Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 1347967, 15 pages
https://doi.org/10.1155/2018/1347967

Research Article

Power Profiling of Context Aware Systems: A Contemporary Analysis and Framework for Power Conservation

Umar Mahmud,[1] **Shariq Hussain**,[1] **and Shunkun Yang**[2]

[1]*Department of Software Engineering, Foundation University Islamabad, Pakistan*
[2]*School of Reliability and Systems Engineering, Beihang University, Beijing, China*

Correspondence should be addressed to Shunkun Yang; ysk@buaa.edu.cn

Received 14 March 2018; Revised 18 July 2018; Accepted 29 August 2018; Published 16 September 2018

Academic Editor: Mubashir H. Rehmani

With the advent of smart, inexpensive devices and a highly connected world, a need for smart service discovery, delivery, and adaptation has appeared. This interconnection is composed of sensors within devices or placed externally in the surrounding environment. Our research addresses this need through a context aware system, which adapts to the users' context. Given that the devices are mobile and battery operated, the main challenge in a context awareness approach is power conservation. The devices are composed of small sensors that consume power in the order of a few mW. However, their consumptions increase manifold during data processing. There is a need to conserve power while delivering the requisite functionality of the context aware system. Therefore, this feature is termed as 'power awareness.' In this paper, we describe different power awareness techniques and compare them in terms of their conservation effectiveness. In addition, based on the investigations and comparison of the results, a power aware framework is proposed for a context aware system.

1. Introduction

Context awareness is termed as the awareness of a system to external and internal stimuli gathered through internal and external sensors. Such a system classifies this as an activity to adapt the services present within an environment. A context aware system is typically found in a handheld device, such as a smart phone [1–3]. Context is simply the description of the current situation as bound by the environment, while context awareness utilizes the context to adapt and change the services present in the environment. The environment becomes smart and delivers better services to the end-user [4]. A simple example is the change in orientation of a handheld device from portrait to landscape, based on how a person holds the phone.

Among various issues and challenges of context awareness is the management of battery power [5]. Battery power is used by different sensors that generate data. For instance, a device orientation sensor generates orientation data, whereas an accelerometer generates acceleration data. This data is then processed by the context aware system to classify the current context or situation. These sensors typically consume low power in the order of 1-2 mW. However, they consume 180–300 mW when processing is carried out [6, 7]. This significant increase in consumption requires sufficient power conservation such that the functionality or effectiveness of a context aware system is not reduced. The term 'power conservation' is interchangeably used as 'energy conservation' in the literature. This paper uses the former terminology. Power is related to energy, wherein the use of power over a period of time is considered energy consumption. The reduction in the use of energy for the same task is considered as power conservation.

This paper presents a literature review of different techniques of power conservation that have been proposed by researchers over the years. The literature, in general, considers power and energy to be synonymous. However, this paper follows the convention of power conservation. Furthermore, we discuss the effectiveness of these techniques and present a framework for power conservation. We describe two

approaches that can be used to achieve substantial power conservation in context aware systems: (a) network communication-based conservation and (b) context processing-based conservation.

The rest of the paper is organized as follows: Section 2 discusses the related work. Section 3 gives an introduction to context awareness. Section 4 presents an overview of power consumption profiling. In Section 5, first we provide a brief discussion on power awareness and then compare the contemporary power conservation techniques in terms of their capabilities and effectiveness. Section 6 describes the proposed framework for power conservation applicable in context aware systems. Finally, Section 7 provides the concluding remarks.

2. Related Work

A number of researchers have investigated various approaches to model, analyze, and evaluate the efficiency of power consumption and have proposed diverse conservation techniques. This section reviews several closely-related investigations and provides a survey of various power optimization methods, as summarized in Table 1. The available literature can be categorized based on the focus areas. These areas are handheld mobile and cloud-based power optimization, application power optimization, and network communication power optimization.

