On the Energy Consumption of Design Patterns

Christian Bunse
University of Applied Sciences Stralsund
Zur Schwedenschanze 15
18435 Stralsund

Sebastian Stiemer
University of Applied Sciences Stralsund
Zur Schwedenschanze 15
18435 Stralsund

March 29, 2013

Introduction. Energy is one of the most limiting factors for information & communication technologies in general and, more specifically for mobile devices such as Smartphones. In most application scenarios, mobile devices do not have a permanent power supply but use rechargeable batteries. Due to the increasing hardware performance and other device properties energy requirements increase further. However, software utilizes hardware and therefore directly affects the energy requirements of the entire system.

Energy-aware software development, energy-aware algorithms and energy-aware sensor substitution are only three examples for recent research that try to reduce energy requirements by optimizing the software rather than the hardware. Energy consumption is an important system property, that has already to be addressed in the early stages of development. In turn, this requires knowledge on best-practices and structures for developing energy-efficient software systems.

Following [1], patterns play many roles in software development: they provide a common vocabulary, reduce system complexity, constitute a base for building reusable software, and act as building blocks. It is a common believe that software quality increases by pattern application. But, the impact of a pattern onto properties such as performance, security or *energy consumption* is widely unknown.

In this paper, we compare the impact of design patterns onto the energy consumption of mobile (i.e., smartphone based) applications. Small apps for the Android platform were developed that either use or not use a specific pattern. The energy consumption of these apps was measured by using the PowerTutor-App, developed at the University of Michigan. The results regarding the selected pattern subset (*facade, abstract factory, observer, decorator, prototype, and template method*) are interesting. Especially the decorator pattern show a significant negative impact onto energy consumption.

Background. The research presented in this paper is rooted in the research fields of energy-aware computing and energy requirement ascertainment techniques. Many energy-aware approaches either try to reduce energy needs by substituting hardware resources [2], or by balancing energy requirements and information quality [3]. In [4] it is illustrated that a simple substitution of the resources central processing unit, and memory helps to reduce the amount of energy required. The authors of [5] showed that processing less precise data requires less energy, and also present a setup for measuring the energy requirements of core and memory of a micro controller based system, running sorting algorithms.

Energy measurement for software can either be based on hardware or software-based approaches [6]. [7] provides an approach for generating energy models for mobile systems by using the smart battery interface accompanied by means to achieve accuracy. Tools such as the Nokia Energy Profiler or PowerTutor [8] enable developers to monitor power consumption. These tools are based on an underlying cost model that, itself, is derived by analyzing a specific device (i.e. Nokia S60). [9] presents a power modeling scheme and an implementation that allows fine-grained energy accounting.

In software engineering, a pattern is a general repeatable solution to a commonly occurring problem [1]. A pattern is an abstract template that needs to be refined and adapted before it can be integrated into the code. Patterns focus on descriptions that communicate the reasons for design decisions. But, little is known about their impact onto system properties [10]. [11] examines the impact of using design patterns onto performance and provide a process for pattern selection. [12] presents an approach for mapping software design to power consumption and analyze how design decisions affect an application's energy usage. [13] analyzes six design patterns and explore the effect of them on energy consumption and performance.

Experiments Existing approaches have shown that pattern usage impacts energy consumption at least when it comes to embedded systems [13] or C++ based code [12]. The goal of our research was evaluating the impact of patterns onto the energy consumption of mobile systems that use Java. The underlying hypothesis being that using or not using a specific pattern will significantly change energy needs. Results can then be used as a starting point for further exploration in order to identify why and how the design patterns impact energy consumption. We selected a subset of the Gamma patterns (*facade, abstract factory, observer, decorator, prototype, and template method*). To evaluate the impact of a single pattern we developed two, comparable applications for each pattern that either use or not use the pattern. Implementation loosely followed standard implementations available in textbooks. Energy consumption was then measured by using the PowerTutor App running on various phones (Nexus One, Galaxy SII, Transformer) whereby experiments and measurements were coordinated by a framework.

