EMMA: A Context-Aware Middleware for Energy Management on Mobile Devices

Andre Ebert, Florian Dorfmeister, Marco Maier and Claudia Linnhoff-Popien
Mobile and Distributed Systems Group
Ludwig-Maximilians-University
Munich, Germany
Email: {andre.ebert, florian.dorfmeister, marco.maier, linnhoff}@ifi.lmu.de

***Abstract*—The rapid increase of smartphones' sensing, computation and communication capabilities is accompanied by a growing demand for energy. Both, short-term and long-term energy allocation is a bottleneck which severely constrains a mobile device's capabilities and usability. It is thus one of the most critical challenges for current device development. Despite of numerous hardware improvements, e.g., concerning the energy consumption of sensors and displays, as well as the development of more capable batteries, this issue remains to be solved. Hence, software-based approaches can be used to optimize the energy management of mobile devices according to a user's preferences, context information and the current energetic state of a device.**

In this paper, we specify the requirements for a modular energy management middleware architecture coined EMMA, which considers the dynamic and modular integration of existing energy improvement concepts in relation to the device's current energy status and active services as well as the users context and preferences. Furthermore, we present a prototype application which demonstrates some of EMMA's core concepts.

Keywords–mobile energy management; contextual service provision

I. Introduction

Depending on a user's usage intensity, the batteries of today's smartphones often do not last longer than for one or two days. Reasons for that are high performance hardware components leading to high energy consumption, extensive sensor usage for data acquisition, improperly developed applications, etc.. With this being a known problem, several hardware- and software-based approaches for optimizing the energy consumption of a mobile device have been developed. However, typically, only single components of a mobile system are subject of optimization, while there are no approaches covering all existing subdomains and the corresponding possibilites for saving energy. Disregarding all hardware-based improvements, this leads to the necessity of an integrated, software-based energy management approach in order to protect the limited resources of a mobile device.

Improving a device's energy consumption is not only about extending its batteries' life span, however. Additional goals are the increase of the user's usage experience and the usability of an energy management system. In order to achieve these objectives, individual user preferences, current context information and the device's energetic system state, as well as currently active services must be considered. Furthermore, an energy management system must be able to identify and prioritize services and functionalities which are important or critical to the user. At its best, such a system is able to present these services to the user with a dynamically tailored Quality of Service (QoS) whenever they are requested. The QoS is described by individual parameters for each service. For example, the QoS of a data transfer service could be described by the transfer speed and the amount of data transferred, the one of a positioning service by its accuracy and the time span needed for acquiring a location.

In order to describe the user's context and to be able to react to the present situation, context modeling is required first. According to Dey [1], "Context is any information, that can be used to characterize the situation of an entity. An entity is a person, place or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves." Relying on this definition, it is possible to identify the raw data which is needed for context modeling and to specify sources and procedures for its acquirement. In the following, we present four categories of sources for determining raw context data.

1) **Conventional Sensors:** Data about the users environment, specific activities or current whereabouts can be acquired with built-in smartphone sensors such as Global Positioning System (GPS), gyroscope or accelerometer.
2) **Communication Interfaces:** Network technologies like Global System for Mobile Communications (GSM), Universal Mobile Telecommunications System (UMTS) or Long Term Evolution (LTE) plus Bluetooth and Wireless Local Area Network (WLAN) can also be used for determining the user's location in different granularities [2][3].
3) **Media Data Analysis:** Data like taken pictures or surrounding noises can also be analyzed in order to extract information about the user's situation, e.g., concerning his current mood or a visited location [4].
4) **User Data Mining:** Personal user data like emails, played media files or calendar entries could be valuable in the process of extracting context information. Analyzing it can provide knowledge about whereabouts, important dates like birthdays or upcoming appointments.

However, using some of the presented approaches for the acquisition of context information can be very costly. As will be shown in Section II, all of them consume energy during the process of raw data gathering, as well as for the procedure of extracting information from it. Consequently, the energy consumption caused by context acquisition always has to be compared to the savings enabled through context-aware energy allocation. Besides, there are also privacy issues to be considered. Especially in 3) and 4), sensitive user data, which may need additional protection by privacy policies, is used for analysis.