The device power optimization method considers the device and provides means to conserve battery power within the device. Naik [8] claimed that the size and weight of the handheld mobile devices are the key constraining factors. This opens avenues for software based energy efficiency. The author provided a survey of different software based techniques, including sleep mode support, energy efficient GUI design, and power efficient communication methods. However, this survey neither provides solutions at the device level nor considers context aware systems. Ravindarnath [9] and Arun [10] have independently provided similar surveys for mobile and cloud computing based devices. Li et al. [11] has provided a survey of cloud power optimization based on the tasks given to a cloud system. Mittal [12] has surveyed various power conservation techniques for embedded systems. These embedded systems conserve power as a whole.

Jofri et al. [13] surveyed energy-aware profiling approaches for mobile devices and placed them into five classes and discussed their interrelationships. Power conservation of applications is one of them. Bernal et al. [14] argued that developing a power efficient context aware system is challenging, as it depends on the context sensor configuration. The authors have used an energy profiler to measure the power consumption of context aware applications and have provided recommendations for sensor configuration. Rault et al. [15] have published a survey of energy efficiency in wearable sensors. The authors argued that continuous sampling and communication deplete the battery power fast. They surveyed the existing energy efficient approaches in context aware activity recognition for healthcare applications. Ismail et al. [16] presented a survey of multimedia content adaptation techniques with respect to power consumption. The amount of power utilized in a multimedia content view is high, which can be adjusted based on the device's battery power. The survey neither considered context awareness nor discussed application level fine-tuning for power conservation. Wang et al. [17] argued that energy can be managed efficiently both at the hardware level and at the software level. The authors have selected DBMS for energy profiling and subsequently recommend various conservation measures.

The noisy nature of mobile networks has led to a high network power consumption. This provides a basis for surveys for power optimization in network communication. Brienza et al. [18] have outlined a power efficiency survey for P2P systems. This survey is focused on file distribution and content streaming features of P2P applications. Bolla et al. [19] published a survey of power efficiency in networks. The leading source of power consumption in mobile devices is network communication. However, the authors did not consider the energy overhead involved in mobile processing. Celenlioglu et al. [20] have published a survey of vertical handoff in 3G and WLAN devices.

3. Context Awareness

The objective of context awareness is to provide better services and applications to the users [21–24]. This includes discovering services among available options within an environment as well as fine-tuning the selected services. Thus, the context describes a complete situation wherein multiple users interact with different services bounded by a common environment [25–30]. The context process has five phases as shown in Figure 1. The phases are context aggregation, context representation, context history, context inference, and service adaptation [26].

3.1. Context Aggregation Module. Initially, the context is gathered from the sensors present in the environment and on board of the devices. Each sensor provides context values that are then aggregated as the contextual data captured at an instant. The contextual data gathered is raw and may need further processing to convert it into a meaningful form [31, 32]. Traditionally, this process is concerned with invoking the external sensors present in the environment as well as collecting data from the internal sensors present on the device. An associated controlled vocabulary is also provided to identify the same context attributes but with different names.

3.2. Context Representation Module. The contextual data is then stored as per the data organization of the context aware system, which is referred to as the current context [33]. Among various mechanisms, ontology is the preferred representation technique [34]. Ontology is based on eXtensible Markup Language (XML). It has the added advantage of providing a heterogeneous method of storage so that a complete context can be shared across diverse platforms.

3.3. Context History Module. The task of this module is to maintain the history of contexts associated with the appropriate labels. The history is generally maintained as records in an SQL table, which can be converted to No-SQL using

Table 1: Surveys for power optimization.

Ref	Contents	Objects	Focused Techniques
[8]	Device power optimization	Handheld mobile devices	Software based energy efficiency
[9]	Enhancing energy efficient	Mobile and cloud computing based devices	Energy aware offloading techniques
[10]	Minimizing energy consumption	Mobile cloud computing	Effective task scheduling method
[11]	Cloud power optimization	Cloud applications	Energy consumption models
[12]	Power conservation	Embedded systems	Power management techniques
[13]	Energy efficiency	Mobile devices	Energy-aware profilers
[14]	Power consumption	Context aware applications	Energy profiler for sensor configuration
[15]	Energy efficiency	Wearable sensors/healthcare application	Approaches for context aware activity recognition
[16]	Power consumption	Multimedia	Content adaptation techniques
[17]	Energy profiling	Software and hardware level	DBMS
[18]	Power consumption and optimization	P2P/Network communication	File distribution and content streaming
[19]	Power efficiency	Networks/mobile devices	Power consumption in network communication
[20]	Energy efficiency	3G and WLAN devices	Vertical handoff algorithm

