A Systematic Method for Evaluating the Generalizability of Mobile-Specific Research: Green Computing as a Case Study

Robin Nunkesser *

Posted Date: 6 January 2026

doi: 10.20944/preprints202601.0328.v1

Keywords: systematic mapping study; systematic literature review; software engineering; mobile software engineering; green computing; energy efficiency

 doi:10.20944/preprints202601.0328.v1

Article

A Systematic Method for Evaluating the Generalizability of Mobile-Specific Research: Green Computing as a Case Study

Robin Nunkesser

Hamm-Lippstadt University of Applied Sciences, Marker Allee 76–78, 59063 Hamm, Germany; robin.nunkesser@hshl.de

Abstract

Mobile Software Engineering has emerged as a distinct subfield, raising questions about the transferability of its research findings to general Software Engineering. This paper addresses the challenge of evaluating the generalizability of mobile-specific research, using Green Computing as a representative case. We propose a systematic method that combines a mapping study to identify potentially overlooked mobile-specific papers with a focused literature review to assess their broader relevance. Applying this approach, we find that several mobile-specific studies offer insights applicable beyond their original context, particularly in areas such as energy efficiency guidelines, measurement, and trade-offs. The results demonstrate that systematic identification and evaluation can reveal valuable contributions for the wider Software Engineering community. The proposed method provides a structured framework for future research to assess the generalizability of findings from specialized domains, fostering greater integration and knowledge transfer across Software Engineering disciplines.

Keywords: systematic mapping study; systematic literature review; software engineering; mobile software engineering; green computing; energy efficiency

1. Introduction

Recent studies, such as [1,2], have identified a temporary split-off of mobile-specific research, resulting in the emergence of Mobile Software Engineering as a distinct subfield. This separation has led to the development of specialized research communities and the potential for valuable mobile-specific findings to remain siloed, rather than informing the broader Software Engineering discipline. Notably, [3] highlights Energy Efficiency—an essential aspect of Green Computing—as a prime example where mobile-specific research may offer insights with wider applicability.

Green Computing, as defined by [4], encompasses research focused on reducing the energy consumption of computing systems, with implications spanning both hardware and software domains. While energy efficiency is a universal concern across all computing platforms, it is particularly critical in the context of mobile devices and applications due to inherent constraints such as limited battery capacity and the need for prolonged device autonomy. Green Computing research within Mobile Software Engineering has produced a wealth of studies addressing energy-efficient design, development, and deployment of mobile applications. It is therefore imperative to assess whether these mobile-specific findings can be generalized to inform energy efficiency practices in the broader Software Engineering landscape.

A systematic mapping conducted by [3] already identified at least 32 publications addressing Energy Efficiency within Mobile Software Engineering. However, the extent to which the findings from these mobile-specific studies can be generalized to the broader field of Software Engineering remains largely unexplored. This gap underscores the need for a structured approach to evaluate the transferability of mobile-specific research outcomes.

In response, this paper proposes a systematic method for assessing the generalizability of mobile-specific research, using Green Computing as a representative case study. Building upon the foundation established by

 doi:10.20944/preprints202601.0328.v1

prior work, we aim to (i) systematically identify mobile-specific research contributions in the area of energy efficiency, (ii) analyze their relevance and potential impact on general Software Engineering practices, and (iii) provide a methodological framework that can be applied to other specialized domains. Through this approach, we seek to bridge the divide between specialized and general research communities, ensuring that significant findings are leveraged to advance the field as a whole.

2. Materials and Methods

This paper proposes a systematic method for evaluating the generalizability of mobile-specific research findings. The method is designed to be applicable across various research domains. As a case study, we focus on Green Computing within the context of Mobile Software Engineering. To contextualize our approach, we first briefly revisit the emergence of Mobile Software Engineering as a distinct subfield.

2.1. Mobile Software Engineering

The emergence of Mobile Software Engineering as a distinct subfield has been driven by the unique challenges and requirements associated with mobile application development. Mobile devices, characterized by their portability, limited resources, and diverse operating environments, necessitate specialized approaches to software design, development, testing, and maintenance. As a result, researchers and practitioners have increasingly focused on addressing these mobile-specific concerns, leading to the establishment of dedicated conferences, journals, and research communities.

The recognition of Mobile Software Engineering as a separate discipline has prompted discussions regarding the transferability of its research findings to the broader Software Engineering community. In particular, there is a growing interest in understanding whether insights gained from mobile-specific studies can inform general software development practices, especially in areas such as energy efficiency, performance optimization, and user experience.

[2] describes the temporary split-off of Mobile Software Engineering as an opportunity:

> Systematically looking at mobile-specific results and checking whether they are generalizable offers opportunities for research.

According to [3] "energy efficiency, defect analysis, and maintenance seem to be very promising" areas for this (this is motivated by the systematic map shown in Figure 1).