We are not aware of any concept for an integrated energy management system, which satisfies all of the aforementioned needs. However, there are a lot of ideas for partial improvements of single components. In the following, we specify the requirements for such a system and introduce EMMA, an energy management middleware architecture which considers the dynamic and modular integration of existing energy improvement concepts, which controls all of the device's services and features and monitors the system's energy status, as well as adapting it according to context, currently active services and user preferences.

Section I supplies an introduction to energy management matters on mobile devices. In Section II, we provide a brief analysis of the energy consumption of smartphones in order to understand which components are responsible for draining the battery and where improvements can be achieved. Based on a literature survey on existing approaches for energy management on mobile devices, as well as context-aware mechanisms for optimizing energy consumption, we present the requirements for an extensive context-aware and user-centric mobile energy management architecture in Section III. Eventually, the concept and implementation of our approach coined Energy Management Middleware Architecture (EMMA), will be described in Section IV. After offering some insights into a prototypical implementation in Section V, a conclusion and upcoming future work are addressed in Section VI.

II. Related Work

In order to optimize the energy consumption of a mobile device, one first needs to understand which of its components or applications are draining most of it. Later on, it is possible to contrive ideas and optimization concepts based on this knowledge, as well as on already existing optimization approaches.

A. Individual measuring of energy consumption

There are two different techniques for measuring energy consumption on smartphones. One relies on software based functions provided by the operating system; the other uses an external power meter. Manweiler et al. describe the installation and usage of the Monsoon Power Monitor for external power consumption measurement [5]. However, this approach does not pay attention to the individual consumption of the devices components since it only identifies the overall consumption. Examples for software based and combined measurement concepts are eProf [6], PowerTutor [7] and WattsOn [8]. They all make use of predefined energy profiles describing the energy consumption of specific device components and indicate, that network and sensor usage, CPUs, displays and media playback are the main consumers of energy. Additionally, Pathak et al. [6] examined the energy consumption of different popular applications like Facebook, Angry Birds and others and figured out, that 65-75% of consumed energy is used by third party advertisement modules. Furthermore, they name typical bugs in operation systems, which are responsible for the dissipation of energy.

B. Optimization concepts

In the following, we review different optimization concepts concerning sensors and data transfer. Furthermore, the usage of prediction and data mining in order to prioritize, schedule and optimize tasks on mobile devices is examined.

1) Sensing optimization: There are several approaches for lowering the energy consumption of mobile devices based on optimizing sensor usage. Adaption of QoS parameters is one attempt that can be used to achieve this goal. In particular, the trade-off between service quality and consumed energy is a relevant matter in this context. Besides the adaption of a services performance, there is also the possibility to substitute sensors. In this case, specific sensors are substituted by other technologies which are capable of providing a comparable service while lowering the energy consumption [9][10]. If a complete substitution is not possible or needed, there is also the way of combining sensors [11][12], e.g., triggering; here, the acquisition of data from one sensor is triggered only when a second sensor reaches some kind of previously defined threshold. An example for this is the combination of a smartphones accelerometer and its GPS sensor in the SenseLess concept introduced by Abdesslem et al. [13]. Their approach is to activate the GPS-Sensor only if the accelerometer detects the user being in motion. For evaluation, a user is equipped with two smartphones, one running the standard iOS positioning methods in a 10 seconds interval, the other one working with the SenseLess algorithm. The results showed, that SenseLess needed 85,5% less energy than the standard approach. The determined locations differed from 0,4m to 41m with an average value of 8m. But because the energy consumption was determined on basis of the battery level, there is cause for inaccuracies through wrong interpreted up-and-down-movement, as well as due to the systems general energy consumption. As shown by Priyanta et al. [14], the combination of different sensors is to be used with caution and after individual evaluation. In their case, the computing of determined accelerometer data, which was gathered in a 10 minutes time interval with a frequency of 4Hz to 6Hz, consumed more energy than the location determination via GPS for 5 minutes, determining one location fix each minute. Chon et al. introduce SmartDC, which not only aims at lowering the energy consumption while determining the users location, but also tries to predict the user's future positions and important places [15]. In order to accomplish that, the authors use unsupervised learning techniques, mobility prediction, as well as prediction of system usage based on Markov models. Using this approach, energy savings of up to 81% and an prediction accuracy up to 80% have been reached. The maximum delay time was 160 seconds.

