another interviewee admitted, "it's more often the hardware rather than the software that we are interested in when we talk about energy consumption." While only 12 out of 122 (12%) of the respondents measure the energy consumption of their software, 79 out of 122 (65%) respondents consider energy con-

PeerJ PrePrints | https://dx.doi.org/10.7287/peerj.preprints.886v2 | CC-BY 4.0 Open Access | rec: 27 Jul 2015, publ: 27 Jul 2015
Digital Object Indentifier 10.1109/MS.2015.83 0740-7459/$26.00 2015 IEEE

This article has been accepted for publication in IEEE Software but has not yet been fully edited.
Some content may change prior to final publication.

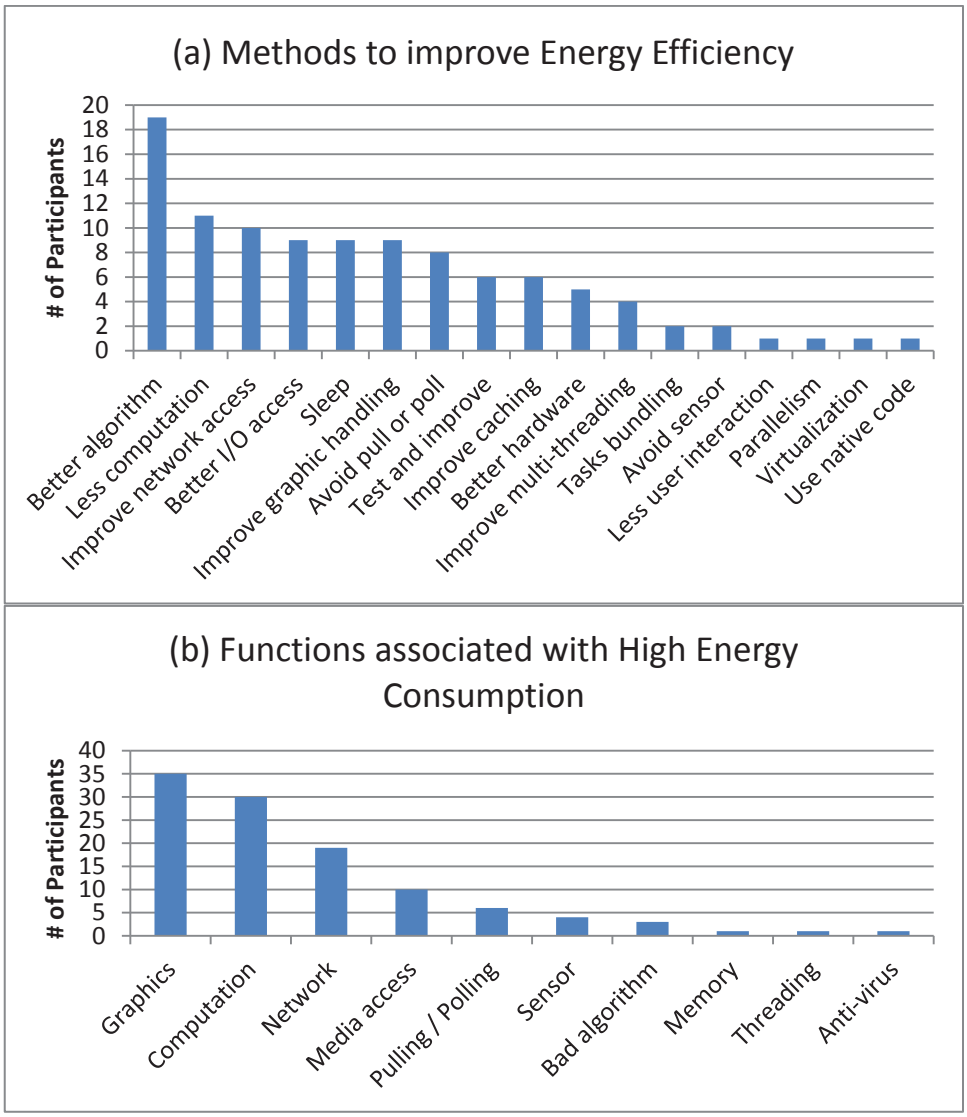

Figure 1: *Respondents textual responses to Q13 & Q12*

sumption as one of the decision factors when choosing a mobile development platform. Many programmers rely on choosing the right platform and hardware to assure the energy efficiency of their software, but do not address energy consumption in software.