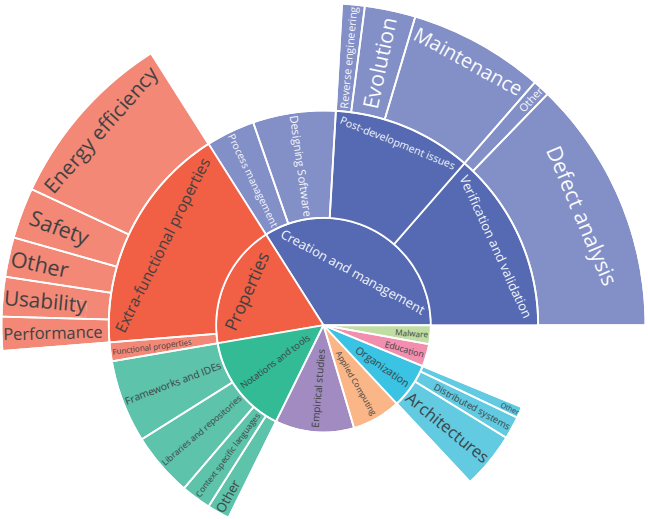

Figure 1. Systematic Map of Mobile Software Engineering (originally published in [3]).

The systematic map shows that Energy Efficiency as part of Green Computing is one of the areas where mobile-specific research has produced a considerable number of publications. Therefore, we

 doi:10.20944/preprints202601.0328.v1

will use Green Computing as a case study to demonstrate the proposed method. Other prominent candidates for case studies would be Defect Analysis or Maintenance.

The methodological approach to check whether mobile-specific results are generalizable is based on a combination of systematic review methods. Therefore, we will briefly introduce the systematic review methods used in this paper.

2.2. Systematic Review

Systematic research methods have become increasingly important in Software Engineering since the seminal work of [5]. Among these, the systematic literature review (SLR) is now a well-established approach for providing a comprehensive overview of existing literature within a specific domain. However, conducting an SLR is often time-consuming and resource-intensive.

To address these challenges, Bailey et al. [6] introduced the concept of a systematic mapping study (SMS) see also [7,8] as a precursor to an SLR. An SMS is a form of secondary study that offers a broad overview of primary studies in a given area. While less rigorous than an SLR, an SMS is valuable for mapping the landscape of existing research and identifying key topics and trends. SMSs are particularly useful for scoping out research areas and identifying gaps in the literature.

In this work, we propose an intertwined method that combines a systematic mapping study with a systematic literature review to evaluate the generalizability of mobile-specific research findings. The first phase concentrates on ideas of systematic mapping studies to identify research that may have received limited attention, potentially due to the emergence of distinct research communities following the split-off of Mobile Software Engineering. Rather than focusing solely on the most prominent publications, our approach aims to uncover studies that may have been overlooked by the broader Software Engineering community. The systematic map thus serves as a foundation for identifying relevant mobile-specific research.

The second phase involves conducting a systematic literature review, adapted to assess the generalizability of the findings from the identified mobile-specific studies. This review focuses on evaluating whether the insights and results from these studies can be applied to general Software Engineering contexts. By combining these two systematic approaches, we aim to provide a comprehensive method for assessing the transferability of mobile-specific research findings.

This approach not only facilitates the identification of potentially overlooked research but also enables a structured evaluation of its broader relevance. The integration of systematic mapping and literature review methods offers a robust framework for exploring the generalizability of specialized research domains, ultimately contributing to a more cohesive understanding of Software Engineering as a whole.

The overall process of the combined study is illustrated in Figure 2. The figure also shows how the steps of the systematic mapping study and the systematic literature review are intertwined in this combined approach. The review protocol is marked as part of SLRs, as it is optional for SMSs. Bailey et al. [6] mention that the review protocol will generally be much shorter for an SMS than for an SLR. Petersen et al. [8] do not mention the review protocol in their description of SMSs and therefore we decided to mark it as part of SLRs only. In our proposed method, the review protocol is also optional and typically rather short if used.

The data extraction step typically used in SLRs has to be adapted in this combined method. Data extraction typically operates on the assumption that all papers share the same context. In our case, we have to extract data from mobile-specific papers and analyze whether the results are generalizable to a broader context. Therefore, we have to adapt the data extraction step to focus on the generalizability of the results reported in the mobile-specific papers.

If the number of papers found in the systematic mapping is small enough, it may be possible to substitute the data extraction step with a full-text analysis of all mobile-specific papers found in the mapping study. In this case, the full-text analysis would directly assess the generalizability of the results reported in the mobile-specific papers.

