Self-Adaptive Energy-Efficent Applications: The HADAS Developing Approach

Jose-Miguel Horcas, Mónica Pinto, Lidia Fuentes
Dpto. de Lenguajes y Ciencias de la Computación
Universidad de Málaga, CAOSD Group
Málaga, SPAIN
{horcas, pinto, lff}@lcc.uma.es

Nadia Gámez
Universidad Internacional de la Rioja
La Rioja, SPAIN
nadia.gamez@unir.net

***Abstract*—Software systems have a strong impact on the energy consumption of the hardware they use. For this reason, software developers should be more aware of the energy consumed by their systems. Moreover, software systems should be developed to be able to adapt their behavior to minimize the energy consumed during their execution. This paper illustrates how to address the problem of developing self-adaptive energy-efficient applications using the HADAS approach. HADAS makes use of advanced software engineering methods, such as Dynamic Software Product Lines and Aspect-Oriented Software Development. The main steps of the HADAS approach, both during the design of the application and also at runtime are illustrated by applying them to a running case study.**

Keywords—energy-efficient applications, self-adaptation, HADAS, Dynamic Software Product Lines, Aspect-Oriented Software Development

I. INTRODUCTION

The percentage of global emissions attributable to Information Systems is expected to further increase in the coming years, due to the proliferation of Internet-connected devices omnipresent in our daily lives [1]. Although software systems do not directly consume energy, they strongly affect the energy consumption of the hardware [2]. So developers should be more aware of the energy consumed by these systems during their usage, and try to develop energy-efficient applications that adapt their behavior to minimize the energy consumed during their execution, i.e., develop self-greening applications [3,4].

Regrettably, there is a narrow view of developers and users about their responsibility for the energy consumed during application execution. They rarely address energy efficiency as some recent studies show [3,4], principally due to a lack of appropriate methodologies and tools which would help them to produce self-adaptable green software at runtime. Therefore, although software energy efficiency is becoming increasingly important, development processes of self-greening systems supported by tools are still in their infancy. There are plenty of approaches that present experimental results about how to optimize energy consumption at design time [5,6,7], but little effort has been made to explore reusable solutions of runtime energy optimizations.

Indeed, once deployed, the energy consumed by a system depends on several factors, determined mainly by the usage context [8]. It depends, for example, on the amount of data the system needs to store, transfer or query, or on how the user interacts with the system. So, the user behavioral pattern impacts very strongly on the final energy expenditure of applications. Therefore, applications should not only be prepared at design time to be energy-efficient; they also need to be self-adaptable to the runtime context usage.

This paper illustrates how advanced software engineering methods, such as Dynamic Software Product Lines (DSPLs) [9] and Aspect-Oriented Software Development (AOSD) [10], can help address the problem of developing self-adaptive energy-efficient applications. Concretely, we present the HADAS approach for the analysis and development of self-adaptive energy-efficient applications. HADAS proposes to collect energy-related information at design time and use it at runtime to adapt the application behavior to the real energy consumption. HADAS bases on the concepts of *runtime energy hotspot* and *energy consuming concerns*. A *runtime energy hotspot* is a point in the application that under certain conditions can consume much energy and, if these conditions change at runtime it is possible to reduce this energy consumption by modifying the application components. The *energy consuming concerns* are the concerns that model the runtime energy hotspots at design time. They could be designed in different ways, with different energy consumption that depends on some input parameters such as size of type of data. All the alternative design solutions for every energy consuming concern are stored in HADAS so that at design time application developers can perform a sustainability analysis of the different variants. HADAS then generates the initial application configuration. This sustainability analysis will also help to identify those situations where the energy expenditure strongly depends on some parameters that can vary at runtime. This information will be used by the developer to specify the self-greening rules that will trigger a reconfiguration at runtime.

After this introduction, in Section II we discuss the main challenges that arise in the development of our approach. Then, in Sections III, IV and V we describe how HADAS addresses these challenges. Finally, our conclusions are presented in section VI.

REFERENCES

[1] Q. Li and M. Zhou. The survey and future evolution of green computing. In Proceedings of the IEEE/ACM International Conference on Green Computing and Communications, GreenCom'11, pages 230–233, 2011.

[2] K. Grosskop, J. Visser. Identification of Application-level Energy-Optimizations. In Proceedings of the conference on ICT for Sustainability – ICT4S'13, pages 101-107, 2013

[3] I. Manotas, C. Bird, R. Zhang, D. Shepherd, C. Jaspan, C. Sadowski, L. Pollock, and J. Clause. An empirical study of practitioners' perspectives on green software engineering. In Proceedings of the 38th International Conference on Software Engineering - ICSE '16, pages 237–248, 2016.

[4] C. Pang, A. Hindle, B. Adams, and A. Hassan. What do programmers know about software energy consumption? IEEE Software, 33(3):83–89, may 2015

[5] K. Grosskop and J. Visser. Identification of application-level energy optimizations. Proceeding of ICT for Sustainability (ICT4S), pages 101–107, 2013.

[6] E. Jagroep, J. M. van der Werf, S. Brinkkemper, L. Blom, and R. van Vliet, "Extending software architecture views with an energy consumption perspective: A case study on resource consumption of enterprise software," Computing, pp. 1–21, 2016.

[7] A. Noureddine and A. Rajan. Optimising energy consumption of design patterns. In Proceedings of the 37th International Conference on Software Engineering - Volume 2, pages 623–626, 2015.

[8] S. Hasan, Z. King, M. Hafiz, M. Sayagh, B. Adams, and A. Hindle. Energy profiles of Java collections classes. In Proceedings of the 38th International Conference on Software Engineering - ICSE '16, pages 225–236, 2016.

[9] S. Hallsteinsen, M. Hinchey, S. Park, and Klaus Schmid. "Dynamic Software Product Lines". Computer 41, 4 (April 2008), 93-95.

[10] G. Kiczales, J. Lamping, A. Mendhekar, C. Maeda, C. Lopes, J.M. Loingtier, and et. al. Aspect-oriented programming. In: ECOOP – Object Oriented Programming, vol. 1241. 1997. p. 220–42.

[11] S. Götz, C. Wilke, S. Cech, and U. Aßmann, "Runtime variability management for energy-efficient software by contract negotiation," in *CEUR Workshop Proceedings*, 2011, vol. 794, pp. 61–72.

[12] A. El Kouche, L. Al-Awami, and H. Hassanein, "Dynamically Reconfigurable Energy Aware Modular Software (DREAMS) Architecture for WSNs in Industrial Environments," Procedia Comput. Sci., vol. 5, pp. 264–271, 2011.

[13] S. J. Chinenyeze, X. Liu, and A. Al-Dubai, "An Aspect Oriented Model for Software Energy Efficiency in Decentralised Servers," in 2nd International Conference on ICT for Sustainability - ICT4S, 2014, pp. 112–119.

[14] Ø. Haugen, B. Møller-Pedersen, J. Oldevik, G. K. Olsen and A. Svendsen. Adding Standardized Variability to Domain Specific Languages. In Proceedings of the 12th International Software Product Line Conference, SPLC'08, pages. 139-148, 2008

[15] R. H. Reussner, S. Becker, J. Happe, R. Heinrich, A. Koziolek, H. Koziolek, M. Kramer, and K. Krogmann. Modeling and Simulating Software Architectures - The Palladio Approach. MIT Press, Cambridge, MA, October 2016.

[16] N. Bencomo, R. France, B. H. Cheng, U. Aßmann (eds.). Models@run.time, LNCS, vol. 8378, pages 279–318. Springer, Heidelberg, 2014