To find out what programmers know about reducing the energy consumption of software, we asked the respondents to write down how they would improve the energy consumption of their software in question 13. The responses are summarized in Figure 1(a).

19 out of 122 (16%) respondents are aware that better algorithms lead to better energy efficiency, which is the most popular suggestion. Respectively, only 9% and 7% of the respondents are aware that less computation and reducing polling are strategies that reduce the energy consumption of software. The results show that the respondents have a different intuition about how to best reduce the energy consumption of

PeerJ PrePrints | https://dx.doi.org/10.7287/peerj.preprints.886v2 | CC-BY 4.0 Open Access | rec: 27 Jul 2015, publ: 27 Jul 2015
Digital Object Indentifier 10.1109/MS.2015.83 0740-7459/$26.00 2015 IEEE

This article has been accepted for publication in IEEE Software but has not yet been fully edited.
Some content may change prior to final publication.

software. Furthermore, the link between the top answer ("better algorithm") and energy consumption is not a topic often taught at universities.

10% of the respondents measure the energy consumption of their software project.

RQ3 Programmers have limited knowledge about software energy consumption

To measure programmers' competency in the energy consumption of software, we asked the respondents to rank the software energy consumption level for the following five components on desktop computers (question 4), then on mobile devices (question 5): CPU, Hard Disk Drive/Data Storage Device, Memory, Network, and Screen & GPU. Figure 2 summarizes the respondents' rankings.

For question 4, the expected ranking on desktop computer is (1) CPU; (2) Hard Disk Drive; (3) Screen & GPU; (4) Network; (5) Memory. For question 5, the expected ranking on mobile device is (1) Screen & GPU; (2) CPU; (3) Network; (4) Hard Disk Drive; (5) Memory. These expected rankings are based on current conventional wisdom, backed up by experiments that we have performed over the last three years and recent studies [3, 5, 9]. As these rankings are not written in stone, they can differ across specific hardware. This RQ focuses especially on the consistency of rankings across all respondents, independent from our expected ranking. For question 4, only 1 out of 122 (1%) respondents ranked the components in our expected order for desktop computer. For question 5, 12 out of 122 (10%) respondents ranked the components in our expected order for mobile devices.

The respondents' rankings are compared with the expected ranking via Spearman's Rank Correlation. If two rankings completely match, they will have a Spearman correlation of 1. If the two rankings are the inverse of each other, the correlation would be -1. If they are uncorrelated or not related, a correlation of 0 can be achieved. In general, positive correlations closer to 1 indicate stronger agreement. For desktop computers, the average correlation between the respondents' rankings and the expected ranking is 0.482 indicating a medium level of agreement. However, for mobile devices, the average correlation is higher at 0.75, indicating a much stronger agreement between the respondents' and the expected ranking. Furthermore, the standard deviation for the desktop computers ranking correlation is 0.25, which is higher than the 0.20 for the mobile devices.

Afterwards, we compared all respondents' rankings against each other (inter-agreement) via Spearman correlation, regardless of our expected ranking. The Spearman correlation for desktop computers is 0.3, which is lower than the 0.6 for mobile devices. Hence, respondents have less internal agreement on desktop hardware components' energy consumption than mobile's. Furthermore, the standard deviation of the Spearman correlation for desktops is 0.48, which is higher than the 0.32 for mobiles. This implies that respondents agree less about which desktop hardware components consume more or less energy. Yet respondents agree more about components' energy consumption of mobile hardware.

