Predicting Energy Consumption by Extending the Palladio Component Model

Felix Willnecker[1], Andreas Brunnert[1], Helmut Krcmar[2]

[1]fortiss GmbH, Guerickstr. 25, 80805 München, Germany
{willnecker, brunnert}@fortiss.org
[2]Technische Universität München, Boltzmannstr. 3, 85748 Garching, Germany
krcmar@in.tum.de

Abstract: The rising energy demand in data centers and the limited battery lifetime of mobile devices introduces new challenges for the software engineering community. Addressing these challenges requires ways to measure and predict the energy consumption of software systems. Energy consumption is influenced by the resource demands of a software system, the hardware on which it is running, and its workload. Trade-off decisions between performance and energy can occur. To support these decisions, we propose an extension of the meta-model of the Palladio Component Model (PCM) that allows for energy consumption predictions. Energy consumption is defined as power demand integrated over time. The PCM meta-model is thus extended with a power consumption model element in order to predict the power demand of a software system over time. This paper covers two evaluations for this meta-model extension: one for a Java-based enterprise application (SPECjEnterprise2010) and another one for a mobile application (Runtastic). Predictions using an extended PCM meta-model for two SPECjEnterprise2010 deployments match energy consumption measurements with an error below 13 %. Energy consumption predictions for a mobile application match corresponding measurements on the Android operating system with an error of below 17.2 %.

1 Introduction

Energy efficiency of software systems becomes a growing software engineering challenge [BVD+14]. Energy consumption of Information and Communication Technology (ICT) is rising due to higher demand in data centers, networks, and consumer devices like mobile devices [SNP+09, WBK14]. Therefore, there is need to investigate the energy saving potential of software systems [JGJ+12]. Today, hardware manufactures increase the capabilities of the hardware and simultaneously increase the energy efficiency. Operating system (OS) providers implement energy saving modes to increase the energy efficiency of the overall system. Though, neither optimization on hardware nor on OS level can compensate the rising demand, yet decrease the energy consumption of ICT systems [GJJW12]. Therefore, software optimizations in terms of energy are investigated more recently as the application software ultimately causes the energy demand on hardware and OS level [PLL14].

Proc. SOSP 2014, Nov. 26–28, 2014, Stuttgart, Germany

Performance of software systems and energy efficiency are often referred to as contradicting optimization goals [HSB12]. For example, the response time of an enterprise application can be decreased by duplicating the number of replicas of the system. By introducing additional replicas, the energy consumption of the corresponding system is directly increased. In contrast, enhancing the efficiency of a system in terms of performance metrics can also decrease the energy consumption of a system when a components resource demand decreases. Thus, performance metrics and energy consumption rely on the same underlying parameters of resource demand and hardware capabilities. This allows us to adapt and extend technologies used for performance evaluation to simulate the power demand of software systems and therefore predict the energy consumption of such systems.

Predicting performance metrics like response time, throughput or hardware utilization are core capabilities of the PCM modeling environment (Palladio-Bench) [BKR09]. This work proposes an extension of the PCM meta-model called power consumption model and corresponding extensions of the Palladio-Bench to take this meta-model extension into account. The power consumption model describes the power demand of hardware servers that are simulated. Furthermore, we extend the Palladio-Bench to calculate the software's energy consumption based on the resource utilization of the components and the power consumption model [BKR09].

This paper starts by introducing the proposed extension of the PCM meta-model including extensions of the Palladio-Bench for the generation of an energy report. We evaluate this extension with an enterprise application to demonstrate that this extension is accurate for data centers and with a mobile application to show the applicability for mobile devices. This work closes with an outline of related approaches, a summary and future work.