 doi:10.20944/preprints202601.0328.v1

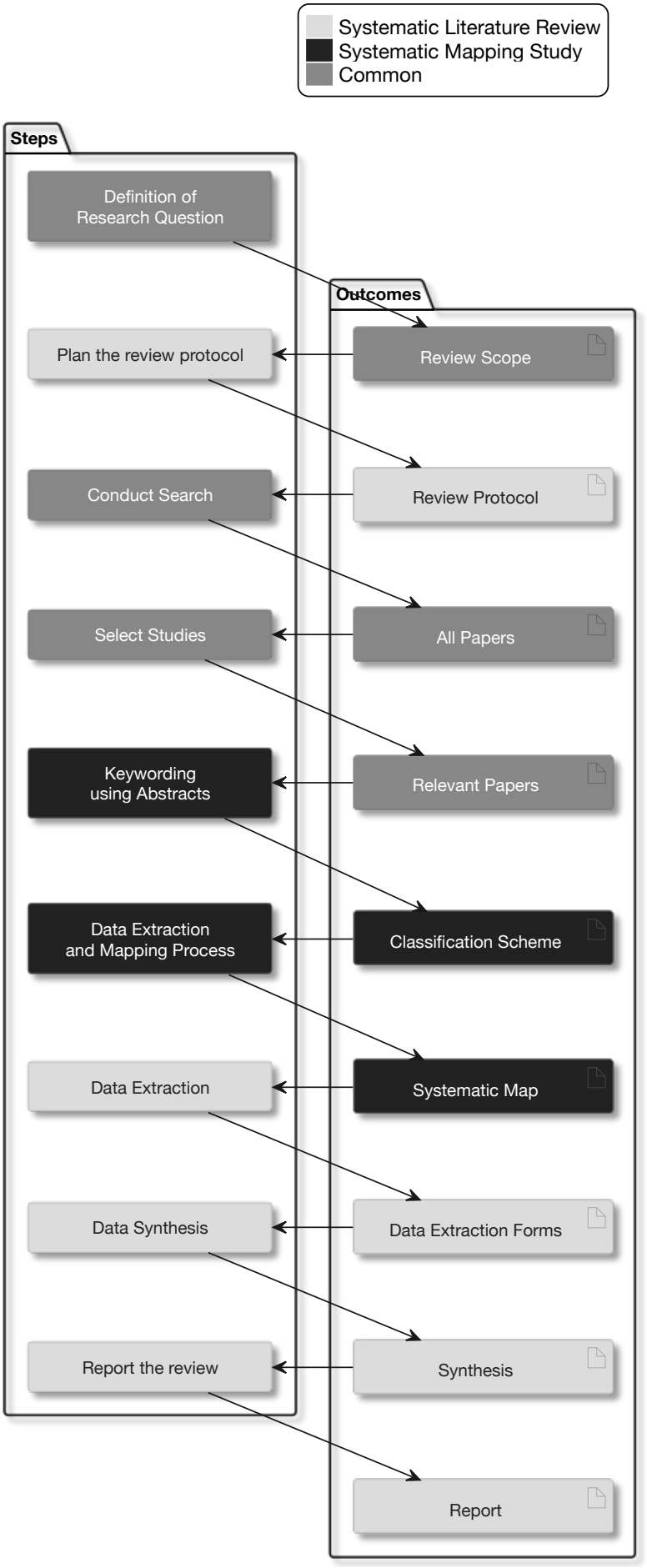

Figure 2. Steps and outcomes in a Combined Generalizability Study.

 doi:10.20944/preprints202601.0328.v1

2.3. Related Work

This work extends the findings of [3], building upon the delineation of Mobile Software Engineering as discussed in [1,2]. The methodological framework is informed by established guidelines for systematic studies in Software Engineering, notably [6,8,9].

While comprehensive literature reviews on Green Computing exist (see, e.g., [10–12]) and further background on Green Software Engineering can be found in [13], this paper introduces a targeted review methodology for evaluating the generalizability of mobile-specific research. Systematic Mapping Studies and Systematic Literature Reviews are now widely recognized in Software Engineering research, with related systematic reviews provided by [14–16].

To the best of our knowledge, this is the first study to integrate a systematic mapping study with a systematic literature review specifically to assess the generalizability of specific research findings to the broader Software Engineering domain.

3. Results

This section presents the findings from our intertwined systematic mapping study and literature review. As outlined in the Methods section, we focus on Green Computing as a case study to evaluate the generalizability of mobile-specific research findings. The results are structured according to the steps of the study. Some steps are only briefly described or omitted here, if they do not contribute significantly to the overall findings.

3.1. Research Questions

It is important to look carefully at what kind of papers we want to consider. We focus on mobile specific papers related to software engineering and green computing.

The research questions for the Mapping Study are:

RQ1 Does siloed mobile-specific research exist?

RQ2 Which mobile-specific papers seem to be generalizable?

- **RQ2.1** Which generalizable topics are covered by the papers?
- **RQ2.2** Which generalizable results are reported by the papers?

Note, that theses questions are applicable to other research areas as well. Also note, that the questions build upon each other. RQ1 is a prerequisite for RQ2 and may make RQ2 obsolete if the answer to RQ1 is negative. RQ2.1 and RQ2.2 are sub-questions of RQ2 and will be applied in the systematic literature review to analyze the generalizable papers found in the systematic mapping.

3.2. Conduct Search

As a first step, we will use the following searches and search terms:

ST1 In the Clarivate Web of Science database, search for:
Topic (TS) is "Green Computing" or "Energy Consumption" or "Energy Efficiency"
$\wedge$ TS is "Mobile" or "iOS" or "Android"
$\wedge$ Web of Science Category (WC) is "Computer Science Software Engineering"

ST2 In the Scopus database, search for:
Title, Abstract, or Keywords (TITLE-ABS-KEY) contain "Green Computing" or "Energy Consumption" or "Energy Efficiency"
$\wedge$ TITLE-ABS-KEY contains "iOS" or "Android"
$\wedge$ Subject Area (SUBJAREA) is "Computer Science" (COMP)

ST3 All mobile-specific Green Computing papers reported by Nunkesser [3]

Note, that the tags used in the search terms are specific to the respective databases. The Scopus Subject Area "COMP" is broader than the Web of Science Category "Computer Science Software

 doi:10.20944/preprints202601.0328.v1

Engineering". Additionally, the tag TITLE-ABS-KEY is much broader than the TS tag used in Web of Science. Therefore, we cannot include the rather unspecific search term "Mobile" in ST2 without generating an unmanageable number of papers.