In other words, there is quite a bit of disagreement whether a particular component consumes more energy than another. One possible explanation could be that different types of software programmers make different assumptions about the energy consumption of hardware components. For example, game programmers interact mostly with Screen & GPU, so they are more likely to identify Screen & GPU as the most energy consuming component. Programmers may blame the most obvious component without understanding how software consumes energy.

Furthermore, programmers may have overly focused on on-screen experience of their users, i.e. on what is observable. Programmers have overwhelmingly ranked Screen & GPU as the highest energy

PeerJ PrePrints | https://dx.doi.org/10.7287/peerj.preprints.886v2 | CC-BY 4.0 Open Access | rec: 27 Jul 2015, publ: 27 Jul 2015
Digital Object Indentifier 10.1109/MS.2015.83 0740-7459/$26.00 2015 IEEE

This article has been accepted for publication in IEEE Software but has not yet been fully edited.
Some content may change prior to final publication.

PeerJ PrePrints

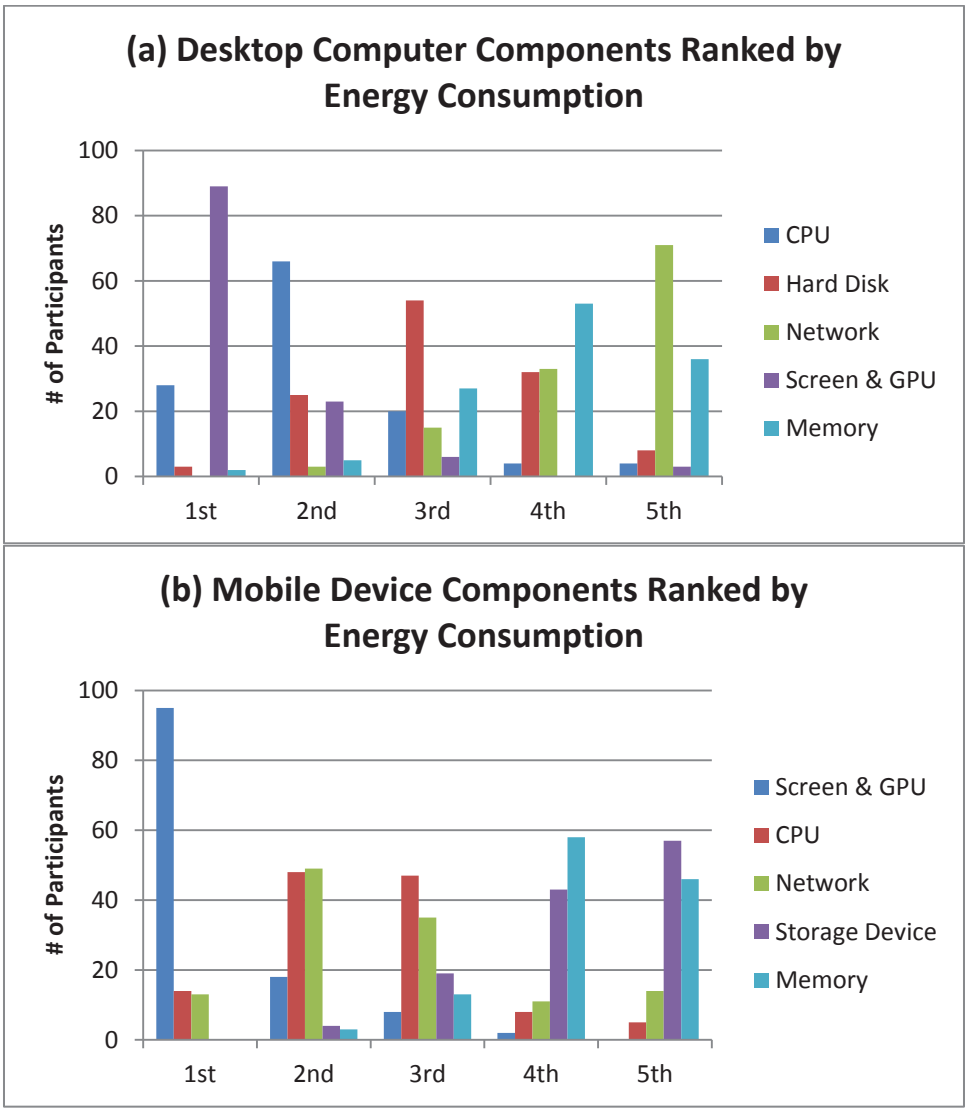

Figure 2: *Respondents' component rankings of most energy consuming components on desktop computers and mobile devices in Q4 & Q5*