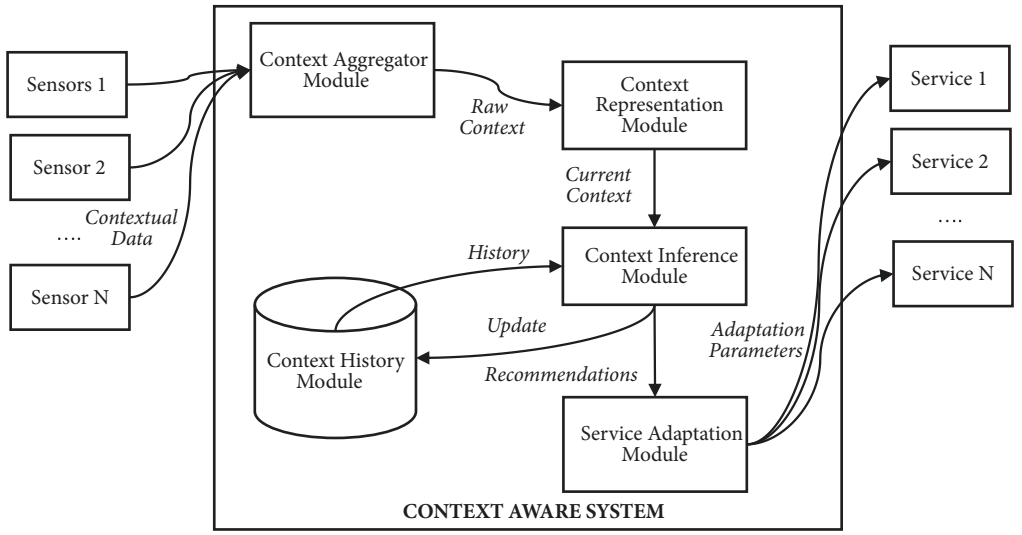

Figure 1: System architectural of a context aware system.

XML. This history information can be purged to reduce the size as well as to identify the user preferences [35].

3.4. Context Inference Module. Through this module, the activity or situation is classified through a context inference mechanism, and labels are given to current contexts [36]. This can include multiple overlapping activities within an environment as well multiple role-based interactions by each user. Here, recommendations for better service discovery and delivery as well as adaptation are made. The context inference mechanism can be designed as rule based or classification-based [37]. The software engineer is thus concerned with selecting and improving known classification algorithms to achieve context awareness.

3.5. Service Adaptation Module. Finally, by this module, the services are contacted and adapted as per the recommendations of the inference phase. This has two aspects: (a) service discovery and delivery: this is concerned with finding and selecting better services for the user and (b) service adaptation: this deals with changing the parameters of a selected service as per the current context. The service could be an external web service that can be invoked upon the change in the state of the current context. The selection of web service may employ certain criteria for QoS-based web service selection [38, 39].

4. Power Consumption Profiling

Power conservation optimization is a challenging area that requires novel approaches to solve the underlying issues. Power is a vital resource for battery-operated systems. Recent technological advancements have paved the way for miniaturization in terms of size and weight of such systems. However, power consumption remains to be a dominant issue in the design process. Both aspects of a system, i.e., software and hardware, are to be considered in evaluating the true power consumption. Typically, the hardware part is considered to be more resource hungry. Very little attention is

Table 2: Comparison of power profiling techniques.

Power profiling		Advantage	Disadvantage	Requirements/suitability
Power consumption profiling methods	Model-based	Easy to implement	Affected by model's accuracy	Suitable at design times
	Real-measurement based	Real time prediction for dynamic power consumption	High overhead for tracking system events	Suitable for process level
Power consumption profiling finesse	Component level	Produce more accurate results	Requires detailed specifications	Suitable for device vendors at design times
	Device level	Easy to implement	Appropriate way	Usage patterns need to be identified

usually given on software development where the operating algorithms and programming styles may influence the overall power consumption of the system [40].

