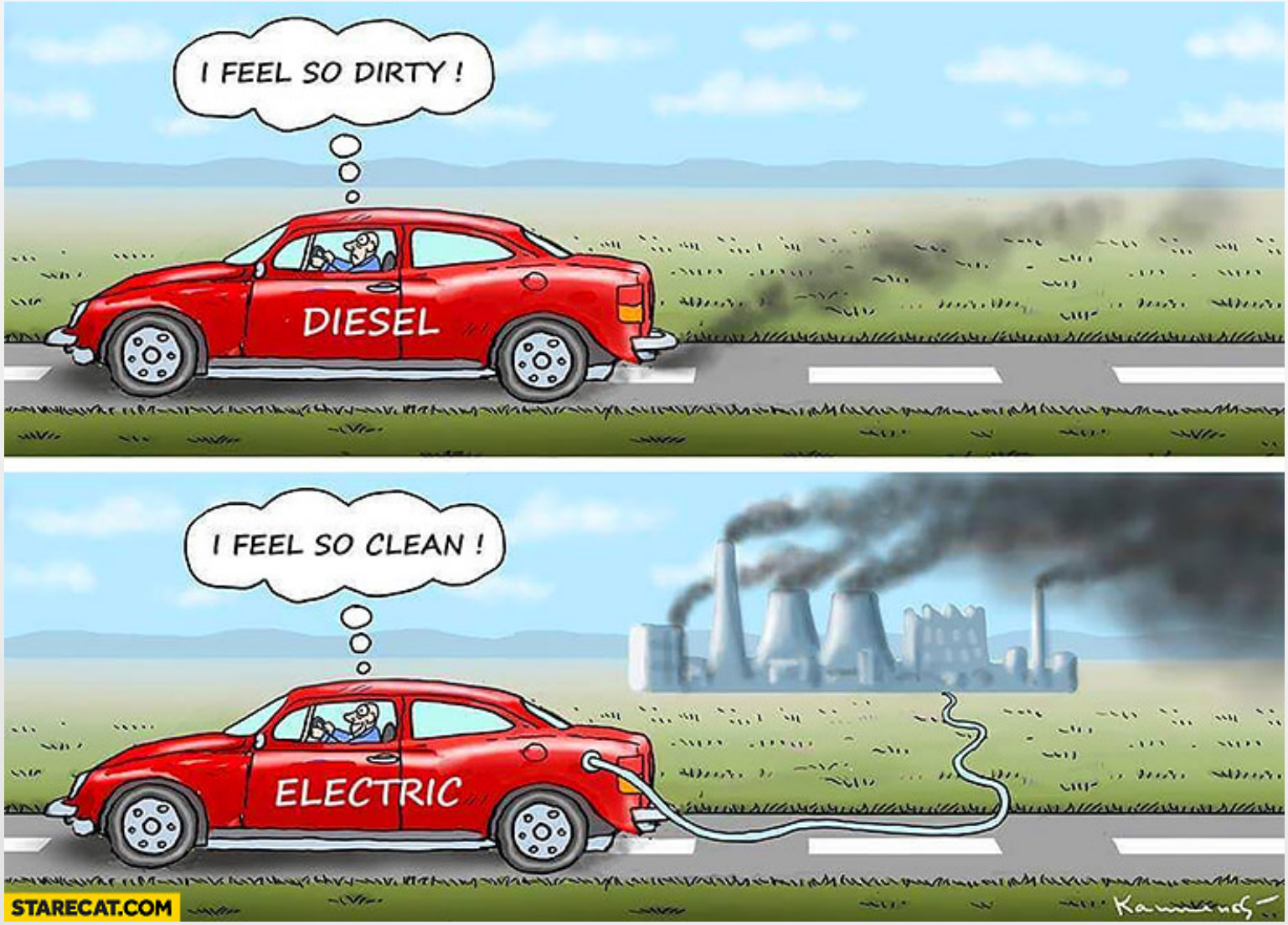
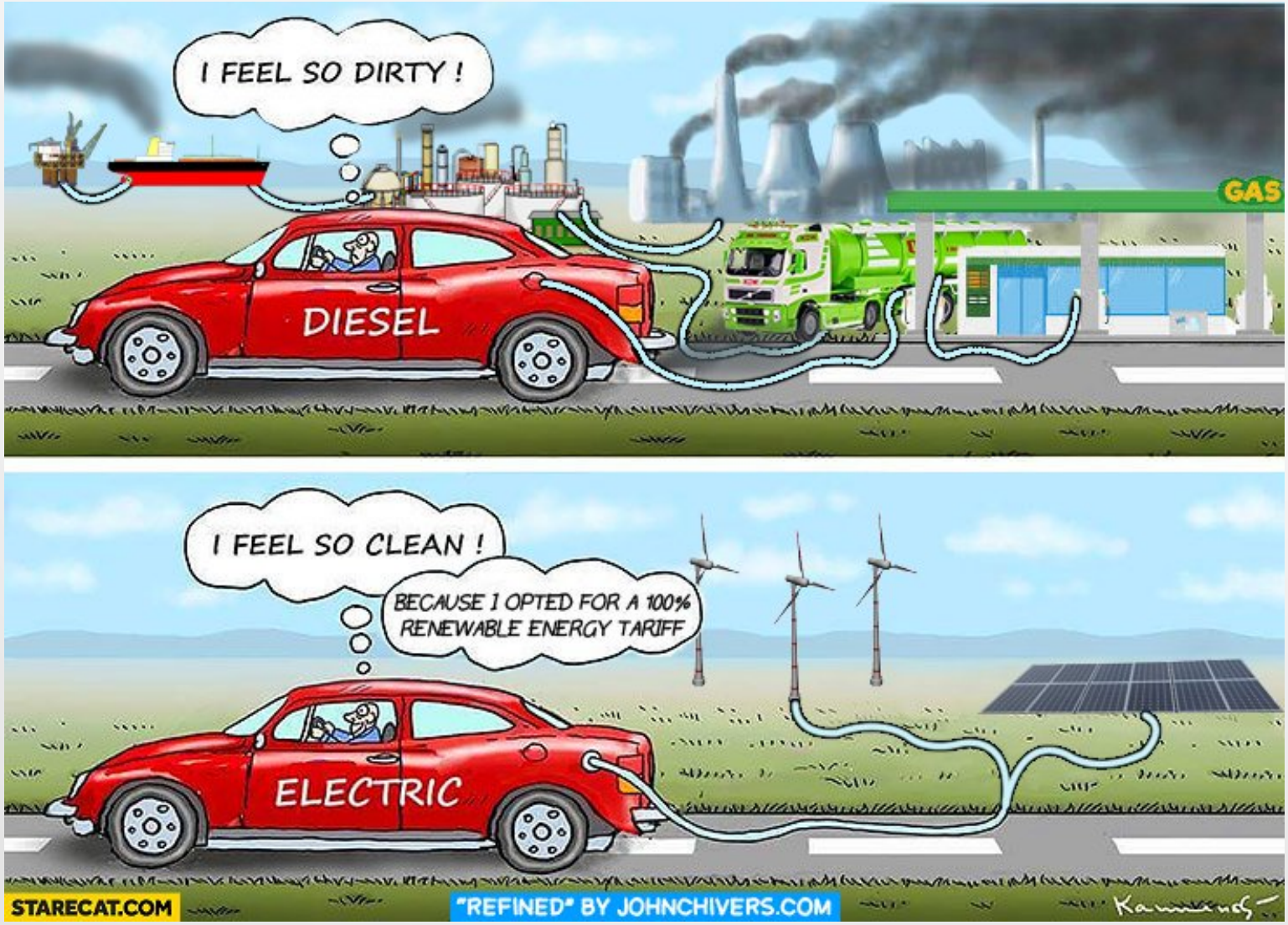
Co-funded by the
Erasmus+ Programme
of the European Union