consuming component for both desktop computers and mobile devices. 82 out of 122 (67%) respondents rank Screen & GPU as the highest for desktop computers, while 95 out of 122 (87%) respondents rank Screen & GPU as the highest for mobile devices. While not correct in all scenarios, Screen & GPU is often the highest energy consuming component for mobile devices.

Rank of energy consuming components is just one of the indicators. For example, the finding in RQ2 that respondents do not know how to measure software energy consumption is another indicator. Together with other factors, such as only 10% of the programmers try to measure the energy consumption of their software, we believe that programmers lack knowledge about software energy consumption.

The overall results show that programmers lack consistent knowledge regarding the energy consump-

PeerJ PrePrints | https://dx.doi.org/10.7287/peerj.preprints.886v2 | CC-BY 4.0 Open Access | rec: 27 Jul 2015, publ: 27 Jul 2015
Digital Object Indentifier 10.1109/MS.2015.83 0740-7459/$26.00 2015 IEEE

This article has been accepted for publication in IEEE Software but has not yet been fully edited.
Some content may change prior to final publication.

tion relationship between software and hardware. Nonetheless, programmers have more consistent knowledge about software energy consumption on mobile devices than on desktop computers. Therefore, it may be more effective to develop education and awareness programs/guidelines for specific domains (e.g. mobile devices, gaming).

> In general, programmers lack consistent knowledge about the energy consumption of software, however their knowledge is more consistent about such consumption on mobile devices than on desktop computers.

PeerJ PrePrints

RQ4 Programmers are unaware of the sources of software energy consumption

To learn exactly what programmers know about sources relevant to software energy consumption, we asked programmers to list high energy consuming software functions in survey question 12. Responses are summarized in Figure 1(b). Some of the high energy consuming software functions have been identified by some programmers. However, most cannot identify the high energy consuming software functions that are known from literature.

Independent from this work, Pinto et. al. [11] identified seven sources of unnecessary software energy consumption through mining StackOverflow data:

- Unnecessary resource usage
- Faulty GPS behavior
- Background activities
- Excessive synchronization
- Background wallpapers
- Advertisement
- High GPU usage

Our results match Pinto's results. Our respondents, as a team of 122 programmers, have identified most of Pinto's sources of unnecessary software energy consumption. For example, in Figure 1(b), network refers to unnecessary resource usage, sensor refers to faulty GPS behavior, threading refers to background activities, pulling/polling refers to excessive synchronization, and graphics refers to background wallpapers and high GPU usage. However, our respondents have not identified advertisements. Together, the respondents identified features match Pinto's sources relatively well and thus corroborate Pinto's findings.

However, individually, only 19 out of 122 (16%) respondents identified network data access as a source of high energy consumption, as confirmed by Li et. al. [9]. 6 out of 122 (5%) respondents identified pulling/polling for excessive synchronization as a source of high energy consumption. Only 4 out of 122 (3%) respondents identified sensor usage, such behavior that leaves GPS turned on too long, as a source of high energy consumption. Most of the respondents, 35 out of 122 (29%), identified graphics as a source of high energy consumption for functions like background wallpapers and animations as identified by Pinto et.al. [11]. The numbers show that only a small portion of programmers can identify high energy consuming functions individually.

> Many programmers are not aware of the primary sources of software energy consumption.

PeerJ PrePrints | https://dx.doi.org/10.7287/peerj.preprints.886v2 | CC-BY 4.0 Open Access | rec: 27 Jul 2015, publ: 27 Jul 2015
Digital Object Indentifier 10.1109/MS.2015.83 0740-7459/$26.00 2015 IEEE

This article has been accepted for publication in IEEE Software but has not yet been fully edited.
Some content may change prior to final publication.