A complete substitution of sensors is used for Ambience-Fingerprinting [16], a technique which is especially suitable for indoor location determination, where no GPS signal can be received. Therefore, raw data of different kind is checked for specific attributes, so called fingerprints. An early concept of Pirantha et al. enables the user to discover his position via ultrasonic and radio fingerprints with an cm-accuracy [16]. Azizyan et al. present SurroundSense, a system for location determination on basis of environmental fingerprints like the spectral composition of noises or visual signatures [17]. The authors also analyze the cumulation of existing fingerprinting techniques, such as motion, noise, acceleration, brightness, color and spectral contents of WLAN signals. In an evaluation with four users in predefined positioning clusters, 93% of all positions could be determined correctly. However, the consumed energy was not compared to the consumption of standard Software Development Kit (SDK) methods.

C. Optimizing data transfer

Due to the extensive energy consumption of data transfer and communication technologies, this field provides a lot of possibilities for achieving energy savings. 2G, 3G and 4G networks each use a different amount of energy for different tasks. Hence, already by choosing the best suited technology for each task, the overall energy consumption can be lowered. For example, 2G networks are suited better for calls than 3G networks, but 3G networks are much more efficient for data transfer [18]. Perucci et al. conclude, that – even though additional power is consumed by the handover process needed for changing networks – significant energy savings can be achieved by consistently switching to a 2G connection for calls [19].

Furthermore, both Balasubramanian et al. [18] and Falaki et al. [20] reveal that not only the overall size of the transferred data is responsible for the consumed energy, but also the size of single transferred packages: Especially, small packages cause a transfer overhead of up to 40%. Moreover, the Transmission Control Protocol (TCP) requires multiple transfers of the same packages due to package loss. To this end, Falaki et al. propose a bigger server side buffer and an optimization of the usage rules for the transfer networks in order to solve these problems, predicting energy savings up to 35%.

Another transfer-related optimization approach is TailEnder, provided by Balasubramanian et al. [18]. TailEnder divides applications into two groups, i.e., applications that tolerate delays in data transfer and those, which can profit from data prefetching. Based on this classification, data is loaded in bundles to minimize the devices staying in the high energy state of the IEEE 802.11 standard. Additionally, TailEnder makes usage of the advantages of the different transfer networks. In comparison to other approaches, 60% more newsfeeds and up to 50% more search results for web requests could be processed while consuming the same amount of energy.

D. History and data-based prediction

The analysis of user preferences and interaction patterns, as well as the usage of data mining techniques on historic or context data can reveal precious insights concerning the energetic regulation of a mobile system. For example, calendar or appointment specific data can be used to predict a user's current and future whereabouts. Predictions about the device's future energy level, combined with the information about upcoming tasks can be used to adapt the energy balance oriented on tasks rated critical by the user [21]. Oliver et al. succeeded in predicting the energy level of mobile devices correctly for a time slot of one hour with an error rate of only 7%. Within 24 hours the error rate was 28%. In order to achieve that, they classified the gathered data of more than 17.300 BlackBerry users and clustered it afterwards [22][23].

Trestian et al. conducted a network-based study which should reveal relations between the user, his movement patterns and used applications in order to predict common interaction patterns. Their results indicate that the usage of specific applications is significantly related to the user's current movements or location [24].