Department of
Computers and Informatics
FEEI TU of Košice

Introduction to Green Software Engineering

Csaba Szabó
Technical university of Košice, Slovakia


I FEEL SO DIRTY !
DIESEL
I FEEL SO CLEAN !
ELECTRIC
STARECAT.COM


I FEEL SO DIRTY !
GAS
DIESEL
I FEEL SO CLEAN !
BECAUSE I OPTED FOR A 100% RENEWABLE ENERGY TARIFF
ELECTRIC
STARECAT.COM
"REFINED" BY JOHNCHIVERS.COM

Green Software Engineering

Green software engineering is a branch of software engineering focusing on energy aspects of software. Please note that software plays here the role of the process, which one's energy consumption can be expressed through the energy consumption of all hardware parts that are used in any way by the examined process. Therefore, when evaluating software greenness, we always examine the usage load on hardware parts during software execution time.

The role of hardware

- Display
- Networking (Wi-Fi, Bluetooth), radio
- Processor
- Memory
- Disks
- Battery
- Sensors

The role of software

- Operating system (difference between Windows, Linux, macOS, Android, iOS)
- Working software
- Computer games
- Application systems
- Databases

The role of the user

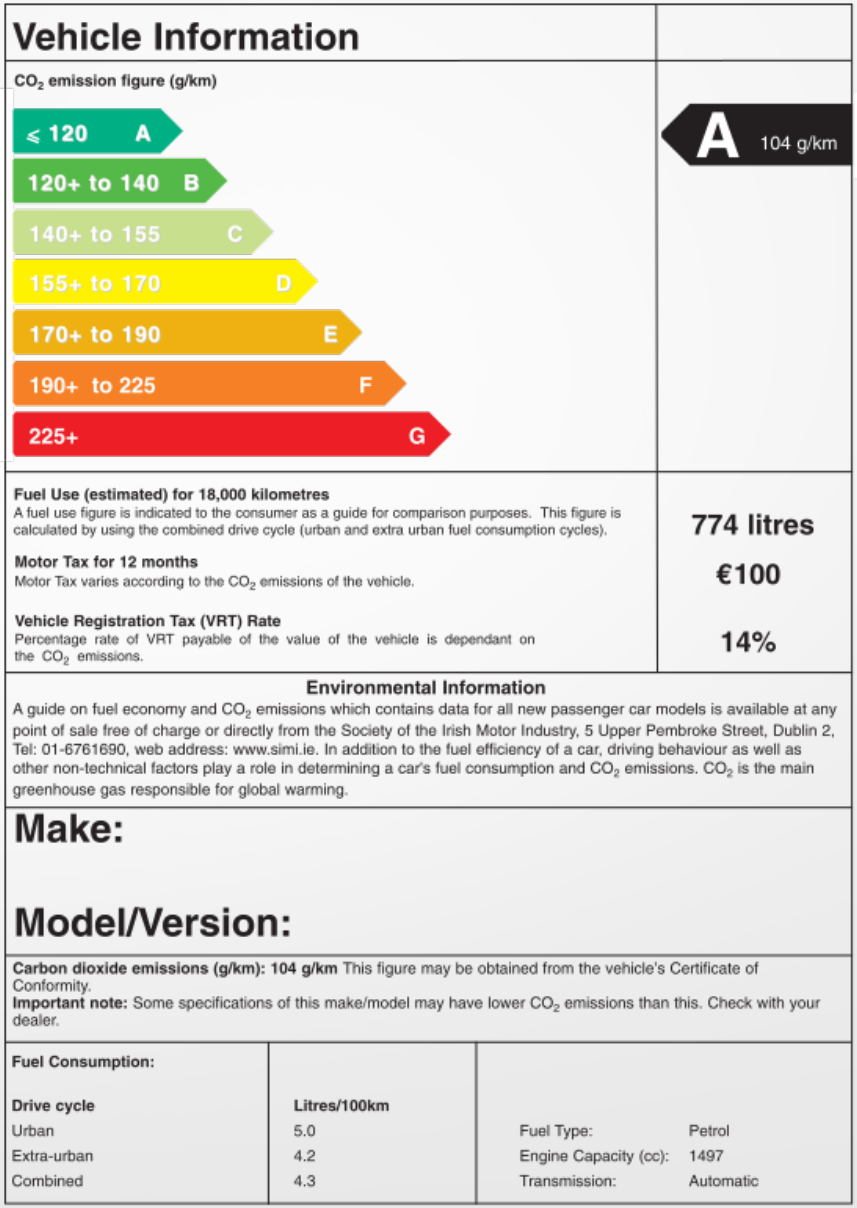
- The user "drives" the software
- Needs individual training (unlike HW/SW)
- Does (s)he receive it? Where?
- Repairing bad configuration is often done by buying a new device…

Green software, green IT

- Goals:
  - Save energy by more efficient hardware
  - Save energy by optimised/custom software
  - Save energy by location of hardware
- To make it really green:
  - Develop new working hardware
  - Develop energy efficient working software
  - Teach users to save energy when using the software
  - Make sure the used energy is also green

SW Energy Label?

HW/SW system energy label?



Vehicle Information

CO_2 emission figure (g/km)

- ≤ 120 A
- 120+ to 140 B
- 140+ to 155 C
- 155+ to 170 D
- 170+ to 190 E
- 190+ to 225 F
- 225+ G

A 104 g/km

Fuel Use (estimated) for 18,000 kilometres — 774 litres
A fuel use figure is indicated to the consumer as a guide for comparison purposes. This figure is calculated by using the combined drive cycle (urban and extra urban fuel consumption cycles).

Motor Tax for 12 months — €100
Motor Tax varies according to the CO_2 emissions of the vehicle.

Vehicle Registration Tax (VRT) Rate — 14%
Percentage rate of VRT payable of the value of the vehicle is dependant on the CO_2 emissions.

Environmental Information

A guide on fuel economy and CO_2 emissions which contains data for all new passenger car models is available at any point of sale free of charge or directly from the Society of the Irish Motor Industry, 5 Upper Pembroke Street, Dublin 2, Tel: 01-6761690, web address: www.simi.ie. In addition to the fuel efficiency of a car, driving behaviour as well as other non-technical factors play a role in determining a car's fuel consumption and CO_2 emissions. CO_2 is the main greenhouse gas responsible for global warming.

Make:

Model/Version:

Carbon dioxide emissions (g/km): 104 g/km This figure may be obtained from the vehicle's Certificate of Conformity.
Important note: Some specifications of this make/model may have lower CO_2 emissions than this. Check with your dealer.

Fuel Consumption:

Drive cycle	Litres/100km
Urban	5.0
Extra-urban	4.2
Combined	4.3

Fuel Type:	Petrol
Engine Capacity (cc):	1497
Transmission:	Automatic

Image from Wikipedia

Measuring energy consumption

Incl. improvements

- System level
- Application level
- Component level
- Code level

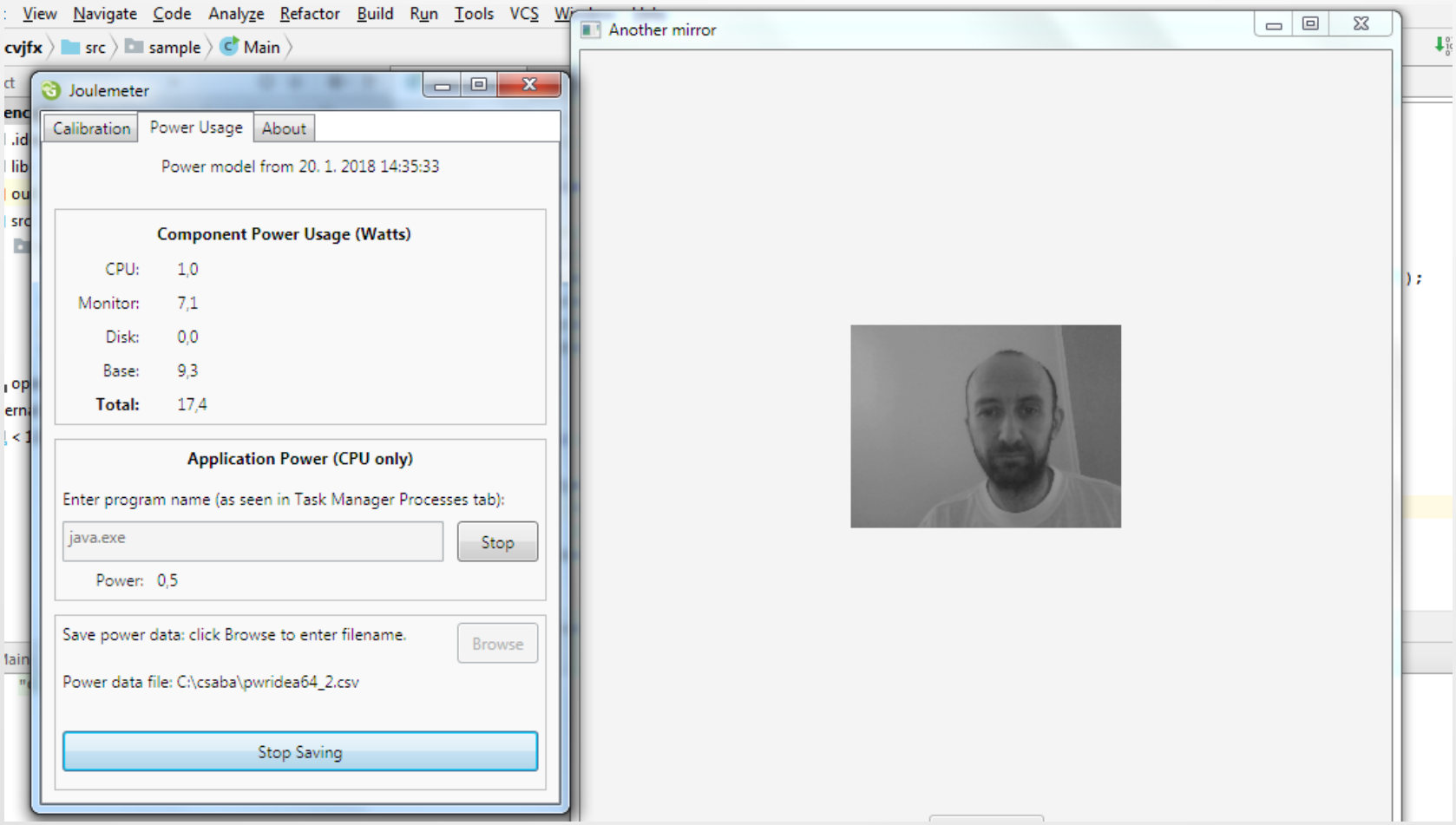
- Process level

More details can be found in Intellectual output 1 of the project 2017-1-SK01-KA203-035402: Focusing Education on Composability, Comprehensibility and Correctness of Working Software