2 Meta-model Extension

The current state of the PCM meta-model cannot predict the energy consumption of a software system. To determine the energy consumption, we specify the power consumption of the hardware resources. A resource's electrical power consumption depends on its usage, caused by the resource demand of the software. Hence, knowing the power consumption P of a certain utilization level of a resource allows to calculate the energy consumption E over the considered time T as presented in equation 1.

$$E = \int_0^T P(t)\,\mathrm{d}t \tag{1}$$

The correlation between resource utilization and energy demand can be approximated for server systems with a linear model [FWB07, RSR$^+$07]. Simple models, in which only the CPU utilization is considered for the power consumption calculation can predict it with high accuracy [RSR$^+$07]. We leverage this correlation and extend the PCM meta-model with a power consumption model to represent such simple models as introduced in our previous work [BWK14]. Power consumption models define the power consumption of a server as a linear equation using utilization values of the server's resources as independent variables [BWK14]. This linear equation calculates the power consumption P_{pred} for a server based on the constant power consumption of a system in idle state C_0 and the sum of the power consumption values of all other resources [BWK14]. The power consumption of a resource is calculated by multiplying a consumption factor C_i with its utilization as presented in equation 2 [BWK14].

$$P_{pred} = C_0 + \sum_{1 \leq i \leq n} C_i * u_i \tag{2}$$

The power consumption calculation based on the utilization of a resource needs to be modified for some resources used in mobile applications. Power consumption of a screen depends on the brightness or color intensity. The power consumption of sensors for the Global Positioning System (GPS) relies on the demanded time of the resource. A constant factor is not sufficient to calculate the power consumption of such sensors [WBK14]. We developed a generalized equation 3 to take non-constant values into account. Therefore, the utilization factor u_i can not only represent the utilization of traditional resources but also represent brightness of a display or other sensor specific behavior such as the accuracy of a GPS sensor. This utilization factor is multiplied by a function that represents the power consumption of a resource (P_{idlei}). Depending on the attached resource, the function can either be a linear or a probabilistic distribution function. Furthermore, we consider the idle state of a device not as constant, but also as a function (P_{idleo}). This accommodates the fact, that background actives of an Operating System (OS) result in varying power consumption. The generalized equation is formulated as follows:

$$P_{pred} = P_{idle0} + \sum_{1 \leq i \leq n} P_{idlei} * u_i \tag{3}$$

In PCM, servers and mobile devices are represented as resource containers. The new power consumption model element is added to the existing resource container meta-model element. This element contains the power consumption characteristics of a server, a mobile device or a network adapter and represents the power consumption as outlined in equation 3.

The same formula can be applied to the power consumption of a network adapter. For mobile devices, one of the largest power consumer is the cellular network adapter [WBK14]. The utilization factor u_i depends on the throughput of the adapter. A typical mobile device is linked to three different network adapters: Wi-Fi, Cellular and Bluetooth each with an independent factor function and an offset function for the power consumption of the adapter in idle state. Therefore, each linking resource is attached to a power consumption model representing the factor function that is multiplied with the throughput of the adapter and offset function for the adapter in idle state for the resource.

To model the power consumption of network adapters and processing resources we extend the PCM resource meta-model. Figure 1 presents our extensions for the PCM meta-model. The extension contains a *Power Consumption Model* and two types of *Power Consumption*