III. Requirements

As shown in Section II-B, there are numerous approaches, which address single optimizations for specific system components or services. None of them tries to wrap all existing concepts in one architecture, which organizes them in a modular way and provides requested services with an adequate output quality. In order to create a holistic energy management system, the following requirements for such an architecture can be identified:

1) **Universal validity and responsibility:** The energy management system is solely responsible for managing all resources of the system and the access to them. It receives all service and resource requests and answers them in an adequate and energy efficient way.
2) **Context and user awareness:** Typical context data like the current time and location, upcoming tasks, individual user preferences or social relations, as well as the systems current energetic state are used to provide services in a customized manner respecting the user's situation and whereabouts.
3) **Modularity:** A holistic energy management system is comprised of a multitude of different components, each responsible for different subtasks. In order to benefit from existing and future ideas in each of these areas, functionality is encapsulated in only loosely coupled modules which can be altered or replaced without affecting the rest of the system.
4) **Scalability and extensibility:** New energy optimization concepts or services can get installed in an easy way, the user is able to use improved modules without non-trivial update processes.
5) **Definition of generic interfaces:** To make the integration of new modules possible, a definition of generic interfaces oriented on the lowest common denominator of installable modules is necessary.
6) **Adaptive data collection:** If data collection is required (e.g., sensor data), the system selects a service which acquires data adaptively to given preconditions. These may be the systems current energy state, the service quality as demanded by the service requester or forecasts concerning needed energy for future tasks. Furthermore, collected data needs to be cached for later usage.
7) **Task prioritization:** In order to take the user's preferences concerning critical tasks into account, the system is able to adhere to task priorities and assign more privileges to higher priority tasks.
8) **Clarity and comprehensibility:** The architecture must be structured in a clear and comprehensible way. This is essential to enable developers to integrate new optimization concepts or module packages into the existing system.

IV. EMMA - An integrated management approach

Based on the requirements identified in Section III, we now present our approach coined *EMMA (Energy Management Middleware Architecture)*. EMMA is a holistic approach for an energy management system on mobile devices, allowing for an easy integration of existing and future energy optimization concepts. It provides its services in an adaptive, context- and preferences-aware manner, controls service parameters and monitors the system's energy state. EMMA runs continuously in the background and is the exclusive interface to the system's resources (such as sensors) and services for other applications. Thus, EMMA is able to coordinate resource demands of all applications and to fulfill their requests in the most energy-efficient way.

A. *Main components*

EMMA consists of two main components, namely, the Control Unit and the Service Provider. These and their sub-components are depicted in Figure 1. In the following, the components are described.

1) Control Unit: The Control Unit is the central component dedicated to respond to service requests, select appropriate services based on context, preferences and demands, and monitor services and the system's energy state. If required, it adapts service parameters to ensure execution of critical tasks. In order to make intelligent decisions, the Control Unit employs information obtained from sub-components such as the Energy Manager, the Schedule Manager and the Service Identificator. Service requests and responses and communication with the Service Provider, as well as among the subcomponents themselves is handled by a dedicated Controller.

The Energy Manager analyzes the energy status of the whole system, calculates energy consumption of individual tasks and tries to predict future energy levels. These results can be combined with information about charging opportunities in vicinity or in near future, to reserve energy for future tasks or to preserve execution of critical tasks by performance adaption for the longest time possible.

The Service Identificator selects the most appropriate service for any given service requests from the whole set of available services – as provided by the Service Provider – based on service parameters and quality of service characteristics. In general, service selection algorithms are geared towards a specific subset of services, since service parameters vary for different kinds of services.

In order to monitor active and upcoming services, the Schedule Manager keeps track of currently running and future tasks, as well as respectively associated information.

2) Service Provider: The Service Provider is a passive component which does not contain any logic or executive power. Its core task is the integration and administration of service modules, as well as the provision of all requested services, the registration of needed callbacks and the caching of sensor data. To achieve that, the Registry Manager scans the folders containing all installed modules and corresponding manifest files in order to identify added or removed modules for installation or removal. Afterwards, the current system state is saved in the Registry Cache, which provides these information for module identification matters to the Control Unit. As soon as a module was identified and released for usage by the Control Unit, the Service Provider creates a new service object and integrates the identified module in order to use its unique functionalities. The service is accessible via specific interface methods (see Chapter IV-C). Furthermore, in order to avoid redundant acquisition of raw data, the Service Provider caches sensor data and provides it if needed.