IV. Limitations

The first limitation is our use of social media to recruit survey respondents, instead of directed emails or phone calls. Through social media, we can reach people in the field whom otherwise cannot be reached. More than half a million users have subscribed to programming related subreddits, which often have over 1000 concurrent users. To avoid being tagged as spam, and avoid negative reactions, we posted our survey invitation to one relevant subreddit at a time. This process has significantly lengthened the data gathering period and limited the number of respondents.

The second limitation is that we did not control for the development process used by the respondents. There are many different development processes in the IT field. An enterprise may employ the full COBIT [2] development cycle. A mid-size shop may use Agile processes. A one-man startup may do whatever is necessary. In a formal process, programmers may not have the opportunity to specify the functional or non-functional requirements. However, it will be very difficult to survey statistically significant number of programmers for each process type. Therefore, we decided to survey a board range of programmers. Future studies need to study the energy consumption knowledge of IT workers in different roles.

The third limitation is that this article focuses exclusively on programmers' software energy consumption knowledge. Software has many non-functional requirements (e.g. memory usage, performance, security, usability) and energy consumption is just one of them, but definitely the least studied one. While non-functional requirements may affect each other, mixing other criteria into our study about programmers' knowledge may blur the results.

V. Conclusion

According to our survey and interviews, programmers lack knowledge and awareness about software energy related issues. More than 80% of the programmers do not take energy consumption into account when developing software. However, more than 60% of the programmers consider software energy consumption to be an important factor when choosing a mobile development platform.

Since only 3% of the programmers received complaints about software energy consumption, it may suggest that users are unaware about software energy consumption. As suggested by Zhang [14], the creation of benchmarks and reporting mechanisms (similar to Energy Star) that inform users of the energy efficiency of software can significantly increase user awareness. Increased user awareness will, in turn, provide incentive for programmers to measurably enhance the energy efficiency of their software. As one Reddit respondent commented that the, "survey has at least made me consider [...] possible costs of doing things".

Pinto et. al. [11] have identified eight general strategies to reduce energy consumption through software modification: (1) keep IO to a minimum, (2) bulk operations, (3) avoid polling, (4) hardware coordination, (5) concurrent programming, (6) lazy initialization, (7) race to idle and (8) efficient data structure. These energy effective strategies should be part of programmers' education. In addition, tools can be created to identify unnecessary energy consuming behaviors in the software and to provide suggestions on how to reduce energy consumption during software development. Slides, videos, projects and assignments could be developed as part of an energy efficiency and sustainability curricular component for undergraduate courses.

PeerJ PrePrints

PeerJ PrePrints | https://dx.doi.org/10.7287/peerj.preprints.886v2 | CC-BY 4.0 Open Access | rec: 27 Jul 2015, publ: 27 Jul 2015
Digital Object Indentifier 10.1109/MS.2015.83 0740-7459/$26.00 2015 IEEE

This article has been accepted for publication in IEEE Software but has not yet been fully edited.
Some content may change prior to final publication.

VI. Authors

Candy Pang is a PhD student of Computing Science (CS) at the University of Alberta (UofA). Her interest is in enterprise application design, implementation, maintenance and education. After graduating with her MSc, Candy worked as an application architect in multiple Government of Alberta ministries for more than 13 years. She received both of her BSc and MSc from CS at UofA. Candy is a member of the ACM, ISP, and ITCP. Contact her at cspang@ualberta.ca.

Abram Hindle is an assistant professor of Computing Science at the University of Alberta. His research focuses on problems relating to mining software repositories, improving software engineering-oriented information retrieval with contextual information, and the impact of software maintenance on software energy consumption. Abram received a PhD in computer science from the University of Waterloo. Contact him at abram.hindle@ualberta.ca (`http://softwareprocess.ca`).

