Challenges and opportunities for sustainable software

Lago, P.

published in
Proceedings of the Fifth International Workshop on Product LinE Approaches in Software Engineering
2015

DOI (link to publisher)
10.1109/PLEASE.2015.8

document version
Publisher's PDF, also known as Version of record

citation for published version (APA)
Lago, P. (2015). Challenges and opportunities for sustainable software. In *Proceedings of the Fifth International Workshop on Product LinE Approaches in Software Engineering* (pp. 1-2). IEEE Press. https://doi.org/10.1109/PLEASE.2015.8

2015 IEEE/ACM 5th International Workshop on Product LinE Approaches in Software Engineering

Challenges and Opportunities for Sustainable Software

(Keynote)

Patricia Lago
Department of Computer Science,
VU University Amsterdam, The Netherlands
p.lago@vu.nl

ABSTRACT

With the increasing role played by software in supporting our society, its sustainability and environmental impact have become major factors in the development and operation of software-intensive systems. Myths and beliefs hide the real truth behind Green IT: IT is energy-inefficient because software is developed to make it so – intentionally or not. But how far are we from being able to control software energy-efficiency? What makes software greener? How can we transform measuring software energy consumption in a general practice? What architectural design decisions will result in more sustainable systems? How can we ensure that new-generation software will be both cloud-ready and environmental-friendly? and How can we make evident the economic and social impact of developing software with 'energy in mind'? These are a few of the challenges ahead for a more sustainable digital society. This talk will discuss them, hence drawing directions for exciting challenges, promising opportunities, and ultimately inspiring research.

WHERE ARE WE, REALLY?

It is impossible to imagine society without software and the IT resources to run it. While we can measure the impact of IT resources on energy consumption, we still have to understand the impact of software and its properties.

What we know for sure is that software is being developed without taking energy into consideration. This is evident if we think of the incredible energy optimizations of hardware over the years, which have been continuously negated by software products[1]. We notice this every day. For example, playing media on our mobile devices drains battery in no time. Or, if we look at the various generations of operating systems, they offer very similar sets of features to end-users while they demand much heavier IT resources.

In short, there is a huge hidden potential. A study in Berkeley estimated to 87% the potential energy savings of migrating US business applications to the cloud [1]. However, it is unclear which applications should be re-engineered, what should go in the cloud and what should remain locally. In other words, how this potential can be engineered. The following explores a few of the many related challenges and opportunities.

I. CHALLENGES AND OPPORTUNITIES

Green Software Labels are farther than we think. In spite of the many publications in the field of green software, most provide only partial (if any) information about e.g. how to develop software that is energy efficient; what metrics, tools or contextual configurations should be considered when measuring or estimating the energy consumption of (certain types of) software systems or applications; and ultimately what is the *real* impact of software applications on the energy efficiency of the computing resources they need for execution. As a result we are witnessing a paradox. From afar, researchers and practitioners have started believing that much has been done already, and the lack of significant adoption in practice would suggest that the impact of software is negligible. This further feeds the myth that major gains are to be searched in hardware optimization, or in using renewable energy resources instead. We are forgetting that computing resources and infrastructures are there because software needs them, and if software is bloat and inefficient any optimization will be just wasted. To be able to recognize the *level of greenness* of a software system we must be able to define theories about the relevant properties, knowledge about which design and coding decisions result in less energy consumption, and classification of contextual information (e.g technology characteristics, versions, dependencies) [2].

Software Integration in a changing context. Thanks to a combination of favorable phenomena (fast technology development, widespread adoption, society digitalizations, extreme availability of software services), software systems are becoming smaller and smaller while the context they interact with, and have to manage, becomes larger and increasingly complex. This has important implications on the software architecture of any modern application: we still have to decide on what to deliver in terms of functional and quality requirements; differently, many of these requirements will be adopted from external providers; will eventually change over time; and will dynamically adapt depending on personalization (per user