				Difference (%)	
Pattern	System	Overall Time	Energy (J)	Time	Energy
Facade	"Clean"	15,40	395,60	1,80	2,50
	Pattern	15,70	405,60		
Abstract Factory	"Clean"	13,50	342,10	14,20	15,90
	Pattern	15,40	396,60		
Observer	"Clean"	15,10	373,70	0,90	0,10
	Pattern	15,20	373,90		
Decorator	"Clean"	15,20	374,00	132,40	133,60
	Pattern	35,40	873,80		
Prototype	"Clean"	11,20	271,80	33,00	33,20
	Pattern	14,90	362,00		
Template Method	"Clean"	15,00	366,40	0,30	0,10
	Pattern	15,10	366,70		

Figure 1: Experimental Results

Table 1 shows the results of the first experiment series. While measurements for patterns such as Facade, Observer or Template Method show no difference, the results for the Prototype and Decorator show a large difference in time and energy needs (15,2 vs. 35.4 and 374 vs. 873,8). The reason for the gross difference might be the large amount of objects instantiations and method calls of the pattern-based system. This supports the findings of [4] that memory consumption using the heap as well as the garbage collector are energy-intensive operations that also have a negative impact onto performance. Although interesting, our results can only be used as an indicator due to several threats to validity. Implementation and measurement might not be generalizable. This warrants further research.

Summary and Conclusions. In this paper, we presented a case study that examined the impact of design pattern application onto a systems energy consumption. Two groups of apps, either using or not using a pattern, were developed and measured. The results for a distinct subset of the Gamma patterns showed, that especially the decorator pattern has a negative impact on the energy needs of an app. Due to the low temporal resolution of the software measurement method, evaluations with a short runtime are error-prone and the used systems might not be representable. However, the interpretation of the evaluation results supports our hypothesis and justifies further research. Using patterns is not always a good idea. Their selection should not solely be based onto function and structure but also according other properties. Although the results of our study are not generalizable, the results indicate, that further research is warranted that examines the impact of patterns regarding different platforms and applications. Results can then be used for meta-analysis.

During our study, we were able to support our hypotheses but, in turn, also identified issues that warrant further research. First, the robustness of our approach regarding the hardware platform has to be evaluated. Furthermore, it is interesting to take a deeper look into the characteristics of the energy requirements of other patterns or idioms. Results might then be used to define anti-patterns regarding software energy consumption.

References

[1] E. Gamma, R. Helm, R. E. Johnson, and J. M. Vlissides, "Design patterns: Abstraction and reuse of object-oriented design," in *ECOOP*, 1993.

[2] C. Bunse and H. Höpfner, "Resource substitution with components — optimizing energy consumption," in *ICSOFT '08 Proc.*, pp. 28–35, INSTICC, 2008.

[3] J. P. Sousa, R. K. Balan, and D. G. et al, "User Guidance of Resource-Adaptive Systems," in *ICSOFT '08 Proc.*, INSTICC, 2008.

[4] C. Bunse, H. Höpfner, S. Roychoudhury, and E. Mansour, "Energy efficient data sorting using standard sorting algorithms," in *Software and Data Technologies*, pp. 247–260, Springer, 2011.

[5] H. Höpfner and C. Bunse, "Energy Aware Data Management on AVR Micro Controller Based Systems," *ACM SIGSOFT SEN*, vol. 35, May 2010.

[6] C. Bunse and H. Höpfner, "Ocemes: Measuring overall and component-based energy demands of mobile and embedded systems," in *GI-Jahrestagung*, 2012.

[7] M. Dong and L. Zhong, "Self-constructive high-rate system energy modeling for battery-powered mobile systems," in *Proc. of the 9th Intl. conference on Mobile systems, applications & services*, ACM, 2011.

[8] L. Zhang, B. Tiwana, and Z. Q. et al., "Accurate Online Power Estimation and Automatic Battery Behavior Based Power Model Generation for Smartphones," in *CODES/ISSS '10 Proc.*, ACM, 2010.

[9] A. Pathak, Y. C. Hu, and M. e. a. Zhang, "Fine-grained power modeling for smartphones using system call tracing," in *Proc. of the 6th conference on Computer systems*, ACM, 2011.

[10] D. Gross and E. S. K. Yu, "From non-functional requirements to design through patterns.," *Requir. Eng.*, vol. 6, no. 1, pp. 18–36, 2001.

[11] N. Mani, D. C. Petriu, and M. Woodside, "Towards studying the performance effects of design patterns for service oriented architecture," in *Proc. of the 2nd WOSP/SIPEW intntl conference on Performance engineering*, ICPE '11, ACM, 2011.

[12] C. Sahin, F. Cayci, and I. L. M. e. a. Gutierrez, "Initial explorations on design pattern energy usage," in *1st International Workshop on Green and Sustainable Software (GREENS)*, ACM, 2012.

[13] A. Litke, K. Zotos, and E. C. et al, "Energy consumption analysis of design patterns," in *Proc. World Academy of Science, Engineering & Technology*, 2005.