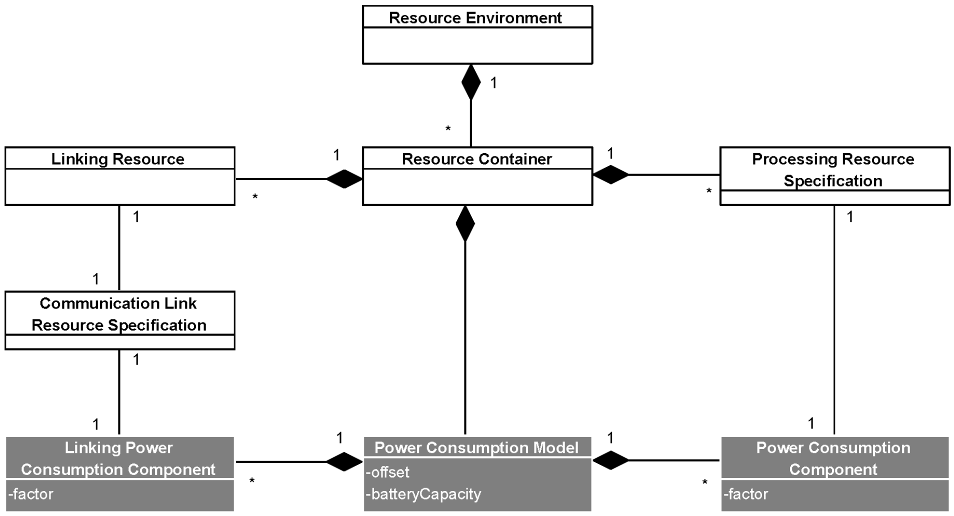


Figure 1: Meta-model Extension for predicting Energy Consumption

Components. The central power consumption model contains the offset function representing the power consumption of the device or linking resource. Furthermore, this model element contains the device's battery capacity in milliWatt-hours (mWh) to calculate the energy consumption of battery depended devices like mobiles and the discharging of its battery. The result of equation 1 can be subtracted from the battery capacity. This calculation provides the remaining battery capacity respectively the loading state.

The power consumption model can reference N *Power Consumption Components* and N *Linking Power Consumption Components*, one for each processing resource respectively one for each linking resource. Each of these components consists of a power consumption factor multiplied with the utilization factor (depending on e.g, utilization, brightness, throughput, demanded time) during simulation. The factor can be constant or a probabilistic function to simulate varying power consumptions.

Figure 2 shows an example of such a power consumption model element for one server with 16 Core Processing Units (CPUs) and one Hard Disc Drive (HDD) [BWK14]. We added a power consumption element to the resource container containing a constant (C_0) representing the idle power consumption of the server [BWK14]. To represent the power consumption of the resources CPU and HDD we added two power consumption component elements [BWK14]. Each of these processing resources is described with the variables of equation 2 [BWK14]. A power consumption component contains a factor (C_i) and a reference to the utilization (u_i) of the resource [BWK14]. Figure 2, shows a power consumption model with a constant C_0 of 200 Watts (W) for the server in idle state, a factor C_{CPU} of 300 W for the CPU and a factor C_{HDD} of 50 W for the HDD [BWK14].

In order to model elements like the GPS sensor or display usage in the resource demanding service effect specifications (RDSEFFs) additional resource types can be added to the

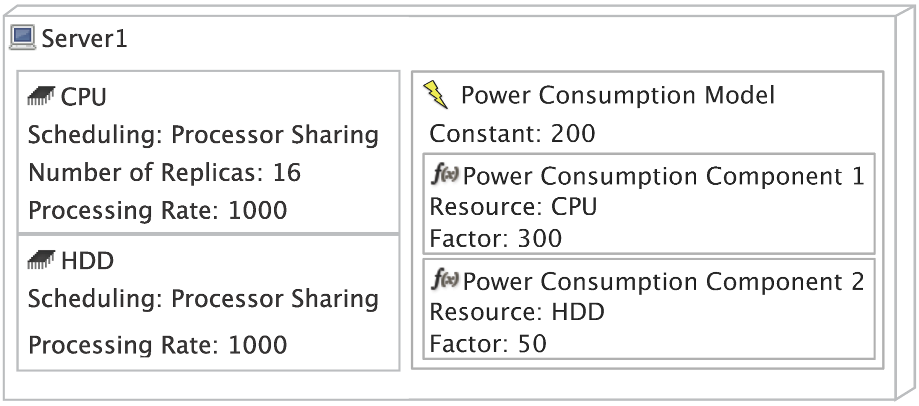


Figure 2: Power consumption model [BWK14]

resource repository of the PCM meta-model. Two key resources that we found necessary for this are purpose are DISPLAY and GPS. The resource demand placed on both resources during a simulation is used for the power consumption calculations depending on a discrete utilization of either 100 % (on) or 0 % (off).