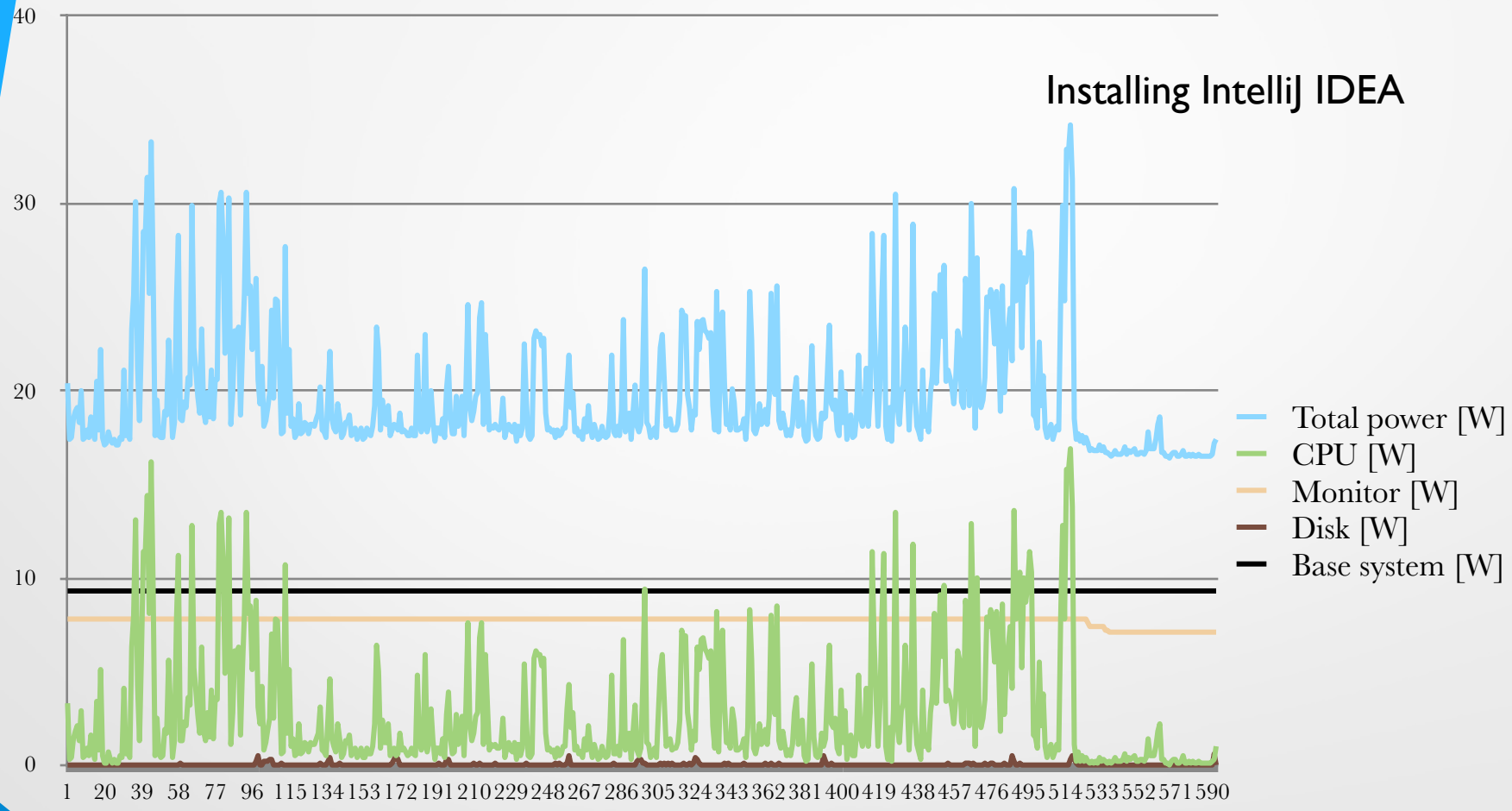
System level measurement

- SW-to-SW/HW solutions (servers, IoT)
- Uptime/availability prediction
- Providing a different evaluation perspective