Remember, that our motivation is to find papers that may have been overlooked by the general Software Engineering community. Therefore, the search is restricted to the Web of Science Category "Computer Science Software Engineering".

In ST1 and ST2 the search is further restricted to conference papers and journal articles in English.

Table 1 gives a summary of the number of papers found for each search.

Table 1. Number of papers found for the searches.

Search	Source	Papers
ST1	Clarivate Web of Science	1.538
ST2	Scopus	1.342
ST3	Nunkesser [3]	32

3.3. Select Studies

The next step is to screen the papers found in the search using inclusion and exclusion criteria. Due to the specific nature of the searches in the preceding step, we only need exclusion criteria. The exclusion criteria are:

EC1 Duplicates of papers returned by ST1, ST2, and ST3

EC2 Papers not published in Mobile Software Engineering specific journals or conference proceedings

EC3 Papers with less than four pages

Again, the motivation for the exclusion criteria is to find papers that may have been overlooked by the general Software Engineering community. All papers not published in Mobile Software Engineering specific journals or conference proceedings are excluded because they literally have already been considered by the general Software Engineering community. After applying the exclusion criteria, we will have a set of papers that will be used for the next step. Due to the list of journals and conference proceedings considered Mobile Software Engineering specific being rather short, EC2 is the most important exclusion criterion. The exclusion of papers based on EC3 is motivated by the fact that these papers are mostly demo papers or short papers that do not provide enough information to be considered for generalization. As EC2 results in a list of journals and conference proceedings, it is easy to check, that EC3 is not redundant and not to strict.

The number of papers after applying the exclusion criteria is 42. While this may seem a low number, it is a good result with regard to our target of strictly restricting the study to potentially overlooked papers.

Due to space constraints, a comprehensive account of the exclusion process cannot be provided here; however, selected examples are presented to illustrate the applied criteria. Journals like *Sustainable Computing: Informatics and Systems* were excluded as they are not specific to Mobile Software Engineering, despite their relevance to Green Computing. A journal like *IEEE Transactions on Mobile Computing*, although frequently appearing in the search results, was excluded because it focuses on Mobile Computing rather than specifically on Mobile Software Engineering. Conference proceedings from events such as *ICSE* and *FSE* were also excluded, as papers published in these venues are already part of the general Software Engineering discourse.

3.4. Keywording and Data Extraction

The keywording or classifying of the papers uses the titles and abstracts of the papers and all papers directly citing the considered papers. The keywords are used to map the papers to the research questions. We distinguish between the following categories:

 doi:10.20944/preprints202601.0328.v1

- Mobile specific papers (not generalizable)
- Potentially generalizable papers

3.5. Systematic Mapping

The data extracted from the papers will be used to answer some of the research questions. As a first step, we map the papers to one of three categories:

CAT1 Papers that where already cited in non mobile specific papers

CAT2 Papers that seem to be generalizable

CAT3 Papers that do not seem to be generalizable

Note, that this mapping is preliminary and subject to further validation through the systematic literature review. Due to the subjective nature of the classification, we only use CAT3 if there is a very clear indication that the paper is not generalizable.

Figure 3 shows the result of the mapping.

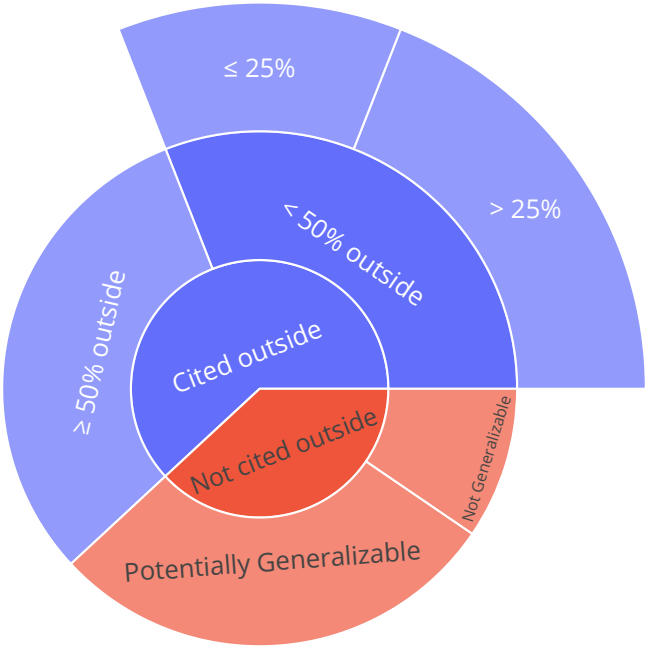

Figure 3. Systematic Map of Citation Area and Generalizability..

The figure is our answer to RQ1. It shows that there is some connection between mobile-specific and general research. However, there are still a considerable number of papers that have not yet been cited in non mobile specific papers. It also gives a first answer to RQ2 by showing that there are potentially generalizable papers. However, RQ2 can only be fully answered by conducting a systematic literature review of the identified papers and will be addressed together with RQ2.1 and RQ2.2 in the following.