Calculation of the energy consumption is conducted after the simulation run. We extend the Palladio-Bench with an energy consumption report. This report is based on the utilization of all resource. After a simulation run, we calculate the power demand for each resource. We build a sum over all resource-specific power demands of a single resource container and add the power demand of the resource container itself. The total power demand is integrated over the simulation time as described in equation 1. The result of this integral is the energy consumption for a single resource container in a PCM resource environment model. For constant power demands this can be simplified by just multiplying the power demand by the simulation time. The result is a report for the energy consumption of each resource container containing its total power consumption, its total simulation time, its energy consumption and a simple cost calculation.

3 Evaluation

Two experiments are performed to validate the extensions proposed in this work. We evaluate the accuracy for server systems by running a SPECjEnterprise2010[1] benchmark. For mobile devices we use the sports app Runtastic[2] and simulate its energy consumption.

[1]SPECjEnterprise is a trademark of the Standard Performance Evaluation Corp. (SPEC). The official web site for SPECjEnterprise2010 is located at http://www.spec.org/jEnterprise2010.

[2]Runtastic is a trademark of the runtastic GmbH. The official website is located at https://www.runtastic.com/

Table 1: Evaluation environment [BWK14]

Component	**AMD-Based Server**	**Intel-Based Server**
Base System	IBM System X3755M3	IBM System X3550M3
CPU	4 x AMD Opteron 6172	2 x Intel Xeon E5645
CPU Cores	2 x 2.1 GHz	6 x 2.4 GHz
Random Access Memory	256 GB	96 GB
OS	openSuse 12.2	openSuse 12.3
Application Server	6 x JBoss Application Server 7.1.1	
Application	SPECjEnterprise2010	
Database	Apache Derby DB version 10.9.1.0	
Java Virtual Machine	64 bit Java OpenJDK version 1.7.0	

3.1 SPECjEnterprise2010

This section is based on our previous work "Using Architecture-Level Performance Models as Resource Profiles" [BWK14]. Two power consumption models based on the presented PCM meta-model extension were generated for the SPECjEnterprise2010 benchmark application on an AMD-based server and an Intel-based server [BWK14]. To construct these power consumption models we used an application that conducted a calibration run on the target hardware as proposed by Economou et al. [ERKR06]. The run charged the resources independently from each other with varying intensity. While the resources were stressed, resource utilization and power consumption values were collected simultaneously. To collect the power consumption of the server systems, we used the Intelligent Platform Management Interface (IPMI[3]).

After the calibration run, we executed different workloads using varying amounts of users and conducted a simulation using the power consumption model for both hardware environments. We measured and compared the energy consumption and calculated the error of the simulated energy consumption. The used environment for this evaluation is described in table 1 [BWK14].

The power consumption of both servers was predicted with an error below 13 %. Table 2 shows the results for the AMD-based server. The load test ran with 1300 - 3500 clients (C) and caused between 367.55 W and 436.47 W Measured Mean Power Consumption (MMPC). The Simulated Mean Power Consumption (SMPC) lied within 320.26 W and 390.95 W resulting in a Power Consumption Prediction Error (PCPE) between 10.43 % and 12.87 %.

Table 3 shows the results for the Intel-based server. Between 1300 and 4300 clients were used by the load test and caused between 197.05 W and 264.29 W MMPC. The SMPC lied within 175.94 W and 232.69 W resulting in a PCPE between 10.71 % and 11.96 %. The power consumption was relatively stable during the steady state of all load levels, therefore the energy consumption was simply calculated by multiplying the mean power consumption values by the simulation time [BWK14].

