HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY
SCHOOL OF APPLIED MATHEMATICS AND INFORMATICS

Technical Writing and Presentation

REPORT

Topic: Solution for greener software engineering

Instructor:	Assoc.Prof.Ph.D.Nguyen Dinh Han
Course ID:	MI2030
Class ID:	142315
Group:	1
Student:	Nguyen Duc Thien
Student ID:	20173589

HA NOI – 2023

Contents

Introdution

The field of software engineering plays a vital role in the development and maintenance of software systems. However, traditional software engineering methods have often overlooked the concept of sustainability. This has led to inefficient efforts or a complete omission of sustainability considerations in software engineering practices. To address this gap, several research papers have explored the aspects of sustainability in software engineering and proposed strategies for its integration.

This final report aims to synthesize the findings from three relevant papers: ”What does Sustainability mean in and for Software Engineering?”, ”Power Consumption Estimation Models for Processors, Virtual Machines, and Servers” and ”A Comprehensive Approach to DRAM Power Management”. These papers highlight the lack of a common understanding of sustainability in software engineering and propose approaches to incorporate sustainability principles throughout the software development lifecycle.

By examining the insights from these papers, this report seeks to provide software engineers with practical strategies and recommendations for integrating sustainability into their practices. The report will focus on the four key aspects of sustainability in software engineering: development process, maintenance process, system production, and system usage. Through the exploration of case studies and real-world examples, software engineers will gain a roadmap for embracing sustainability and contributing to a more sustainable future through their work.

Chapter 1

What does Sustainability mean in and for Software Engineering?

1.1 The contributions

The main contribution is the description of the aspects of sustainability in software engineering. It provides a framework for understanding and addressing sustainability in the context of software development and maintenance. By identifying the four aspects (development process, maintenance process, system production, and system usage), the abstract offers a comprehensive view of sustainability considerations throughout the software lifecycle.

Additionally, the abstract offers exemplary actions that can be taken to improve sustainability in each aspect. These actions serve as practical guidelines for software engineers to integrate sustainability practices into their development processes and software systems. The examples range from knowledge management and quality assessment to energy-aware software design and business process optimization.

By presenting these aspects and actions, the abstract contributes to bridging the gap between sustainability and software engineering, providing a starting point for further research and development of sustainable software practices. It highlights the need for a common understanding of sustainability in software engineering and suggests directions for future work, including case studies and the development of an encompassing approach supported by an assessment model.

1.2 The Personal judgements or comments

The provided paper offers a detailed explanation of various methods for measuring or estimating power consumption in computing systems. It provides a comprehensive overview of direct instrumentation, indirect instrumentation, and estimation models based on hardware-provided or OS-provided metrics. This comprehensive coverage allows readers to understand the different approaches available and their respective advantages and limitations.

The paper effectively highlights the importance of accurately measuring or estimating power consumption in computing systems, as it directly impacts energy efficiency and resource management. By discussing real-world examples and research studies, the paper provides practical insights into the implementation and applicability of these approaches.

One strength of the paper is its inclusion of technical details, such as the use of shunt resistors, amplifiers, and specific performance monitoring counters, which adds depth and credibility to the explanations. Additionally, the mention of different benchmarks and their role in training and testing power estimation models further enhances the practical relevance of the discussion.

However, one potential limitation of the paper is that it focuses more on the technical aspects of power measurement and estimation, and less on the broader implications and potential applications of these approaches. It would be valuable to explore how accurate power measurement or estimation can inform energy-efficient system design, resource allocation, and workload scheduling strategies.

In conclusion, the paper provides a comprehensive and informative overview of power measurement and estimation approaches in computing systems. It effectively presents technical details and real-world examples, making it a valuable resource for researchers and practitioners working in the field of energy-efficient computing.

1.3 The relevance of this paper to my group suvery

The paper identifies four aspects of sustainability in software engineering, namely the development process, maintenance process, system production, and system usage. These aspects align with our survey's focus on greener software engineering by providing a structure to explore and assess practices and solutions in each of these areas.