We will concentrate on two of the identified groups in the following: potentially generalizable papers that were not yet cited in non mobile specific papers and papers that were already cited in mobile specific and non mobile specific papers (and are therefore generalizable) with a maximum of 25% non mobile specific paper citations (nonetheless potentially overlooked). Note, that some of the papers considered are fairly new and may not have been cited yet outside of mobile specific papers for this reason alone.

Table 2 shows the papers resulting from this selection. The papers are from the following conferences: DeMobile, ICOCOE, MOBILESoft, and SBESC. The number of citations is given in the last column.

 doi:10.20944/preprints202601.0328.v1

Table 2. Papers identified in the mapping study.

Paper	Citations
Moshnyaga [17]	2
Ramasamy et al. [18]	4
Banerjee and Roychoudhury [19]	43
Banerjee et al. [20]	21
Lee et al. [21]	5
Banerjee and Roychoudhury [22]	12
Hall et al. [23]	2
Corbalan et al. [24]	16
Song et al. [25]	4
Rua et al. [26]	5
Malavolta et al. [27]	11
Rammos et al. [28]	1
Ferreira et al. [29]	2
Jacques et al. [30]	5
Bogdan and Malavolta [31]	1
Guégain [32]	0
Kurtz et al. [33]	2

3.6. Data Extraction and Synthesis

Due to the low number of papers, we do not need to use data extraction, as all papers can be analyzed directly. The data extraction is therefore integrated into the data synthesis step.

The data synthesis for our case is based on a descriptive synthesis for papers covering the same topics. We do not expect a lot of overlap between the papers, so apart from the synthesis, we will also provide a summary of individual results.

3.7. Report

The results of the systematic literature review are presented in the following section. We distinguish between generalizable topics (answering RQ2.1) and generalizable individual results (answering RQ2.2).

3.7.1. Synthesis of Results

In the following, we will provide a synthesis of the results of the papers identified in the mapping study. The following topics are covered by at least two of the papers:

- Guidelines for Energy Efficiency
- Trade-off Between QoS and Energy Efficiency
- Energy Consumption Measurement
- Energy Consumption Differences of Frameworks

Guidelines for Energy Efficiency

Banerjee and Roychoudhury [19] describe four guidelines for energy efficiency in mobile applications: Sub-optimal Bindings, Nested Usages, Trade-offs of Quality-of-Service (QoS) vs Energy-efficiency, and Resource Leaks. Banerjee et al. [20] adds additional guidelines: Wakelock Bug, Tail-energy Hotspot, Vacuous Services, Expensive Services, Immortality Bug, and Loop-energy Hotspot. Hall et al. [23] focus on Wakelock bugs. While many of these guidelines result e.g. from sensor usage, they are nevertheless generalizable to other contexts.

These guidelines, while derived from mobile contexts, address common software engineering concerns such as resource management, efficient service usage, and prevention of energy leaks—issues that are equally relevant in desktop, embedded, and cloud-based systems. For example, the concept of resource leaks and wakelock bugs can be mapped to improper resource handling in any long-running application, not just mobile apps. Similarly, the trade-off between QoS and energy efficiency is a universal challenge in distributed and cloud systems, where balancing user experience with

 doi:10.20944/preprints202601.0328.v1

operational costs is critical. This topic is also addressed by other papers in the mapping study [18,21,27,29] and will be discussed in more detail below.

The systematic identification and codification of such guidelines in mobile research thus provide a valuable reference for broader software engineering practices, supporting the argument that these findings are indeed generalizable beyond their original domain.

Trade-off Between QoS and Energy Efficiency

The trade-off between QoS and energy efficiency is a recurring theme in mobile-specific research, but its implications extend well beyond the mobile domain. For instance, Ramasamy et al. [18] and Lee et al. [21] demonstrate how offloading computation to the cloud can optimize both energy consumption and service quality—a strategy increasingly relevant for edge computing, IoT, and cloud-native applications. Similarly, Malavolta et al. [27] shows that caching strategies, while improving responsiveness, can also impact energy usage, a consideration pertinent to web and enterprise systems. The work by Ferreira et al. [29] highlights the inherent tension between implementing robust security measures and maintaining energy efficiency, a dilemma faced in any resource-constrained environment, including embedded and cyber-physical systems.

These studies collectively underscore that managing the balance between QoS and energy efficiency is a universal software engineering challenge. The techniques and frameworks developed in the mobile context—such as adaptive offloading, dynamic resource allocation, and context-aware optimization—can inform best practices and architectural decisions in a wide range of computing environments. Thus, the insights from mobile-specific research on this trade-off are highly generalizable and can guide the design of energy-aware systems across diverse platforms.

Energy Consumption Measurement

Rua et al. [26], Rammos et al. [28] emphasize the importance of standardized and reproducible measurement methodologies for assessing energy consumption. They measure the difference in energy consumption between different apps for the same task. Large parts of the measurement methodologies are generalizable.

Guégain [32] propose a general framework for assessing the energy consumption of software, which is not limited to mobile applications. Their approach provides guidelines for setting up experiments, selecting appropriate metrics, and interpreting results, making it applicable to a wide range of software systems.