[3]http://www.intel.com/design/servers/ipmi/

Table 2: Measured and simulated results for the AMD-based server [BWK14]

C	MMPC	SMPC	PCPE
1300	367.55 W	320.26 W	12.87 %
2300	403.87 W	352.22 W	12.79 %
3300	433.76 W	384.52 W	11.35 %
3500	436.47 W	390.95 W	10.43 %

Table 3: Measured and simulated results for the Intel-based server [BWK14]

C	MMPC	SMPC	PCPE
1300	197.05 W	175.94 W	10.71 %
2300	220.47 W	194.93 W	11.58 %
3300	241.67 W	213.91 W	11.49 %
4300	264.29 W	232.69 W	11.96 %

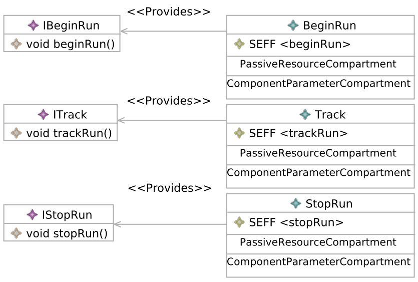
3.2 Runtastic for Android

A power consumption model for two devices running the Android OS is generated and used for the evaluation of power consumption models for mobile devices. Power consumption models for mobile devices can either use vendor profiles[4] provided for the Android OS or stress the resources independently and measure the discharging current. The discharging current C multiplied with the battery voltage V results in the power demand of a hardware resource P as presented in equation 4 [Lei14].

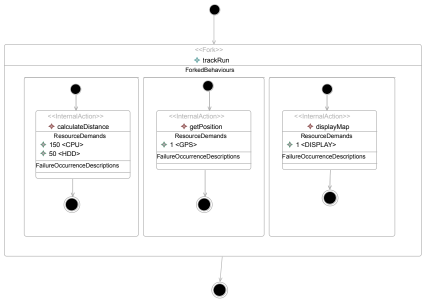
$$P = V * C \tag{4}$$

We use a calibration app to stress the resources of the mobile devices as the vendor profiles accuracy and completeness varied between the devices. The power consumption model is created after the calibration by running a regression on the measured data. An example of such a model with a CPU, display and a GPS sensor for a mobile device is presented in figure 3(c). To calculate the power demand of this device, the utilization of CPU, GPS and display is considered. To calculate the power demand of the CPU its utilization is multiplied with a factor of 800 milliWatts (mW). As soon as the GPS sensor is used, a utilization of 100 % is assumed. The GPS sensor in figure 3(c) therefore consumes 250 mW as soon as it is used. Similarly to the GPS sensor, the display can either be used or not. To take different brightness or color intensity levels into account, the power demand is either 300 mW in 50 % of the cases or 420 mW in the other 50 %. To calculate the power consumption P_{pred} we use the equation: P_{pred} = ((300 * 0.5) + (420 * 0.5)) + 800 * u_{CPU} + 250 * u_{GPS} + ((450 * 0.4) + (520 * 0.6)) * $u_{DISPLAY}$. A CPU utilization of 60 % and a utilization of the GPS and Display of 100 % would thus lead to a predicted power consumption P of 1582 mW. According to equation 1 we build the integral for the predicted power consumption P with the time T of the simulation to determine the energy

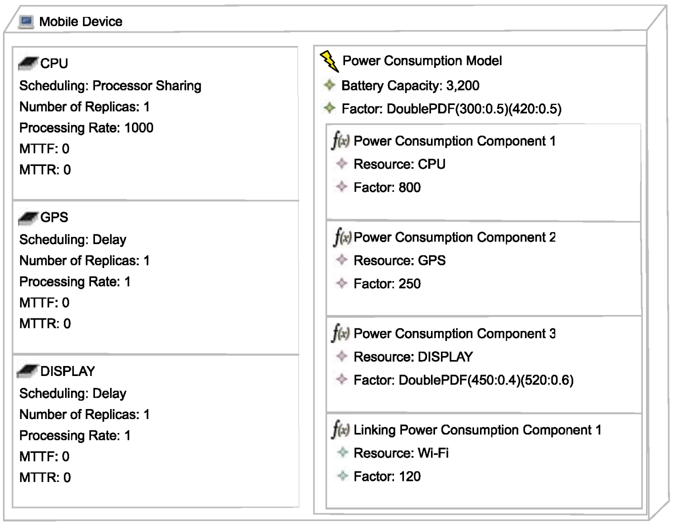
[4] https://source.android.com/devices/tech/power.html



(a) Repository model



(b) RDSEFF



(c) Power Consumption Model

Figure 3: Power Consumption Example for Mobile

consumption E of the system [BWK14]. The battery capacity of the modeled devices is 3.200 mWh. A constant use of the display and GPS sensor while the CPU is active at 60 % utilization would therefore reduce the battery capacity by about 50 % per hour.