Apart from the hardware, limited considerations have been given to optimization of software applications [41–43]. For power optimization, it is necessary to correctly assess the power usage of the software components of a system. A number of approaches have been presented in the literature that focus on profiling the techniques in order to optimize the power consumption of the software applications [42–47]. Abkenar et al. [48] have employed a mechanism to effectively measure the consumption during a group activity recognition by monitoring the battery drain of a mobile device.

Thermal aspects are also of high interest while profiling the power consumption, as these can offer useful insights. Marcu et al. [49] used thermal profiling and defined benchmarks to identify the impact of the CPU work load (i.e., the executed applications) using the thermal response of the CPU cores. However, this approach is not generic enough, as it is only applicable to a specific test-bed. Profiling is not a trivial task as it involves identifying miniature-level functional relationships and associated power consumption among the complex processes. Duan et al. [50] proposed a power estimation model for mobile devices that considers the interconnected relationship among the three layers, i.e., the hardware, the operating system, and the application under execution. To get the maximum device uptime, optimum resource utilization, in terms of battery energy, is crucial. Hence, true power profiling is the only critical success factor in accurate lifetime prediction.

Prior to understanding energy profiling, the power consumption states used by the devices for power management need to be determined. A brief overview of the power consumption states is presented in the following section, and different power profiling techniques are compared in Table 2.

4.1. Power Consumption States. Generally, three states are available in modern devices to support power conservation: (a) 'active' or 'working,' (b) 'idle' or 'low power,' and (c) 'suspend' or 'sleep.' These states correspond to the device, and do not scale-down to the processes or applications running on the device. These states are shown in Figure 2. Their descriptions are given below:

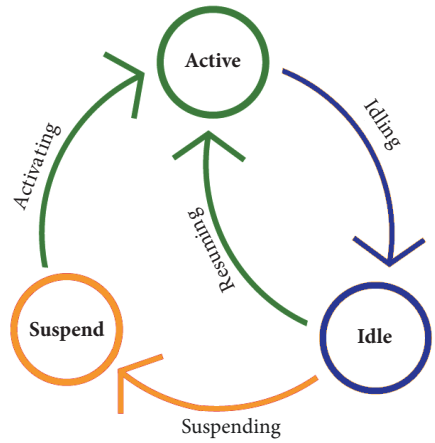

Figure 2: Power consumption states.

(i) **Active (Working):** In this state, the device is fully functional and is processing the user's tasks. Power consumption in this state depends on the usage scenario and involvement of various components that are associated with the task. These components can be the internal sensors present on the device.

(ii) **Idle (Low Power):** When there are no running tasks, the device is switched to idle mode and the core components as well as the associated peripheral components are switched to minimum energy levels. Only the basic functionality of a mobile device is available, e.g., sending and receiving messages or phone call.

(iii) **Suspend (Sleep):** In suspend mode, all unused components are switched-off to save the power. Only the core components are kept alive at predefined minimum energy levels. Only a minimal amount of activity is allowed with which the device can perform the essential tasks.

4.2. Power Consumption Profiling Methods. Generally two methods are considered for power consumption profiling: (a) model-based and (b) real measurement-based [51]. These methods are discussed below:

(i) **Model-Based:** In model-based method, a model is built to analyze the power consumption of the events in the system. Then, through a series of measurements, the power consumption for each of the system's component is determined, and the model is calibrated based on the measured data. Thus, the power consumption for the whole system can be evaluated by monitoring the rendering of the events through profiling. This method is easy to implement. However, the model's accuracy is a key factor in a true evaluation of the power consumption.