Bram Adams is an assistant professor at Polytechnique Montréal, where he heads the MCIS lab on Maintenance, Construction and Intelligence of Software (`http://mcis.polymtl.ca`). He obtained his PhD at Ghent University (Belgium). His research interests include software release engineering in general, and software integration, software build systems, software modularity and software maintenance in particular. He is one of the organizers of the International Workshop on Release Engineering (RELENG), see `http://releng.polymtl.ca`. Contact him at bram.adams@polymtl.ca.

Ahmed E. Hassan is the Canada Research Chair (CRC) in Software Analytics, and the NSERC/BlackBerry Software Engineering Chair at the School of Computing at Queen's University, Canada. His research interests include mining software repositories, empirical software engineering, load testing, and log mining. Hassan received a PhD in Computer Science from the University of Waterloo. He spearheaded the creation of the Mining Software Repositories (MSR) conference and its research community. Hassan also serves on the editorial boards of IEEE Transactions on Software Engineering, Springer Journal of Empirical Software Engineering, Springer Journal of Computing, and PeerJ Computer Science. Contact him at ahmed@cs.queensu.ca.

References

[1] Stack Overflow. http://stackoverflow.com, 2008.

[2] Control Objectives for Information and Related Technology (COBIT 5). http://www.isaca.org/cobit/, 2012.

[3] A. Banerjee, L. K. Chong, S. Chattopadhyay, and A. Roychoudhury. Detecting energy bugs and hotspots in mobile apps. In *Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering*, pages 588–598. ACM, 2014.

[4] B. Göetz, T. Peierls, J. Bloch, J. Bowbeer, D. Holmes, and D. Lea. *Java concurrency in practice*. Addison-Wesley, 2006.

[5] A. Hindle. Green Mining: A Methodology of Relating Software Change to Power Consumption. In *Proceedings of the 9th Working Conference on Mining Software Repositories*, 2012.

[6] M. E. Jed Scaramella. Solutions for the datacenter's thermal challenges. http://whitepapers.zdnet.com/abstract.aspx?docid=352318, January 2007. IDC white paper.

PeerJ PrePrints

PeerJ PrePrints | https://dx.doi.org/10.7287/peerj.preprints.886v2 | CC-BY 4.0 Open Access | rec: 27 Jul 2015, publ: 27 Jul 2015
Digital Object Indentifier 10.1109/MS.2015.83 0740-7459/$26.00 2015 IEEE

This article has been accepted for publication in IEEE Software but has not yet been fully edited.
Some content may change prior to final publication.

[7] H. Khalid, E. Shihab, M. Nagappan, and A. E. Hassan. What do mobile app users complain about? A study on free iOS apps. *Accepted to be published in IEEE Software*, 2014.

[8] P. Kurp. Green computing. *Communications of the ACM*, 51(10):11–13, 2008.

[9] D. Li, S. Hao, J. Gui, and W. Halfond. An Empirical Study of the Energy Consumption of Android Applications. In *Proceedings of the IEEE International Conference on Software Maintenance and Evolution, 2014*, pages 121–130. IEEE, 2014.

[10] C. Pang. Technical Summary: What do developers know about Software Energy consumption and power use? http://webdocs.cs.ualberta.ca/ hindle1/2014/green-programmers/, 2013.

[11] G. Pinto, F. Castor, and Y. D. Liu. Mining Questions About Software Energy Consumption. In *Proceedings of the 11th Working Conference on Mining Software Repositories*, 2014.

[12] Reddit. reddit: the front page of the internet. http://www.reddit.com, 2005.

[13] TIOBE. Tiobe index for november 2014. http://www.tiobe.com/index.php/tiobe_index, 2014.

[14] C. Zhang, A. Hindle, and D. M. Germán. The impact of user choice on energy consumption. *IEEE Software*, 31(3):69–75, 2014.

PeerJ PrePrints

PeerJ PrePrints | https://dx.doi.org/10.7287/peerj.preprints.886v2 | CC-BY 4.0 Open Access | rec: 27 Jul 2015, publ: 27 Jul 2015
Digital Object Indentifier 10.1109/MS.2015.83 0740-7459/$26.00 2015 IEEE