The methodologies and frameworks developed in these studies are highly generalizable. Accurate energy measurement is a foundational requirement for energy-aware software engineering in any context, including desktop, server, embedded, and cloud environments. The lessons learned from mobile-specific measurement challenges—such as accounting for hardware heterogeneity and usage patterns—can inform best practices for energy benchmarking and evaluation in broader domains. Thus, these contributions support the development of standardized energy measurement protocols that benefit the entire software engineering community.

Energy Consumption Differences of Frameworks

Corbalan et al. [24], Kurtz et al. [33] describe the energy consumption differences of development frameworks and approaches see e.g. for an overview of development approaches [34]. The general idea of analyzing energy consumption differences of development frameworks is generalizable.

These studies highlight that the choice of development frameworks and libraries can have a significant impact on the energy consumption of software, regardless of the target platform.

Jacques et al. [30] extend this line of inquiry to machine learning frameworks, demonstrating that the energy footprint of training and inference can vary widely depending on the chosen library and configuration. Their findings suggest that careful selection and tuning of frameworks is essential for energy-aware development, not only in mobile applications but also in server-side and cloud-based machine learning workloads.

 doi:10.20944/preprints202601.0328.v1

Overall, these results underscore the importance of considering energy consumption as a criterion when selecting development frameworks and libraries. The methodologies and comparative analyses developed in the mobile context can be readily adapted to other domains, supporting more sustainable software engineering practices across the board.

3.7.2. Individual Results

In the following, we will provide a summary of generalizable individual results of the papers identified in the mapping study. An overview of the topics is given by:

- Life Cycle Energy Assessment
- Debugging Field Failures
- Automated Refactoring for Energy Efficiency
- Software Maintenance
- Device-specific Energy Consumption
- Resource Sharing Across Heterogeneous Devices
- Impact of CSS Prefixes on Energy Consumption

Moshnyaga [17] describes an approach for the life cycle energy assessment of mobile devices. The novelty of his approach is the inclusion of software development energy cost. While the approach is specific to mobile devices, the general idea of including software development energy cost in life cycle assessments is generalizable.

Banerjee et al. [20] describe a method for debugging field failures in mobile applications. The method is based on the analysis of crash reports. While the method is specific to mobile applications, the general idea of improving the analysis of real-use crash reports is generalizable.

Banerjee and Roychoudhury [19] describe an approach for automated refactoring for energy efficiency. The generalizable approach consists of comparing so called design expressions and defect expressions.

Banerjee and Roychoudhury [22] describe the impact of energy efficiency on software maintenance. They present a hypothesis that energy efficiency is one of the most important reasons for lacking software acceptance in mobile apps. While the hypothesis is specific to mobile apps, the general idea that energy efficiency is important for software acceptance is generalizable. Banerjee and Roychoudhury [22] also describe the impact of device-specific energy consumption. The general idea of analyzing device-specific energy consumption is generalizable.

Song et al. [25] describe a generalizable approach for resource sharing across heterogeneous devices.

Bogdan and Malavolta [31] describe the impact of CSS prefixes on energy consumption. CSS prefixes are a general concept, so the general idea of analyzing the impact of CSS prefixes on energy consumption is generalizable.

4. Discussion

This paper presented a systematic method for evaluating the generalizability of mobile-specific research findings, using Green Computing as a case study. The approach combined a systematic mapping study to identify potentially overlooked mobile-specific papers with a focused literature review to assess their broader relevance. The systematic mapping study identified 17 papers that were either not cited in non-mobile-specific contexts or had a low citation rate outside of mobile-specific research. The subsequent systematic literature review analyzed these papers in detail, revealing several areas where mobile-specific research offers insights applicable to the broader Software Engineering community.

5. Conclusions

This paper presented a systematic method to evaluate the generalizability of mobile-specific research results to broader Software Engineering contexts, using Green Computing as a case study. Our approach combined a systematic mapping study to identify potentially overlooked mobile-specific

 doi:10.20944/preprints202601.0328.v1

papers with a systematic literature review to assess their generalizability. The systematic mapping study identified 17 potentially overlooked mobile-specific papers, which were then analyzed in detail through a systematic literature review.

Our findings suggest that several mobile-specific studies have broader implications, offering valuable insights for the general Software Engineering community. Key areas of generalizability include guidelines for energy efficiency, trade-offs between quality of service and energy efficiency, energy consumption measurement, and differences in energy consumption across frameworks. Additionally, individual results were found to be generalizable.

The proposed method demonstrates that systematic mapping studies can effectively identify potentially overlooked research, and systematic literature reviews can validate the generalizability of these findings. This approach can be applied to other research areas to bridge the gap between specialized and general research communities, fostering a more integrated and comprehensive understanding of the field.