(ii) **Real Measurement-Based:** In a real measurement-based method, the power consumption of the system is measured in real-time. The power consumption profile of the system is generated after harmonizing the power data with the system events. This scheme is useful to predict the dynamic power consumption of the system. However, it requires a higher degree of synchronization between both entities, i.e., power records and event records. Another concern while tracking the system events is that it creates a high overhead. Due to this reason, most of the power consumption profiling techniques, as proposed by the researchers to date, focus on the process-level power consumption.

4.3. Power Consumption Profiling Finesse. The finesse of power consumption profiling is broadly based on either coarse-grained or fine-grained methods. The coarse-grained method considers the overall device, while the fine-grained method considers only the processes on the device. The power consumption profiling methods can be applied either at the component level or the device level [52].

(i) **Component Level method:** Under this method, the power consumption for each component of the system is measured independently. By summing up the power consumption values for different components, the total power consumption of the system can be measured. This method produces more accurate results. However, it requires detailed specifications of both the hardware and the software for a particular component. The device vendors can benefit from this approach, as they have detailed specifications and can possibly implement this at the design stage.

(ii) **Device Level method:** This method, to be more appropriate, can be termed as the device usage scenario, as it measures the power consumption of a device for different use-cases [53]. This method is easy to implement, and is widely used in power consumption profiling. For this method, two considerations need to be considered. Firstly, the number of required test runs need to be optimized to achieve coherence in results. Secondly, the usage patterns need to be identified in order to synchronize the power consumption with the use-cases. An example of possible usage scenarios in mobile devices where a user can use any number of applications concurrently is shown in Figure 3.

5. Power Awareness

Among the issues within context awareness, power conservation is of prime importance. Since a context aware system can be mobile, and it relies on the internal sensors present on the device as well as the external sensors present in the environment, there is a need to conserve power. Furthermore, the context recognition is itself a power-consuming task. Once the context is gathered, it must be processed, and processing consumes power [54].

Depending on the classification approach, the algorithms employed have different power profiles. For example, some algorithms, such as artificial neural network (ANN) and genetic algorithm (GA), train prior to classification and have higher training times. Algorithms, such as K-Means and K nearest neighbors (KNN), have zero training but maximum classification times [55]. The design of algorithms allows the scientists to calculate the time and space complexity only. Currently, no convention exists to measure the power conservation of an algorithm. Context aware applications consume power in three phases: context aggregation, context processing, and context adaptation.

Power awareness is a mechanism through which a context aware system can optimize itself to conserve power while providing acceptable functionality. Researchers have proposed various techniques to effectively conserve the power of a context processing mobile device as discussed in this section. Most of these recommended techniques are based on the power consumption profile of a mobile device. Without power consumption, the power conservation technique is useless.

5.1. Contemporary Power Conservation Techniques. In the following sections, various techniques to monitor the power are discussed. These techniques are primarily based on investigations performed by different researchers.

5.1.1. Threshold-Based Conservation. This technique is a simple rule where a threshold is preset for power conservation, e.g., the activation of power saving mode at 15% battery power. This technique, on a whole, is applied to mobile devices. In the power saving mode, the data connections are cut-off. This results in longer battery life—at the cost of context awareness.

Threshold-based conservation, though being the simplest, has the lowest intelligence as the threshold is preset in factory configurations. This mechanism does not adjust the power conservation to the application and is used at the device level.

5.1.2. Demand-Based Conservation. An improvement over the threshold-based technique is to adaptively adjust the threshold based on the demand. The system must now proactively predict the demand and conserve when the demand is low. This is possible at the operating system (OS) or at the device level, where the system predicts the overall demand and adjusts the threshold. The power awareness at the OS level can be termed as 'battery-aware' as it considers the