The example in figure 3(a) and 3(b) shows a representation of a simplified sports tracking application in PCM that tracks the position via GPS, displays a map and calculates the running distance. Display and GPS are accessible as processing resource. Display and GPS run concurrent to the calculations representing three different threads: A thread for the user interface, one thread for the location tracking and one thread for calculating the distance.

We used Runtastic for Android for this evaluation as this application uses a broad number of resource types. We predicted and measured the power consumption of a 30-minute-

Table 4: Measured and simulated results for Runtastic Android application [Lei14]

Device	MMPC	SMPC	PCPE	BLPE
Samsung Galaxy Tab	1.251 W	1.084 W	13.35 %	0.67 %
LG Google Nexus 5	0.883 W	0.732 W	17.12 %	1.01 %

long Runtastic run on a Samsung Galaxy Tab and a LG Google Nexus 5. The Nexus device runs Android 4.4.4 and the Galaxy Tab has Android 4.3 installed. The evaluation run was conducted with both devices simultaneously. Additionally, both devices were connected to the same network carrier in order to reduce power-relevant variables (e.g, different signal strengths). During the run we collected hardware utilizations data with the Qualcomm Trepn-Profiler[5]. The utilization data is used to build a simple PCM usage, system, repository and allocation model. The repository model was constructed similar to the one shown in figure 3(a) and 3(b) and thus, simply represents the distribution of resource demands placed by the Runtastic application on mobile device hardware. We chose this approach as we did not have access to the source code or debugging interface of the Runtastic application. The usage model contains only one user starting and tracking a run. The power consumption during the run was logged for comparing it with the simulation. We conducted two simulations for the two devices. Both simulations used the same PCM repository model only with different resource environment models and therefore different power consumption models. Afterwards, we calculated the error between the power consumption prediction and the power consumption measured during the run [Lei14].

Table 4 shows the results for the mobile devices. The application causes between 1.251 W and 0.883 W MMPC. The SMPC lies within 1.084 W and 0.732 W resulting in a PCPE between 13.35 % and 17.12 %. The accuracy quality decreases for lower power consumptions. For a 30-minutes run we predicted the battery level of the device with an Battery Level Prediction Error (BLPE) of 0.7 % to 1 % [Lei14].

4 Related Work

This chapter outlines related approaches that measure, compare or predict the energy consumption of software systems.

Capra et al. [CFFG10] compared the energy consumption of customer relationship management (CRM) and database management systems (DBMS). The energy efficiency significantly varied between the compared systems when processing the same workload. They reasoned that energy efficiency should be considered when buying or developing software. The extension for the PCM meta-model proposed in our work takes energy efficiency into account as a key quality metric of a software system.

Jwo et al. [JWH$^+$11] proposed to calculate the energy consumption of an enterprise application by multiplying the time a transaction is processed by mean power consumption

[5] https://developer.qualcomm.com/mobile-development/increase-app-performance/trepn-profiler

of the host. The consumption therefore relies on the workload but still depends on the deployment environment and can therefore not convey a general energy efficiency metric. The extension in our work proposes a similar concept but based on resource demand values instead of response time values.

Johann et al. [JDNK12] proposed energy efficiency measurement methods for software systems. They define energy efficiency as the ratio of "useful" work relative to the energy required to process it. The calculation is based on single methods or components and is evaluated during the development process. They conclude that this method supports the creation of energy efficient applications.