B. *Modular service concept*

A core feature of EMMA is its modular service concept. A service is defined as any functionality, that a user or an application might request, e.g., data transfer, acquisition of sensors data or media playback. In order to maximize the efficiency of providing these services, EMMA uses a highly modular concept. To this end, for each functionality a

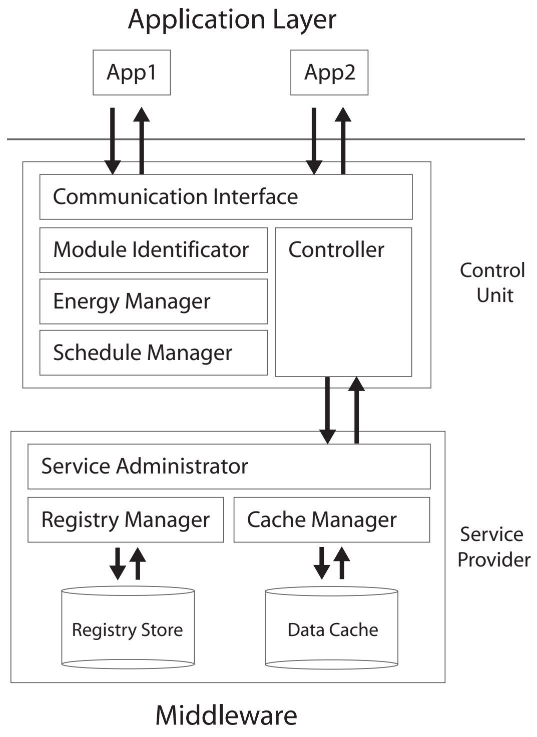


Figure 1. General overview of the EMMA concept and its components.

service class is defined. Each service class might use different technologies to provide a requested service, e.g.,. depending on the demanded service quality and the available system energy, The internal logics of these different technologies are wrapped completely transparent in a service module. The number of possible service modules per service class is unlimited. Later on, after the best suited module for a specific use case was identified, it is integrated into the service object at runtime. Apart from the growing diversity of available technologies, this simple plug-and-play integration is another advantage of the modular concept. It enables optimized logic and improved concepts being installed without a complete system update (see Chapter V).

C. *Communication interfaces*

The EMMA concept defines a fixed set of internal and external interfaces used for communication. Internal interfaces are used for routing information and services, e.g., to realize a working connection between a service object and its integrated service module (see Section V). In contrast, external interfaces are responsible for processing service requests and responding to them. Therefore, data bundles are used to maintain the communication between EMMA and any applications requesting a service. These bundles contain the ID of the requested service class, as well as information about the requester's priority, the expected service runtime and parameters concerning the requested QoS.

The structure of the corresponding response bundles is similar. If a request is answered positively, the reply contains the requested service, as well as information about its actual instantiation, as the returned service may either be a callback function the requester can use, a dedicated service object or simply the requested data. The information about its actual instantiation facilitate correct data unmarshalling.

D. Identification of suitable services

In order to be able to choose the most appropriate service module, EMMA executes an individual identification algorithm for each service class. This is necessary to ensure that the performance of the chosen module matches both, the energy deallocated by the system and the demands of the service requester. The reason for having separate selection algorithms for each class is the different numbers and kinds of parameters, which are necessary to describe a service's energy consumption and QoS-characteristics.

The simplest case of describing a module's characteristics is by only two different performance parameters, which form a Performance Set. One parameter always represents the module's energy consumption, the other one describes a corresponding QoS-specific feature. Each module can possess any number of performance sets describing varying pairs of consumption and service quality related to different modes of operation. In this two dimensional space a target area can be identified between the point-of-origin and the intersection of the values of deallocated energy and requested service quality. In the following, the identification algorithm iterates across all existing modules of the requested service class and checks if any set of their performance parameters is located within the target area. If only one module can be found, this module will be selected. If several modules match the given thresholds, the one with the least energy consumption is selected. Otherwise, if no module can be found, the target area is extended and the same procedure is started again.