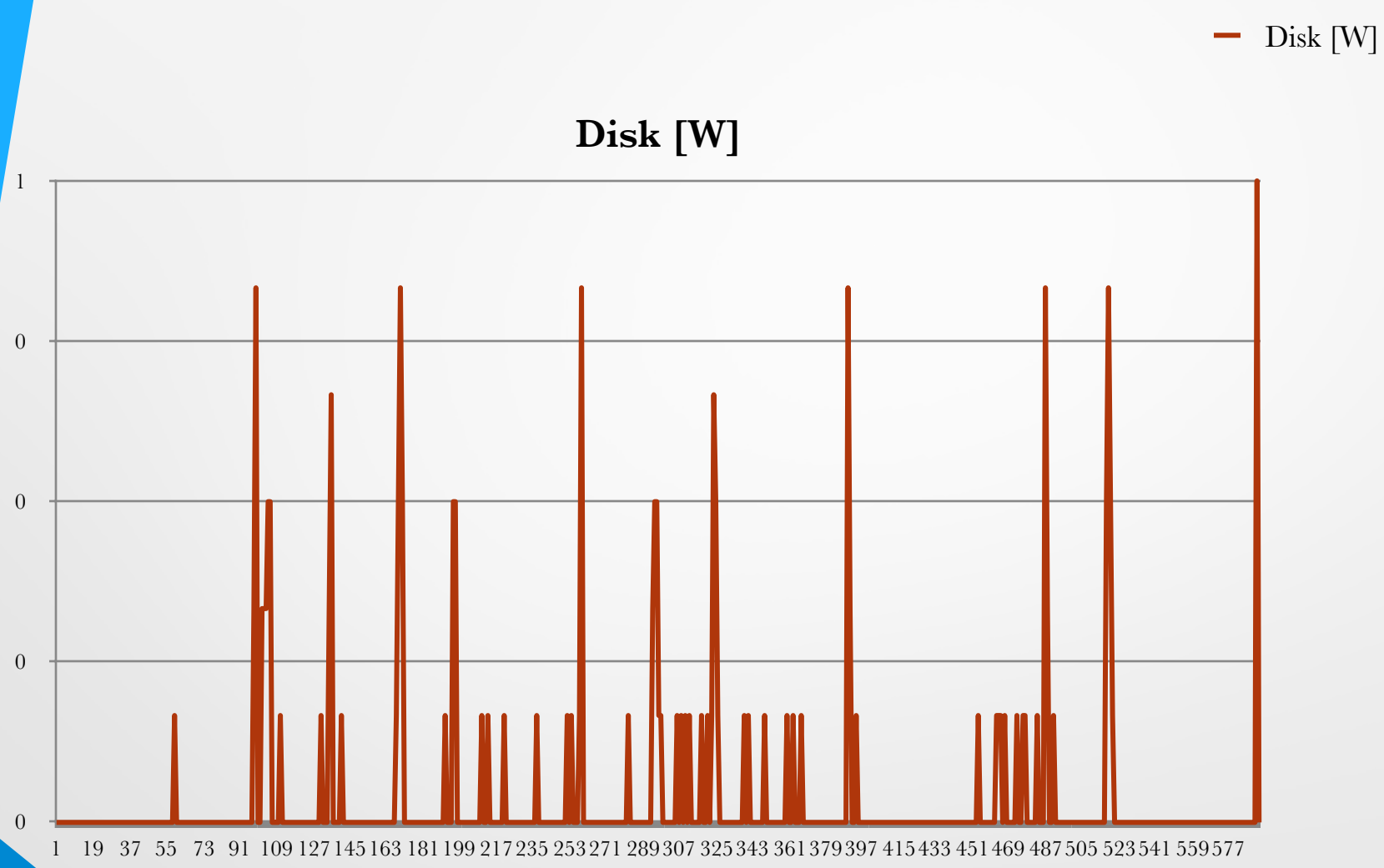
More details can be found in Intellectual output 1 of the project 2017-1-SK01-KA203-035402: Focusing Education on Composability, Comprehensibility and Correctness of Working Software