Hönig et al. [HEKSP11] suggest a model-based approach for energy-aware software development for mobile applications. The energy consumption is based on vendor profiles provided by Android device manufactures. The accuracy of these models varies between different manufactures and devices but provides a baseline for the energy consumption calculation. Vendor profiles can be used to create resource environment models including power consumption models for the extension presented in our work.

Josefiok et al. [JSW^{+}13] compared power measurements on different Android devices and OS versions. They discovered that the measurements and the Application Programming Interfaces (API) for monitoring the power consumption differ between manufactures and OS versions. They propose an energy abstraction layer to handle the multitude of APIs and granularity levels. Such a generalized API would help to create resource environment models including power consumption models and therefore predict the energy consumption of mobile applications on a broader scale.

These different approaches show the growing importance of evaluating energy consumption of software systems. Better measurements can help developers to decrease the energy consumption of their systems and increase the overall efficiency. Comparisons help users to choose the software with the best energy efficiency, which can also lead to lower operations costs. Energy consumption prediction capabilities can help to estimate the energy consumption and subsequently efficiency of a system without owning the target environment.

5 Conclusion

This work proposed a PCM meta-model extension to predict the energy consumption of software systems. This extension has been validated for server and mobile systems, for different hardware environments and workloads. The results show that the power consumption of these systems can be predicted with an error below 17.2 %. The evaluation used the SPECjEnterprise2010 benchmark for server systems and the sports tracking application Runtastic for mobile devices. The extension allows to specify the power consumption of hardware resources relative to their utilization. Two additional processing resource types have been added to the meta-model to represent mobile applications.

Constructing and analyzing energy efficient software is becoming an important research area for the software engineering community. The extension proposed in this work allows predicting energy consumption of an application and to predict battery life of a device running a mobile application. The model extensions help developers to understand the varying power demands of different devices and to optimize applications in order to save battery power and reduce profiling effort [WBK14]. PCM meta-model instances for mobile devices are created manually and with a limited number of resources. The calibration application for mobile devices used in this work can create the power consumption model automatically but lacks the means to automatically generate performance models. Such model generators are already available for enterprise applications [BVK13, BHK11]. A future challenge in this research area is to create accurate models for mobile applications automatically.

References

[BHK11] F. Brosig, N. Huber, and S. Kounev. Automated Extraction of Architecture-Level Performance Models of Distributed Component-based Systems. In *Automated Software Engineering (ASE), 2011 26th IEEE/ACM International Conference on*, pages 183–192, Nov 2011.

[BKR09] Steffen Becker, Heiko Koziolek, and Ralf Reussner. The Palladio Component Model for Model-driven Performance Prediction. *Journal of Systems and Software*, 82(1):3 – 22, 2009.

[BVD+14] Andreas Brunnert, Christian Vögele, Alexandru Danciu, Matthias Pfaff, Manuel Mayer, and Helmut Krcmar. Performance Management Work. *Business & Information Systems Engineering*, 6(3):177–179, 2014.

[BVK13] Andreas Brunnert, Christian Vögele, and Helmut Krcmar. Automatic Performance Model Generation for Java Enterprise Edition (EE) Applications. In *Computer Performance Engineering*, pages 74–88. Springer, 2013.

[BWK14] Andreas Brunnert, Kilian Wischer, and Helmut Krcmar. Using Architecture-level Performance Models As Resource Profiles for Enterprise Applications. In *Proceedings of the 10th International ACM Sigsoft Conference on Quality of Software Architectures*, QoSA '14, pages 53–62, New York, NY, USA, 2014. ACM.

[CFFG10] Eugenio Capra, Giulia Formenti, Chiara Francalanci, and Stefano Gallazzi. The Impact of MIS Software on IT Energy Consumption. In *ECIS 2010 Proceedings*, 2010.

[ERKR06] Dimitris Economou, Suzanne Rivoire, Christos Kozyrakis, and Partha Ranganathan. Full-system Power Analysis and Modeling for Server Environments. In *Workshop on Modeling, Benchmarking, and Simulation*, Boston, Massachusetts, USA, 2006.