The complexity of the individual identification algorithms depends on the number of features necessary for describing a module's performance. According to the current concept state, the mere provision of a service is regarded more important than the meeting of given performance parameters or the approved energy consumption. In case that no comparable module is offering the same service at a lower energy consumption, even a module with a much higher consumption than approved can be provided in order to meet the user's demands.

E. Continuous monitoring

In order to adapt the system's performance and to keep critical services running as long as possible, EMMA continuously monitors the system's energy state. Therefore, not only the current energy consumption but also the predicted consumption for future tasks, which can be determined by analyzing the active and upcoming tasks schedules, combined with prediction techniques (see II-D), is included. Additionally, relevant context information is considered, e.g., loading stations near a user's position.

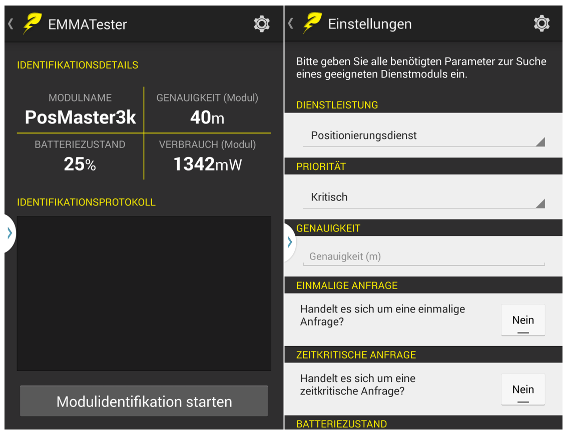
After determining the system's current energetic state and the prediction of the amount of energy needed for upcoming tasks, the system's performance can be adapted in two directions: Given that either more energy than expected is available and the systems performance has been throttled before or a request demands a higher service quality than currently granted, the performance of a service can be increased. In contrast, if there is less energy available than expected or a new high-priority service is started, the performance of active services can be throttled in order to have more power available.

To adapt the performance of a service module, it can be modified within its possible spectrum. If there are more adaptions needed than a specific module can perform, it can be replaced by another of the same service class with different performance parameters. The adaption of service performance commonly results in a change of service quality.

V. Implementation

In order to assess the feasibility of our approach, we developed a prototype application of EMMA with the aid of the Google Android API (target version 18). In its current form, however, the prototype was not yet integrated as an exclusive service manager into the operating system. Instead, due to access and rights restrictions of the Android SDK it has been implemented as part of the application layer. Our goal in this first version was to evaluate the technical feasibility of EMMA's core concepts without analyzing any possible increase in energetic efficiency in detail yet.

In order to cope with EMMA's claim for modularity, the logic of service modules is capsuled in `dex.`-archive files, whose content is described both machine- and human-readable in corresponding `XML` manifest documents. The latter contain all information about a module's performance, nature and energy consumption, and can hence be used to dynamically identify any modules matching a service request. Our prototype is able to respond to incoming requests for either a continuous or a one-time positioning service by selecting the most appropriate service module, taking into account the current energy state of the device, as well as context information and currently active, as well as scheduled services.



IDENTIFIKATIONSDETAILS

MODULNAME: PosMaster3k
GENAUIGKEIT (Modul): 40m
BATTERIEZUSTAND: 25%
VERBRAUCH (Modul): 1342mW

IDENTIFIKATIONSPROTOKOLL

Bitte geben Sie alle benötigten Parameter zur Suche eines geeigneten Dienstmoduls ein.

DIENSTLEISTUNG
Positionierungsdienst

PRIORITÄT
Kritisch

GENAUIGKEIT
Genauigkeit (m)

EINMALIGE ANFRAGE
Handelt es sich um eine einmalige Anfrage? Nein

ZEITKRITISCHE ANFRAGE
Handelt es sich um eine zeitkritische Anfrage? Nein

BATTERIEZUSTAND

Figure 2. Screens of the EMMA prototype application.