The exemplary actions provided in the paper can serve as valuable references for our survey participants. These actions suggest practical steps that can be taken to improve sustainability in software engineering, such as using common artifact models, implementing knowledge management practices, optimizing resource usage, and analyzing the environmental impact of software services. Including questions related to these actions in our survey can help gauge the adoption and effectiveness of these practices in promoting greener software engineering.

Moreover, the paper's emphasis on the need for a common understanding of sustainability in software engineering and the proposed future directions align with our survey's exploration of solutions. You can incorporate questions that assess the awareness, challenges, and potential areas for improvement in implementing sustainable practices in software engineering based on the insights provided in the paper.

Overall, the paper's insights, framework, and suggested actions make it highly relevant to our survey on solutions for greener software engineering. It provides a foundation for understanding sustainability aspects and offers practical guidance that can inform the development of sustainable practices in software engineering.

Chapter 2

Power Consumption Estimation Models for Processors, Virtual Machines, and Servers

2.1 The contributions

The main contribution of the survey is to provide an overview of different approaches for measuring or estimating the power consumption of computing systems. It discusses the advantages and disadvantages of direct instrumentation, indirect instrumentation, and estimation models based on hardware-provided or OS-provided metrics.

The text also highlights several research studies that have employed these approaches, describing their methodologies and findings. It mentions studies that use direct measurements with shunt resistors and amplifiers, custom hardware for measuring specific subsystems, and PMC-based models for estimating CPU power consumption.

By providing this comprehensive overview and discussing the merits and limitations of different approaches, the text contributes to the understanding of how power consumption can be measured or estimated in computing systems. It also emphasizes the importance of considering the system architecture and workload characteristics when developing power estimation models.

Overall, the text serves as a valuable resource for researchers and practitioners interested in power measurement and estimation in computing systems, providing insights into existing approaches and their applicability in different scenarios.

2.2 The Personal judgements or comments

The article emphasizes the potential of Machine Learning (ML) methods in improving climate services and addressing climate change challenges. It highlights the benefits of ML algorithms in climate modeling, hazard assessment, and impact and adaptation assessment. The use of ML techniques, such as deep learning and random forest models, is discussed in relation to regional climate information, local-scale weather estimation, and post-processing of climate data.

The paper acknowledges the increasing availability of large environmental datasets and computational capacities, which provide opportunities for data-driven approaches and the application of ML in environmental research. It also mentions specific applications of ML in climate services, such as flood and wind damage modeling and health impact assessment.

Overall, the content of the paper suggests that ML has the potential to enhance understanding, prediction, and decision-making in the field of climate science. It highlights the advantages of ML in handling complex climate data and improving the accuracy of climate models and assessments.

2.3 The relevance of this paper to my group survey

The paper discusses various techniques and approaches for measuring and estimating power consumption in computing systems. This knowledge is essential for software engineers aiming to develop greener software solutions. By understanding the power consumption of different components and subsystems, software engineers can make informed decisions and design energy-efficient software.

The paper provides insights into the power consumption characteristics of CPUs and other hardware components. This information can guide software engineers in designing energy-efficient algorithms, optimizing resource utilization, and minimizing power-hungry operations. By incorporating power measurement and estimation techniques into the software development process, engineers can identify power-intensive areas and devise strategies to reduce energy consumption.

The paper discusses the use of performance monitoring counters and metrics to estimate power consumption. This information can be leveraged by software engineers to make informed decisions about resource allocation. By considering the power consumption profiles of different components, software engineers can distribute computational tasks and allocate resources in a way that minimizes energy usage.

In summary, the paper on power consumption measurement and estimation in computing systems can provide valuable insights and techniques that align with the goal of developing greener software engineering solutions. It can guide software engineers in making energy-efficient design choices, optimizing resource allocation, and selecting appropriate benchmarks for evaluating energy efficiency.