View Navigate Code Analyze Refactor Build Run Tools VCS
cvjfx src sample Main
Joulemeter
Calibration
Power Usage
About
Power model from 20. 1. 2018 14:35:33
Component Power Usage (Watts)
CPU: 1,0
Monitor: 7,1
Disk: 0,0
Base: 9,3
Total: 17,4
Application Power (CPU only)
Enter program name (as seen in Task Manager Processes tab):
java.exe
Stop
Power: 0,5
Save power data: click Browse to enter filename.
Browse
Power data file: C:\csaba\pwridea64_2.csv
Stop Saving
Another mirror

Application level measurement



More details can be found in Intellectual output 1 of the project 2017-1-SK01-KA203-035402: Focusing Education on Composability, Comprehensibility and Correctness of Working Software



More details can be found in Intellectual output 1 of the project 2017-1-SK01-KA203-035402: Focusing Education on Composability, Comprehensibility and Correctness of Working Software

Component level measurement

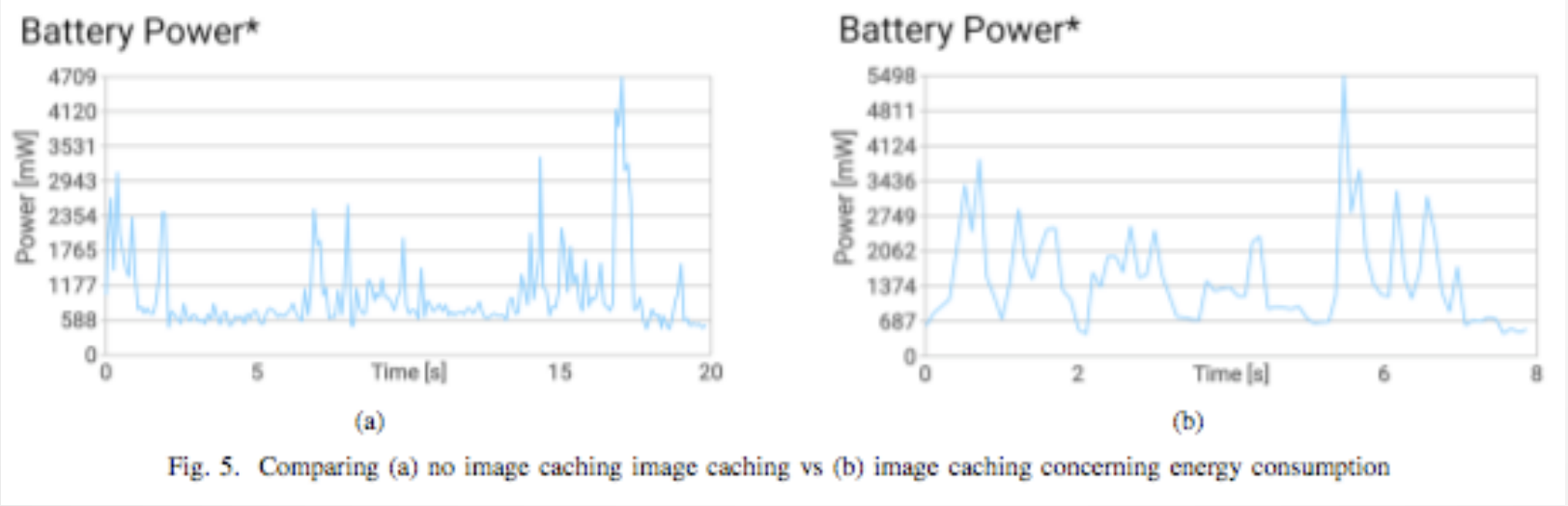


Fig. 5. Comparing (a) no image caching image caching vs (b) image caching concerning energy consumption

- Test oracles
- Comparing different versions
- The driver of energy (r)evolution

More details can be found in Intellectual output 2 of the project 2017-1-SK01-KA203-035402: Focusing Education on Composability, Comprehensibility and Correctness of Working Software

Code level measurement

- Which version of an algorithm is consuming less energy?
- Is it more efficient to store objects in an array than in a list?
- How significantly does the length of execution impact on the consumption measured when generating file MD5/SHA-n values?