To achieve this, at first the device's current energy consumption is determined. In order to keep the implementation overhead for the energy measurements low, energy consumption is approximated by a simple model based on test results obtained from a HTC Desire using the PowerTutor application [7]. By combining these information with knowledge about active and scheduled services as well es their estimated runtime, it is now possible to calculate the energy remaining for new services. In order to identify a suitable service module, the requested QoS parameter contained in a request bundle and the amount of remaining energy are considered for building a target area (see IV-D). After identification, the chosen module is integrated into a singleton service object. The logic contained in the module archive is loaded dynamically into the program code by using the Android-specific reflection-based

DexFile [25] and DexClassLoader-classes [26]. According to the current concept, the identified module is not restricted in its energy consumption once it was installed, but it can be replaced by a more economical one if its consumption is extensive.

In order to be able to use the module's functionalities after its integration, a standard LocationListener-callback provided by the Android SDK is installed in the service object, serialized and returned to the service requester. The latter is now able to listen for positioning updates after rebuilding the callback from the serialized version.

VI. Conclusion and future work

By introducing EMMA, we provide a holistic, modular concept for individual and context sensitive energy management which meets the requirements stated in Section III. The modular concept renders the usage of partial optimization concepts and their energetic savings potential, as well as making their integration a trivial task.

EMMA in its current state is to be considered a basic architecture framework. Lots of details need an initial clarification or further composition, e.g., the specification of different service classes and service modules. Furthermore, a capable security concept must be provided due to the possibility of including foreign code dynamically into the system at runtime. This is necessary in order to avoid the execution of malware and to avoid abuse.

In a following expert's discussion consisting of 8 participants with IT background, we found that the procedure of integrating modules and the conventions for developing them, as well as the interface descriptions for requesting services need to become more facilitated. Furthermore, a pending step is a second implementation of the EMMA concept with the goal to evaluate its actual energy savings potential. Especially, the integration of the architecture as part of the operation system proves to be a challenge, since current platforms like iOS or Android do not allow this without significant interventions in their system's core.

References

[1] A. K. Dey, "Providing architectural support for building context-aware applications," Ph.D. dissertation, Georgia Institute of Technology, 2000.

[2] P. A. Zandbergen, "Accuracy of iphone locations: A comparison of assisted gps, wifi and cellular positioning," Transactions in GIS, vol. 13, no. s1, 2009, pp. 5–25.

[3] B. Li, J. Salter, A. G. Dempster, and C. Rizos, "Indoor positioning techniques based on wireless lan," in LAN, First IEEE International Conference on Wireless Broadband and Ultra Wideband Communications. Citeseer, 2006.

[4] M. Schirmer and H. Höpfner, "Senst*: approaches for reducing the energy consumption of smartphone-based context recognition," in Modeling and Using Context. Springer, 2011, pp. 250–263.

[5] J. Manweiler and R. Roy Choudhury, "Avoiding the rush hours: Wifi energy management via traffic isolation," in Proceedings of the 9th international conference on Mobile systems, applications, and services. ACM, 2011, pp. 253–266.

[6] A. Pathak, Y. C. Hu, and M. Zhang, "Where is the energy spent inside my app?: fine grained energy accounting on smartphones with eprof," in Proceedings of the 7th ACM european conference on Computer Systems. ACM, 2012, pp. 29–42.

[7] L. Zhang, B. Tiwana, Z. Qian, Z. Wang, R. P. Dick, Z. M. Mao, and L. Yang, "Accurate online power estimation and automatic battery behavior based power model generation for smartphones," in Proceedings of the eighth IEEE/ACM/IFIP international conference on Hardware/software codesign and system synthesis. ACM, 2010, pp. 105–114.

[8] R. Mittal, A. Kansal, and R. Chandra, "Empowering developers to estimate app energy consumption," in Proceedings of the 18th annual international conference on Mobile computing and networking. ACM, 2012, pp. 317–328.