Chapter 3

A Comprehensive Approach to DRAM Power Management

3.1 The contributions

- Power-Down Mechanism: The paper introduces a power-down mechanism for DRAM that puts idle memory devices into low-power mode. By leveraging this mechanism, the proposed policy improves DRAM energy efficiency by up to 43.4% for different benchmark suites. The policy determines when to transition into low-power mode based on specific conditions, such as the rank's idle status, the absence of imminent commands, and the counter value.
- Power-Aware Memory Scheduler: The paper augments the Adaptive History-Based (AHB) scheduler to include power savings as a criterion. By grouping commands targeting the same rank closer together, the scheduler reduces the number of rank power-down operations, leading to improved power efficiency. The augmented scheduler probabilistically combines three finite state machines: performance-based, power-based, and balance-based, achieving optimization for both performance and power goals.
- Adaptive Memory Throttling: The paper presents an adaptive throttling approach that modulates DRAM power consumption. The approach involves blocking commands within the memory controller for a fixed throttling delay, allowing ranks to remain in low-power mode for longer durations. An accurate delay estimation model is implemented in the

memory controller to determine the throttling delay based on a target power threshold. The proposed delay estimation model shows performance within a few percentage points of an ideal oracular model.

Overall, the paper's main contributions lie in introducing effective mechanisms and policies for managing DRAM power consumption, improving energy efficiency, supporting power budget control, and minimizing performance degradation. These contributions are evaluated and demonstrated through simulations using an IBM Power5+ processor and a DDR2-533 SDRAM.

3.2 The Personal judgements or comments

Power-Down Mechanism: The paper introduces a power-down mechanism for DRAM that puts idle memory devices into low-power mode. By leveraging this mechanism, the proposed policy improves DRAM energy efficiency by up to 43.4% for different benchmark suites. The policy determines when to transition into low-power mode based on specific conditions, such as the rank's idle status, the absence of imminent commands, and the counter value.

Power-Aware Memory Scheduler: The paper augments the Adaptive History-Based (AHB) scheduler to include power savings as a criterion. By grouping commands targeting the same rank closer together, the scheduler reduces the number of rank power-down operations, leading to improved power efficiency. The augmented scheduler probabilistically combines three finite state machines: performance-based, power-based, and balance-based, achieving optimization for both performance and power goals.

Adaptive Memory Throttling: The paper presents an adaptive throttling approach that modulates DRAM power consumption. The approach involves blocking commands within the memory controller for a fixed throttling delay, allowing ranks to remain in low-power mode for longer durations. An accurate delay estimation model is implemented in the memory controller to determine the throttling delay based on a target power threshold. The proposed delay estimation model shows performance within a few percentage points of an ideal oracular model.

Overall, the paper's main contributions lie in introducing effective mechanisms and policies for managing DRAM power consumption, improving energy efficiency, supporting power budget control, and minimizing performance degradation. These contributions are evaluated and demonstrated through simulations using an IBM Power5+ processor and a DDR2-533 SDRAM.

3.3 The relevance of this paper to my group survey

The paper's findings and techniques may indirectly contribute to the broader goal of greener software engineering by improving overall system energy efficiency. By reducing power consumption in the memory subsystem, it can help optimize the energy usage of server systems. This, in turn, can have positive implications for the energy footprint of software systems running on such hardware.

Conclusion

In conclusion, this final report has explored the concept of sustainability in software engineering by analyzing and synthesizing the insights from three prominent research papers. The aim was to establish a comprehensive understanding of sustainability and its implications for software development practices.

Through this study, I have identified four key aspects of sustainability in software engineering: development process, maintenance process, system production, and system usage. Each aspect presents unique opportunities for incorporating sustainable practices and mitigating environmental impact while maintaining economic balance.

By examining various actions and strategies proposed in the research papers, I have gained valuable insights into how software engineers can actively contribute to sustainability. These actions include process improvements, resource optimization, knowledge management, and the analysis of software systems and business processes. Emphasizing the system usage aspect has emerged as a potential area for significant improvement and impact.

References

[1] Birgit Penzenstadler. ”What does Sustainability mean in and for Software Engineering?”. Chalmers University of Technology, 2013

[2] Alexander Schill, Waltenegus Dargie, Christoph Mobius. ”Power Consumption Estimation Models for Processors, Virtual Machines, and Servers”. IEEE, 2014

[3] Ibrahim Hur, Calvin Lin. ”A Comprehensive Approach to DRAM Power Management”. IEEE, 2008.