---

[1] Wirth's Law, see http://en.wikipedia.org/wiki/Wirth's_law

978-1-4673-7061-5/15 $31.00 © 2015 IEEE
DOI 10.1109/PLEASE.2015.8

or per organization), on contextual changes (location-aware or situation-aware), and on evolving requirements (interacting with changing IoT components). Being able to manage dynamic, flexible integration of such a continuously changing context poses novel challenges for a.o. software architecture, SPL engineering, enterprise application integration, and system-driven requirements elicitation. Examples of domains already expecting solutions include smart grids, cyber-foraging [3], e-health, sustainable data management [4], and green cloud provisioning [5]. From a software engineering perspective, it will be essential to make design decisions that address the various quality properties of a sustainable system, from both technical, environmental, economic and social perspectives [6].

Knowledge and Awareness are key. To progress in a still-pioneering field like green software engineering we need to build factual information for at least two types of purposes. The first is for IT professionals to both develop software that is energy-efficient in the first place, and modernize pre-existing software to improve its energy footprint. To this end, research is needed to create engineering practices like design patterns and architectural tactics, which deliver greener software; tools that help engineers monitor energy consumption of the software in use as well as estimate the potential footprint while developing it. In spite of common *beliefs*, current state of the art is largely insufficient in providing such tools especially considering the increasing complexity of software execution contexts that make traceability an even more complex problem that it used to be.

The second purpose is for the users of software systems and applications to become aware of the level of greenness of the software they buy and of the their usage behavior. Various organizations (like CEPIS, Data Center Alliance, Milieukeur) are making initial steps to e.g. create reference practices and criteria toward green software labeling. Still we need much more software engineering experimentation before a sound know-how is available. Also, software itself must feed back to its users its own energy consumption so that more sustainable usage behaviors can be recommended.

Only when we will have a sound body of knowledge to develop-measure-label green software we will be able to claim that the field is mature.

BIOGRAPHY

Patricia Lago is full professor at VU University Amsterdam. Her research interests include software- and service-oriented architecture, sustainability and green software engineering. She has a PhD in Control and Computer Engineering from Politecnico di Torino, Italy. She is chair of the IEEE/IFIP WICSA Steering Committee, member of the Steering Committee of the ICT4S conference series, of the IFIP 2.10 Working group on Software Architecture, the IFIP 2.14 Working group on Services-based Systems, and the Dutch Knowledge Network Green Software. She is initiator and co-organizer of the GREENS workshop series on Green and Sustainable Software, and co-organizer of the first Track on Software Engineering in Society (SEIS) at ICSE 2015. She is in the Editorial Board of the Journal of Systems and Software (Elsevier), and the Journal of Software - Evolution and Process (Wiley). More info at: www.cs.vu.nl/~patricia

REFERENCES

[1] E. Masanet, A. Shehabi, L. Ramakrishnan, J. Liang, X. Ma, B. Walker, V. Hendrix, and P. Maantha, "The energy efficiency potential of cloud-based software: A u.s. case study," Laurence Berkeley National Lab, Berkeley, California, Tech. Rep., June 2013.

[2] G. Procaccianti, P. Lago, A. Vetró, D. Méndez Fernandez, and R. Wieringa, "The green lab: Experimentation in software energy efficiency," in *International Conference on Software Engineering (ICSE)*, May 2015, p. 2.

[3] G. Lewis and P. Lago, "A catalog of architectural tactics for cyber-foraging," in *International Conference on the Quality of Software Architectures*. ACM, May 2015.

[4] Q. Gu, P. Lago, H. Muccini, and S. Potenza, "A categorization of green practices used by Dutch data centers," in *3rd International Conference on Sustainable Energy Information Technology*. Elsevier, jun 2013, p. 7.

[5] G. Procaccianti, P. Lago, and G. Lewis, "Green architectural tactics for the cloud," in *Proceedings of the 11th Working IEEE/IFIP Conference on Software Architecture (WICSA)*, Apr. 2014.

[6] P. Lago, S. Akinli Kocak, I. Crnkovic, and B. Penzensradler, "Framing sustainability as a software quality property," *Commun. ACM*, p. 9, to appear 2015.