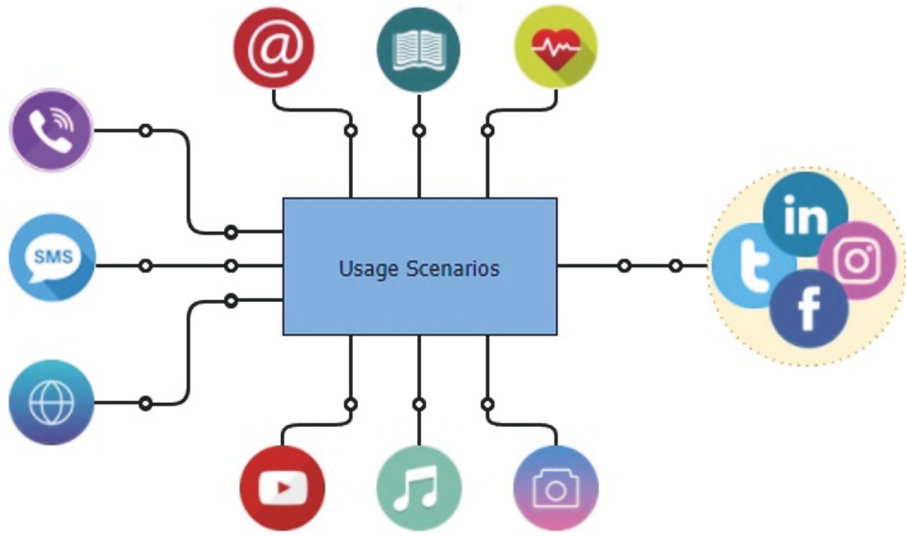

Figure 3: Usage scenarios for power consumption at the device level.

discharge management by the complete system rather than the applications [56]. This technique is implemented as a rule based system (RBS) using mapping and is limited to fine-tuning at the application level [57, 58].

Elmalaki et al. have pointed out that using the OS layers for sensor communication as well as context processing would consume less power as the layers (and subsequently the number of lines in the code) are reduced [59]. The researchers further placed the context awareness layers within the Android framework of a smart phone. The operating system must perform power consumption profiling on all applications, i.e., at the device level.

Moghimi et al. approached the problem by creating a Fuzzy RBS based power manager [60]. The power manager is responsible for inferring the rate of sensor access. For example, if the smart phone is idle then the update rate could be reduced to an hourly basis.

Fei et al. have proposed the use of software based dynamic power management as a trade-off between power conservation and an application's Quality of Service (QoS) parameters [61]. They have identified the power requirement to be a measure of the quality of applications.

A demand-based conservation technique cuts off all processing when the threshold is reached just like the threshold-based conservation. Therefore, it lacks fine-tuning at the application level. The implementation as a rule based system, though simple, has its challenges due to increases in the rule space. The system must measure the current conservation through software-hardware interfaces, which may be propriety [62].

5.1.3. Cloud-Based Conservation. Some researchers have proposed the use of cloud as an alternative to using local architecture for context awareness [6, 63]. This mechanism effectively conserves power by not using the local system for processing and awareness-oriented tasks. A broker is implemented as a centralized middleware, which performs context processing over the cloud if the threshold is reached. However, energy profiling for cloud environments is another challenge. A number of researchers have proposed various approaches to address this issue [64, 65]. In our work, we consider power profiling for mobile devices only and leave the cloud environment for future work.

This technique does not use the local resources for context processing. Thus, it conserves power by utilizing the cloud. The challenge is to adaptively establish a threshold after which the computations are sent over the cloud. In effect, this technique is a load balancer. The power consumption profile has higher power consumption cost for cloud communication as compared to the power consumption cost of context processing. The mobile devices only use the internal sensors and communicate with the external sensors present in the environment as well as the cloud.

5.1.4. Context Change-Detection Based Conservation. The context is classified by continuous monitoring of the contextual data in a context aware system. This data is acquired from internal and external sensors as discussed in Section 3. Kang et al. have proposed proactive monitoring on the basis of change detection [66]. The sensors are continuously monitored but the context processing is suspended until a change is detected. The context processing is now carried out proactively if the change in sensor data results in change of context.

This technique though fine-tunes conservation at the application level requires a method to predict the change in the context a priori. An associated history manager must be provided in the context aware module of the complete system.

5.1.5. Statistics-Based Conservation. Sathan et al. have proposed the use of statistics to predict when to poll sensors [67]. The effect of continuous sensing is a constant drain on the battery. If the change in sensor can be predicted, the sensors can be invoked in a noncontinuous manner. The context processing time is also reduced. An innovation is to provide

a sensor layer between the sensor and the context gathering module. The sensor layer could easily predict the sensor change based on the history of accesses.