Scaling up

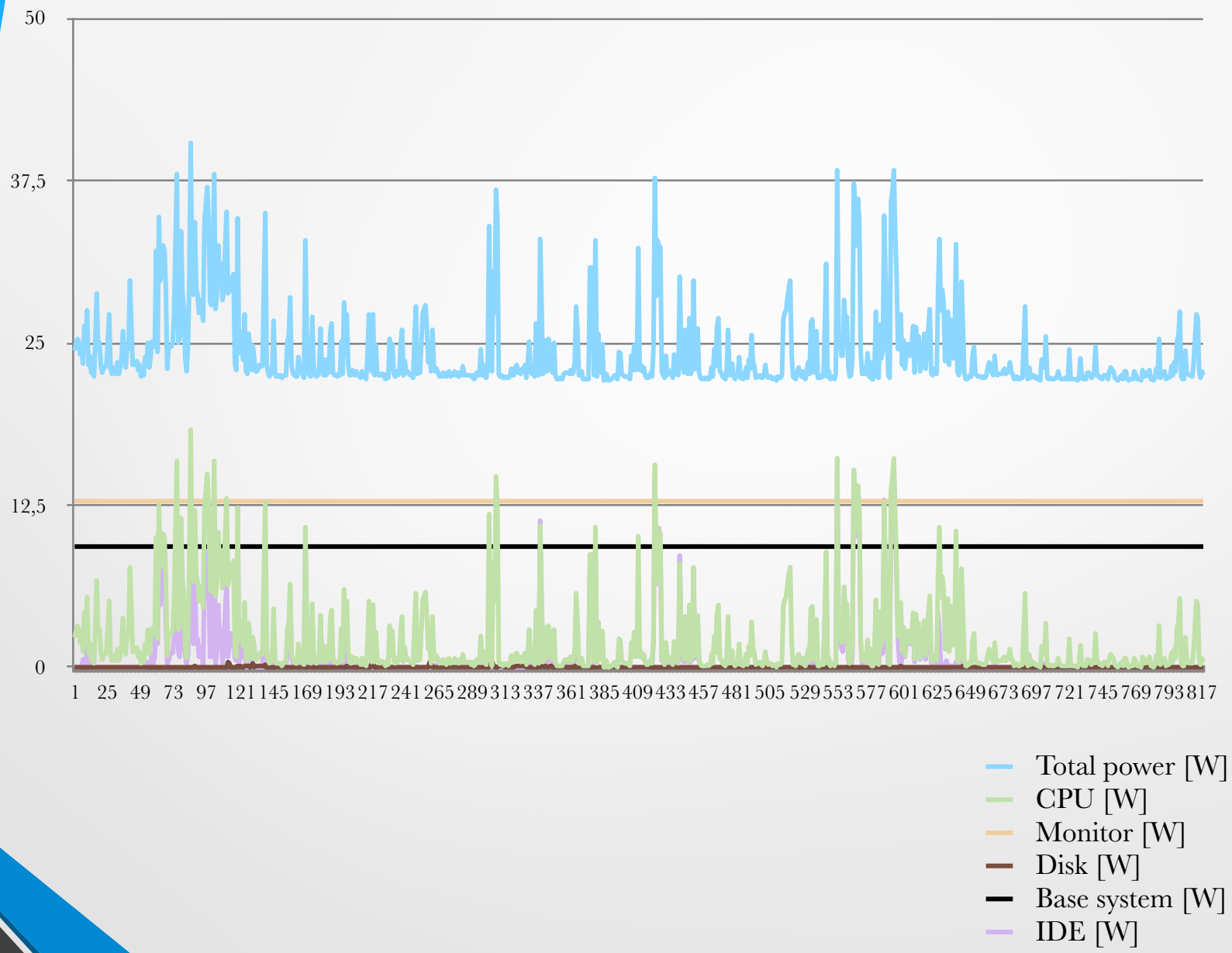
- Usual energy efficiency measurement focuses on software or hardware products.
- But, in our case we will measure the development (host) system's energy efficiency using a black-box testing method.
- We start the measurement before starting the browser and the IDE and we will stop measuring after closing all used tools.

More details can be found in Intellectual output 2 of the project 2017-1-SK01-KA203-035402: Focusing Education on Composability, Comprehensibility and Correctness of Working Software

During development it is normal to compile and run an application many times and use other design and testing tools as well, which will have an effect on energy consumption. The goal of our measurements is to point out this energy.

Process level measurement

More details can be found in Intellectual output 2 of the project 2017-1-SK01-KA203-035402: Focusing Education on Composability, Comprehensibility and Correctness of Working Software


50
37,5
25
12,5
0
1 25 49 73 97 121 145 169 193 217 241 265 289 313 337 361 385 409 433 457 481 505 529 553 577 601 625 649 673 697 721 745 769 793 817
Total power [W]
CPU [W]
Monitor [W]
Disk [W]
Base system [W]
IDE [W]

The energy-measured development game

1. Setup the environment
2. Start the energy monitor
3. Develop (think, code, test, fix) for 15 minutes
4. Have a 5 minutes break (stop energy usage monitoring, set up the next one, get a coffee)
5. Finish (for this time) if there is no further idea
6. Repeat (jump to label 2)
7. Analyse collected data (energy efficiency of your development process) inside the team

More details can be found in Intellectual output 1 and 2 of the project 2017-1-SK01-KA203-035402: Focusing Education on Composability, Comprehensibility and Correctness of Working Software

GSE

The future of

- Research:
  - Unification of principles
  - Standardisation of representation of results
  - Software energy efficiency (label?)
- Industry:
  - Scaling over the path “single developer -> team -> management (project) -> company maturity”
- Academia:
  - Prepare this future by educating the people

Further reading

[1] A. Chandrakasan, S. Sheng, and R. Brodersen, "Low-power cmos digital design," Solid-State Circuits, IEEE Journal of, vol. 27, no. 4, pp. 473– 484, Apr 1992.

[2] V. Tiwari, S. Malik, and A. Wolfe, "Power analysis of embedded software: a first step towards software power minimization," Very Large Scale Integration (VLSI) Systems, IEEE Transactions on, vol. 2, no. 4, pp. 437–445, Dec 1994.

[3] W. Yuan and K. Nahrstedt, "Energy-efficient soft real-time cpu scheduling for mobile multimedia systems," in Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles. ACM, 2003, pp. 149–163.