[FWB07] Xiaobo Fan, Wolf-Dietrich Weber, and Luiz Andre Barroso. Power Provisioning for a Warehouse-sized Computer. In *Proceedings of the 34th Annual International Symposium on Computer Architecture*, ISCA '07, pages 13–23, New York, NY, USA, 2007. ACM.

[GJJW12] Marion Gottschalk, Mirco Josefiok, Jan Jelschen, and Andreas Winter. Removing Energy Code Smells with Reengineering Services. In *GI-Jahrestagung*, pages 441–455, 2012.

[HEKSP11] Timo Hönig, Christopher Eibel, Rüdiger Kapitza, and Wolfgang Schröder-Preikschat. SEEP: Exploiting Symbolic Execution for Energy-aware Programming. In *Proceedings of the 4th Workshop on Power-Aware Computing and Systems*, HotPower '11, pages 4:1–4:5, New York, NY, USA, 2011. ACM.

[HSB12] Hagen Höpfner, Maximilian Schirmer, and Christian Bunse. On Measuring Smartphones' Software Energy Requirements. In *ICSOFT*, pages 165–171, 2012.

[JDNK12] Timo Johann, Markus Dick, Stefan Naumann, and Eva Kern. How to Measure Energy-efficiency of Software: Metrics and Measurement Results. In *Proceedings of the International Workshop on Green and Sustainable Software*, pages 51–54, Zurich, Switzerland, 2012.

[JGJ+12] Jan Jelschen, Marion Gottschalk, Mirco Josefiok, Cosmin Pitu, and Andreas Winter. Towards Applying Reengineering Services to Energy-Efficient Applications. In *Software Maintenance and Reengineering (CSMR), 2012 16th European Conference on*, pages 353–358, March 2012.

[JSW+13] Mirco Josefiok, Marcel Schröder, Andreas Winter, et al. An Energy Abstraction Layer for Mobile Computing Devices. In *2nd Workshop EASED@ BUIS 2013*, page 17, 2013.

[JWH+11] Jung-Sing Jwo, Jing-Yu Wang, Chun-Hao Huang, Shyh-Jon Two, and Hsu-Cheng Hsu. An Energy Consumption Model for Enterprise Applications. In *Proceedings of the IEEE/ACM International Conference on Green Computing and Communications*, pages 216–219, Washington, DC, USA, 2011. IEEE.

[Lei14] Johannes Leimhofer. Predicting the Energy Consumption of Mobile Applications using Simulations. Bachelor's thesis, Technische Universität München, October 2014.

[PLL14] Giuseppe Procaccianti, Patricia Lago, and Grace A Lewis. A Catalogue of Green Architectural Tactics for the Cloud. In *Proceedings of 8th International Symposium on the Maintenance and Evolution of Service-Oriented and Cloud-Based Systems*, pages 29–36. IEEE, 2014.

[RSR+07] S. Rivoire, M.A. Shah, P. Ranganathan, C. Kozyrakis, and J. Meza. Models and Metrics to Enable Energy-Efficiency Optimizations. *Computer*, 40(12):39–48, 2007.

[SNP+09] Lutz Stobbe, Nils Nissen, Marina Proske, Andreas Middendorf, Barbara Schlomann, Michael Friedewald, Peter Georgieff, and Timo Leimbach. Abschätzung des Energiebedarfs der weiteren Entwicklung der Informationsgesellschaft. *Abschlussbericht an das Bundesministerium für Wirtschaft und Technolgogie. Berlin, Karlsruhe*, 12, 2009.

[WBK14] Felix Willnecker, Andreas Brunnert, and Helmut Krcmar. Model-based Energy Consumption Prediction for Mobile Applications. In Jorge Marx Gómez, Michael Sonnenschein, Ute Vogel, Andreas Winter, Barbara Rapp, and Nils Giesen, editors, *Proceedings of Workshop on Energy Aware Software Development (EASED) @ EnviroInfo 2014*, pages 747–752. BIS-Verlag, 2014. ISBN 978-3-8142-2317-9.