Yuryur proposed a lightweight online and unsupervised mechanism to classify the context using the accelerometer data [7]. To conserve power, a Markov reward process is used to profile the power consumption. Yuryur has identified that a 1 mW accelerometer consumes 370 mW power when the context is being processed. The reward process monitors the power consumption as triggered by a change in the sensor data.

This technique requires a priori information to effectively predict when to poll the sensors, and when to process the context if continuous data is provided from the sensors. If the classification of the context is carried out as a predictive algorithm, it would support the power awareness task. Otherwise the system would use two different learning algorithms, i.e., one for classification of the context and other for sensor polling [35, 37].

5.1.6. Selective Sensor Access-Based Conservation. The context classification tasks require data from the sensors present in the environment as discussed in Section 3. The sensors correspond to variables, which are provided to the context aware system. Hermann et al. argued that the classification of all context situations does not require the use of all variables. For instance, to identify the activity being performed in a cafeteria might require the location and time values but may not need the outside barometric pressure value for correct classification [68]. The researchers propose the use of selective sensors for the classification purposes. If an accurate classification cannot be achieved, then further sensors can also be invoked. The context aware system then starts, suspends, and changes the sampling rate of the sensors according to the user's known context.

This technique demands a minimum list of sensors that must be used for classification of the context. The number of sensors, both internal and external, may be large. In addition, not all of them need to be invoked for a target context. The list must be made adaptive to include unknown contexts having different minimal list of sensors.

5.1.7. Source Code Optimization-Based Conservation. In addition to active power conservation techniques presented in above sections, researchers have suggested that source code optimization is another key approach for power conservation. This technique is based on the concept of power aware software development and considers both the programming languages as well as the associated compilers.

Compiler-based optimizations were given little considerations as it has meagre effect in power consumption optimization on the software. Several methods can be identified from the literature that target power conservation by fine-tuning the compilers in terms of instruction rearrangements, optimized use of memory banks, and reduction of the operands in an instruction to a minimum [69–72].

Apart from compiler optimizations, methods such as loop unrolling, software pipelining, recursion elimination, and algorithm evaluation may also prove to be power aware. However, their effectiveness need to be evaluated under a given problem-set [73]. Further, power conservation for the software depends on coding styles, selection of language constructs, choice of algorithms, and selection of data types. Capra et al. compared a number of open source software for power consumption in terms of design and functional aspects and also examined the effects of the application development environments' usage on power conservation [74].

Hao et al. proposed a technique for energy consumption in mobile devices based on instruction-level energy modeling. Many researchers have proposed models for energy consumption either for some specific components, or for all components of a mobile device [75].

Tsao et al. focused on I/O functions in mobile devices and analyzed the energy consumption of I/O events [76]. All these factors exhibit an association with the source code-based power conservation techniques.

Source code optimization-based conservation is carried out during compile time. Developers create less lines of code that yield the same outcome allowing less power consumption during processing. This technique is cumbersome as the development effort is increased many folds.

5.2. Comparison of Contemporary Techniques. In order to compare the representative contemporary techniques comprehensively, we develop a framework that consists of the following attributes:

(i) **Intelligence:** A technique exhibits intelligence if it has a learning mechanism or at least is better than a simple threshold-based systems. The comparison does not consider the algorithm used for power awareness.

(ii) **Adaptive threshold:** This refers to establishing a dynamic threshold that can be changed as opposed to factory configurations. This requires a better intelligence to adaptively select when to change the power state of a device. Adaptive threshold is a smarter version of the threshold-based conservation technique.

(iii) **Network communication power conservation:** The context aware system communicates with external sensors present in the environment mostly through wireless links. Wireless links being noisy lead to multiple retries for communication, and thus power is wasted. A technique heavily dependent on the network needs to conserve power for network communication.