Future work could extend this method to other subdomains of Software Engineering and explore the practical implications of integrating mobile-specific research into broader contexts. By doing so, we can ensure that valuable insights from specialized research are not missed and can contribute to the advancement of the entire field.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Nunkesser, R. The End of Mobile Software Engineering (As We Know It). In Proceedings of the 19th International Conference on Software Technologies - ICSOFT. INSTICC, SciTePress, 2024, pp. 131–136. https://doi.org/10.5220/0012780500003753.
2. Nunkesser, R. Mobile Software Engineering is Coming to an End (Like All Good Things Must). *SIGSOFT Softw. Eng. Notes* **2024**, *49*, 15–17. https://doi.org/10.1145/3696117.3696121.
3. Nunkesser, R. The Past, Present, and Future of Mobile Software Engineering. In Proceedings of the Software Technologies; Fill, H.G.; Domínguez Mayo, F.J.; van Sinderen, M.; Maciaszek, L.A., Eds., Cham, 2026. To appear.
4. Kurp, P. Green computing. *Commun. ACM* **2008**, *51*, 11–13. https://doi.org/10.1145/1400181.1400186.
5. Kitchenham, B.A.; Dyba, T.; Jorgensen, M. Evidence-Based Software Engineering. In Proceedings of the 26th International Conference on Software Engineering, USA, 2004; ICSE '04, p. 273–281.
6. Bailey, J.; Budgen, D.; Turner, M.; Kitchenham, B.; Brereton, P.; Linkman, S. Evidence relating to object-oriented software design: A survey. In Proceedings of the First International Symposium on Empirical Software Engineering and Measurement (ESEM 2007), 2007, pp. 482–484. https://doi.org/10.1109/ESEM.2007.58.
7. Petticrew, M.; Roberts, H. *Systematic reviews in the social sciences: A practical guide.*; Systematic reviews in the social sciences: A practical guide., Blackwell Publishing: Malden, 2006. https://doi.org/10.1002/9780470754887.
8. Petersen, K.; Feldt, R.; Mujtaba, S.; Mattsson, M. Systematic Mapping Studies in Software Engineering. 2008, pp. 1–10. https://doi.org/10.14236/ewic/EASE2008.8.
9. Kitchenham, B.A.; Charters, S. Guidelines for performing Systematic Literature Reviews in Software Engineering. Technical report, Keele University and University of Durham, 2007.
10. Georgiou, S.; Rizou, S.; Spinellis, D. Software Development Lifecycle for Energy Efficiency: Techniques and Tools. *ACM Comput. Surv.* **2019**, *52*. https://doi.org/10.1145/3337773.
11. Paul, S.G.; Saha, A.; Arefin, M.S.; Bhuiyan, T.; Biswas, A.A.; Reza, A.W.; Alotaibi, N.M.; Alyami, S.A.; Moni, M.A. A Comprehensive Review of Green Computing: Past, Present, and Future Research. *IEEE Access* **2023**, *11*, 87445–87494. https://doi.org/10.1109/ACCESS.2023.3304332.
12. Balanza-Martinez, J.; Lago, P.; Verdecchia, R. Tactics for Software Energy Efficiency: A Review. In Proceedings of the Advances and New Trends in Environmental Informatics 2023; Wohlgemuth, V.; Kranzlmüller, D.; Höb, M., Eds., Cham, 2024; pp. 115–140.
13. Calero, C.; Piattini, M. *Green in Software Engineering*; Springer Cham, 2015.