[4] C. Becker, R. Chitchyan, L. Duboc, S. Easterbrook, M. Mahaux, B. Penzenstadler, G. Rodríguez-Navas, C. Salinesi, N. Seyff, C. C. Venters, C. Calero, S. A. Koçak, and S. Betz, "The karlskrona manifesto for sustainability design," CoRR, vol. abs/1410.6968, 2014.

[5] C. Sahin, P. Tornquist, R. Mckenna, Z. Pearson, and J. Clause, "How does code obfuscation impact energy usage?" in 30th IEEE International Conference on Software Maintenance and Evolution, Victoria, BC, Canada, September 29 - October 3, 2014, 2014, pp. 131–140.

[6] M. L. Vásquez, G. Bavota, C. Bernal-Cárdenas, R. Oliveto, M. D. Penta, and D. Poshyvanyk, "Mining energy-greedy API usage patterns in android apps: an empirical study," in 11th Working Conference on Mining Software Repositories, MSR 2014, Proceedings, May 31 - June 1, 2014, Hyderabad, India, 2014, pp. 2–11.

Some more sources

[7] C. Sahin, L. Pollock, and J. Clause, "How do code refactorings affect energy usage?" in Proceedings of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, 2014, pp. 36:1–36:10.

[8] G. Pinto, F. Castor, and Y. D. Liu, "Understanding energy behaviors of thread management constructs," in Proceedings of the 2014 ACM International Conference on Object Oriented Programming Systems Languages & Applications, ser. OOPSLA '14. New York, NY, USA: ACM, 2014, pp. 345–360. [Online]. Available: http://doi.acm.org/10.1145/2660193.2660235

[9] K. Liu, G. Pinto, and Y. Liu, "Data-oriented characterization of application-level energy optimization," in Proceedings of the 18th International Conference on Fundamental Approaches to Software Engineering, ser. Lecture Notes in Computer Science, vol. 9033, 2015, pp. 316–331.

[10] Luis Gabriel Lima, Francisco Soares-Neto, Paulo Lieuthier, Fernando Castor, Gilberto Melfe, João Paulo Fernandes: Haskell in Green Land: Analyzing the Energy Behavior of a Purely Functional Language. SANER 2016: 517-528

[11] Luis Gabriel Lima, Francisco Soares-Neto, Paulo Lieuthier, Fernando Castor, Gilberto Melfe, João Paulo Fernandes, On Haskell and energy efficiency. Journal of Systems and Software 149: 554-580 (2019)

[12] Rui Pereira, Marco Couto, João Saraiva, Jácome Cunha, João Paulo Fernandes: The influence of the Java collection framework on overall energy consumption. GREENS@ICSE 2016: 15-21

Even more sources

[13] Rui Pereira, Pedro Simão, Jácome Cunha, João Saraiva: jStanley: placing a green thumb on Java collections. ASE 2018: 856-859

[14] D. Li, W. G. J. Halfond, An investigation into energy-saving programming practices for android smartphone app development, in Proc. of the 3rd International Workshop on Green and Sustainable Software, GREENS 2014, ACM, 2014, pp. 46–53.

[15] D. Li, Y. Jin, C. Sahin, J. Clause, W. G. J. Halfond: Integrated energy-directed test suite optimization, in Proc. of the 2014 International Symposium on Software Testing and Analysis, ISSTA 2014, ACM, 2014, pp. 339–350.

[16] M. Santos, J. Saraiva, Z. Porkolab, D. Krupp: Energy Consumption Measurement of C/C++ Programs Using Clang Tooling, in Proceedings of the SQAMIA 2017: 6th Workshop of Software Quality, Analysis, Monitoring, Improvement, and Appli- cations, Z. Budimac, ed., Belgrade, Serbia, 11-13.9.2017, Paper No. 15, 8 pages, also published online by CEUR Workshop Proceedings No. 1938 ISSN 1613-0073.

[17] J.Saraiva, M.Couto, Cs.Szabo, D.Novak: Towards Energy-Aware Coding Practices for Android, Acta Electrotechnica et Informatica, Vol. 18, No. 1, 2018, pp. 19–25. https://doi.org/10.15546/aeei-2018-0003

[18] Cs. Szabo, E.M.M. Alzeyani: Measuring Energy Efficiency of Selected Working Software, Studia Universitatis Babe¸s-Bolyai Informatica, Vol. 63, No. 1, 2018, pp. 5–16. https://doi.org/10.24193/subbi.2018.1.01

[19] Cs. Szabo, J. Saraiva: Focusing software engineering education on green application development, in Conference of Information Technology and Development of Education – ITRO 2017, Novi Sad, Serbia, pp. 165–169, ISBN 978-86-7672-302-7.

TECHNICAL UNIVERSITY OF KOŠICE
Faculty of Electrical Engineering
and Informatics

Liked it? ☺

Csaba.Szabo@tuke.sk

Disclaimer

The information and views set out in this publication are those of the author(s) and do not necessarily reflect the official opinion of the European Union. Neither the European Union institutions and bodies nor any person acting on their behalf may be held responsible for the use which may be made of the information contained therein.

This presentation contains parts of Intellectual output 1, 2 and 4 of the ERASMUS+ project No. 2017-1-SK01-KA203-035402: Focusing Education on Composability, Comprehensibility and Correctness of Working Software.

Co-funded by the
Erasmus+ Programme
of the European Union