[9] J. Paek, J. Kim, and R. Govindan, "Energy-efficient rate-adaptive gps-based positioning for smartphones," in Proceedings of the 8th international conference on Mobile systems, applications, and services. ACM, 2010, pp. 299–314.

[10] Z. Zhuang, K.-H. Kim, and J. P. Singh, "Improving energy efficiency of location sensing on smartphones," in Proceedings of the 8th international conference on Mobile systems, applications, and services. ACM, 2010, pp. 315–330.

[11] I. Constandache, S. Gaonkar, M. Sayler, R. R. Choudhury, and L. Cox, "Enloc: Energy-efficient localization for mobile phones," in INFOCOM 2009, IEEE. IEEE, 2009, pp. 2716–2720.

[12] M. Schirmer and H. Höpfner, "Senst*: approaches for reducing the energy consumption of smartphone-based context recognition," in Modeling and Using Context. Springer, 2011, pp. 250–263.

[13] F. Ben Abdesslem, A. Phillips, and T. Henderson, "Less is more: energy-efficient mobile sensing with senseless," in Proceedings of the 1st ACM workshop on Networking, systems, and applications for mobile handhelds. ACM, 2009, pp. 61–62.

[14] B. Priyantha, D. Lymberopoulos, and J. Liu, "Littlerock: Enabling energy-efficient continuous sensing on mobile phones," Pervasive Computing, IEEE, vol. 10, no. 2, 2011, pp. 12–15.

[15] Y. Chon, E. Talipov, H. Shin, and H. Cha, "Mobility prediction-based smartphone energy optimization for everyday location monitoring," in Proceedings of the 9th ACM conference on embedded networked sensor systems. ACM, 2011, pp. 82–95.

[16] N. B. Priyantha, "The cricket indoor location system," Ph.D. dissertation, Massachusetts Institute of Technology, 2005.

[17] M. Azizyan, I. Constandache, and R. Roy Choudhury, "Surroundsense: mobile phone localization via ambience fingerprinting," in Proceedings of the 15th annual international conference on Mobile computing and networking. ACM, 2009, pp. 261–272.

[18] N. Balasubramanian, A. Balasubramanian, and A. Venkataramani, "Energy consumption in mobile phones: a measurement study and implications for network applications," in Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference. ACM, 2009, pp. 280–293.

[19] G. P. Perrucci, F. H. Fitzek, G. Sasso, W. Kellerer, and J. Widmer, "On the impact of 2g and 3g network usage for mobile phones' battery life," in Wireless Conference, 2009. EW 2009. European. IEEE, 2009, pp. 255–259.

[20] H. Falaki, D. Lymberopoulos, R. Mahajan, S. Kandula, and D. Estrin, "A first look at traffic on smartphones," in Proceedings of the 10th ACM SIGCOMM conference on Internet measurement. ACM, 2010, pp. 281–287.

[21] N. Ravi, J. Scott, L. Han, and L. Iftode, "Context-aware battery management for mobile phones," in Pervasive Computing and Communications, 2008. PerCom 2008. Sixth Annual IEEE International Conference on. IEEE, 2008, pp. 224–233.

[22] E. Oliver and S. Keshav, "Data driven smartphone energy level prediction," University of Waterloo Technical Report, 2010.

[23] E. Oliver, "Diversity in smartphone energy consumption," in Proceedings of the 2010 ACM workshop on Wireless of the students, by the students, for the students. ACM, 2010, pp. 25–28.

[24] D. Willkomm, S. Machiraju, J. Bolot, and A. Wolisz, "Primary user behavior in cellular networks and implications for dynamic spectrum access," Communications Magazine, IEEE, vol. 47, no. 3, 2009, pp. 88–95.

[25] "DexFile - Android Developers," http://developer.android.com/reference/dalvik/system/DexFile.html, 2014, [Online; accessed 25-July-2014].

[26] "DexClassLoader - Android Developers," http://developer.android.com/reference/dalvik/system/DexClassLoader.html, 2014, [Online; accessed 25-July-2014].