 doi:10.20944/preprints202601.0328.v1

14. Moreira, J.S.; Alves, E.L.G.; Andrade, W.L. A Systematic Mapping on Energy Efficiency Testing in Android Applications. In Proceedings of the XIX Brazilian Symposium on Software Quality, New York, NY, USA, 2021; SBQS '20. https://doi.org/10.1145/3439961.3439964.
15. Monteiro, E.; Cavalcante, H.; Barreto, R.; de Freitas, R. Analysis of Energy Consumption on Android Devices for Developers: A Systematic Mapping Study. *SBC Reviews on Computer Science* **2023**, *3*, 1–18. https://doi.org/10.5753/reviews.2023.2531.
16. Huber, S.; Lorey, T.; Felderer, M. Techniques for Improving the Energy Efficiency of Mobile Apps: A Taxonomy and Systematic Literature Review. In Proceedings of the 2023 49th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), 2023, pp. 286–292. https://doi.org/10.1109/SEAA60479.2023.00051.
17. Moshnyaga, V.G. Lifecycle energy assessment of mobile applications. In Proceedings of the 2013 International Workshop on Software Development Lifecycle for Mobile, New York, NY, USA, 2013; DeMobile 2013, pp. 17–23. event-place: Saint Petersburg, Russia, https://doi.org/10.1145/2501553.2501557.
18. Ramasamy, R.K.; Chua, F.F.; Haw, S.C. Web Service Composition Using Windows Workflow for Cloud-Based Mobile Application. In Proceedings of the Advanced Computer and Communication Engineering Technology; Sulaiman, H.A.; Othman, M.A.; Othman, M.F.I.; Rahim, Y.A.; Pee, N.C., Eds., Cham, 2015; pp. 975–985. https://doi.org/10.1007/978-3-319-07674-4_91.
19. Banerjee, A.; Roychoudhury, A. Automated re-factoring of Android apps to enhance energy-efficiency. In Proceedings of the International Conference on Mobile Software Engineering and Systems, New York, NY, USA, 2016; MOBILESoft '16, p. 139–150. https://doi.org/10.1145/2897073.2897086.
20. Banerjee, A.; Guo, H.F.; Roychoudhury, A. Debugging energy-efficiency related field failures in mobile apps. In Proceedings of the International Conference on Mobile Software Engineering and Systems, New York, NY, USA, 2016; MOBILESoft '16, p. 127–138. https://doi.org/10.1145/2897073.2897085.
21. Lee, W.; Sunwoo, D.; Gerstlauer, A.; John, L.K. Cloud-Guided QoS and Energy Management for Mobile Interactive Web Applications. In Proceedings of the 2017 IEEE/ACM 4th International Conference on Mobile Software Engineering and Systems (MOBILESoft), 2017, pp. 25–29. https://doi.org/10.1109/MOBILESoft.2017.4.
22. Banerjee, A.; Roychoudhury, A. Future of Mobile Software for Smartphones and Drones: Energy and Performance. In Proceedings of the 2017 IEEE/ACM 4th International Conference on Mobile Software Engineering and Systems (MOBILESoft), 2017, pp. 1–12. https://doi.org/10.1109/MOBILESoft.2017.1.
23. Hall, S.; Nataraj, S.; Kim, D.K. Detecting no-sleep energy bugs using reference counted variables. In Proceedings of the 5th International Conference on Mobile Software Engineering and Systems, New York, NY, USA, 2018; MOBILESoft '18, p. 161–165. https://doi.org/10.1145/3197231.3197257.
24. Corbalan, L.; Fernandez, J.; Cuitiño, A.; Delia, L.; Cáseres, G.; Thomas, P.; Pesado, P. Development frameworks for mobile devices: a comparative study about energy consumption. In Proceedings of the 5th International Conference on Mobile Software Engineering and Systems, New York, NY, USA, 2018; MOBILESoft '18, p. 191–201. https://doi.org/10.1145/3197231.3197242.
25. Song, Z.; Chadha, S.; Byalik, A.; Tilevich, E. Programming support for sharing resources across heterogeneous mobile devices. In Proceedings of the 5th International Conference on Mobile Software Engineering and Systems, New York, NY, USA, 2018; MOBILESoft '18, p. 105–116. https://doi.org/10.1145/3197231.3197250.
26. Rua, R.; Fraga, T.; Couto, M.; Saraiva, J.a. Greenspecting Android virtual keyboards. In Proceedings of the IEEE/ACM 7th International Conference on Mobile Software Engineering and Systems, New York, NY, USA, 2020; MOBILESoft '20, p. 98–108. https://doi.org/10.1145/3387905.3388600.
27. Malavolta, I.; Chinnappan, K.; Jasmontas, L.; Gupta, S.; Soltany, K.A.K. Evaluating the impact of caching on the energy consumption and performance of progressive web apps. In Proceedings of the IEEE/ACM 7th International Conference on Mobile Software Engineering and Systems, New York, NY, USA, 2020; MOBILESoft '20, p. 109–119. https://doi.org/10.1145/3387905.3388593.
28. Rammos, S.; Mundra, M.; Xu, G.; Tong, C.; Ziolkowski, W.; Malavolta, I. The Impact of Instant Messaging on the Energy Consumption of Android Devices. In Proceedings of the 2021 IEEE/ACM 8th International Conference on Mobile Software Engineering and Systems (MobileSoft), 2021, pp. 1–11. https://doi.org/10.1109/MobileSoft52590.2021.00007.
29. Ferreira, J.; Santos, B.; Oliveira, W.; Antunes, N.; Cabral, B.; Fernandes, J.P. On Security and Energy Efficiency in Android Smartphones. In Proceedings of the 2023 IEEE/ACM 10th International Conference on Mobile Software Engineering and Systems (MOBILESoft), 2023, pp. 87–95. https://doi.org/10.1109/MOBILSoft59058.2023.00018.

 doi:10.20944/preprints202601.0328.v1

30. Jacques, V.M.F.; Alizadeh, N.; Castor, F. A Study on the Battery Usage of Deep Learning Frameworks on iOS Devices. In Proceedings of the IEEE/ACM 11th International Conference on Mobile Software Engineering and Systems, New York, NY, USA, 2024; MOBILESoft '24, p. 1–11. https://doi.org/10.1145/3647632.3647990.
31. Bogdan, A.; Malavolta, I. An Empirical Study on the Impact of CSS Prefixes on the Energy Consumption and Performance of Mobile Web Apps. In Proceedings of the IEEE/ACM 11th International Conference on Mobile Software Engineering and Systems, New York, NY, USA, 2024; MOBILESoft '24, p. 12–21. https://doi.org/10.1145/3647632.3647989.
32. Guégain, E. Assessing the environmental impact of mobile applications: a measure framework toward DevGreenOps. In Proceedings of the IEEE/ACM 11th International Conference on Mobile Software Engineering and Systems, New York, NY, USA, 2024; MOBILESoft '24, p. 88–91. https://doi.org/10.1145/3647632.3651391.
33. Kurtz, K.; Noguez, M.; Zanini, F.; Ferreira, P.R.; Brisolara, L. Comparing Performance and Energy Consumption of Android Applications: Native Versus Web Approaches. In Proceedings of the 2017 VII Brazilian Symposium on Computing Systems Engineering (SBESC), 2017, pp. 147–154. https://doi.org/10.1109/SBESC.2017.26.
34. Nunkesser, R. Beyond web/native/hybrid: a new taxonomy for mobile app development. In Proceedings of the 5th International Conference on Mobile Software Engineering and Systems, New York, NY, USA, 2018; MOBILESoft '18, p. 214–218. https://doi.org/10.1145/3197231.3197260.