(iv) **On-device context Processing:** Whether the context classification or context processing is carried out on a mobile device or some other middleware is used. A cloud-based technique uses the external servers while all other methods utilize the device for context processing.

(v) **Cut-off all applications:** This refers to a device level conservation method, and does not adapt to application level adjustments. Cutting off all applications effectively halts the context aware processing as well as all other processing in favor of the basic or minimal tasks on the device.

(vi) **Rule based system:** This method refers to using simple rules for power conservation where rules are prewritten in the applications as well as in the device. A rule based system is efficient in classification, but it is resource-heavy due to the need for large rule space. The designers have suggested rule based systems for small rule spaces.

(vii) **Load balancing:** This is primarily used where external processing is utilized as in the case of cloud-based conservation. The system thus balances the load of context processing on the device as well as on the external middleware.

(viii) **Use of history:** History is used for predictive conservation as well as to identify a priori information. A history module is part of the context aware system as discussed in Section 3. Using history information for power awareness is considered in this facet.

Based on the aforementioned attributes, a comparative matrix for contemporary power conservation techniques is presented in Table 3. The goal of the comparison is to identify supportive and non-supportive criterion for each technique.

A detailed comparison is presented in Table 4, where factors, such as advantages, disadvantages, requirements/suitability, and technical aspects, are included. We hope the comparison will provide useful insights for each conservation technique.

6. Framework for Power Awareness in Context Aware Systems

In this section, we present the proposed framework for power awareness and describe its characteristics. The motivation behind building such a framework is to provide power optimization in context aware systems in a power aware fashion. The proposed framework is illustrated in Figure 4. Awareness is achieved once the power consumption is modeled. The power conservation can then be enhanced. Power conservation can be achieved in processing, storage, and communication phases that comprise a context aware system.

6.1. Power Consumption Model. This section discusses how power is consumed and utilized in every module of a context aware system. This will allow us to better understand the power requirements and to propose conservation tactics at different levels.

6.1.1. Power Consumption in Context Aggregation Module. The context aggregation module is responsible for communicating with internal sensors as well as external sensors present in the environment as discussed in Section 3. The power consumption of external and internal sensors is of the order of a few mW. However, the processing consumes much more power [6]. For external sensors, the power is also consumed during network communication. Being wireless nodes, the number of retries due to noisy channels and dense networks is large. Hence, a number of retries in communication can increase the power consumption many fold. This module does not perform any processing, and simply organizes the current context as an attribute-value tuple.

Denoting the power consumption by the sensors as p_s, the power consumption by the network as p_{nw}, the number of retries as t, and the number of sensors as n, the power consumption of context aggregation module P_{cam} is given by (1). For all internal sensors, the power consumption by the network is zero.

$$P_{cam} = \sum_{i=1}^{n} \left(p_s + \left(p_{nw} \times t\right)\right) \tag{1}$$

6.1.2. Power Consumption in Context Representation Module. The raw current context as acquired by context aggregation module is then processed and represented in a hierarchical and platform independent mechanism [53]. This also involves some processing to convert raw context to a meaningful form. As an example, the temperature in Celsius is to be listed as 'Warm' or 'Cold.'

Denoting the power consumed in converting a single value to the appropriate form as p_{cnv} and the power consumed in processing low level task as p_{lt}, there are n sensors and the power consumed by the representation module p_{crm} is given by

$$P_{crm} = \sum_{i=1}^{n} \left(p_{cnv} + p_{lt}\right) \tag{2}$$

6.1.3. Power Consumption in Context History Module. The current context and the previously known states associated with the classification state are kept in a data store. Thus, the consumed power information is accessed into the history. For the systems where predictive or statistical power awareness is not carried out and the history is absent, the power consumed is thus zero.

Denoting the power consumed in a single record access of history as p_{accrec} and the number of records as r, the power consumed by the context history module P_{chm} is given by

$$P_{chm} = \sum_{i=1}^{r} p_{accrec} \tag{3}$$

6.1.4. Power Consumption in Context Inference Module. The main task of context inference module is to process the current context and classify it as a context situation or activity. Being the core of a context aware system, maximum amount of processing is carried out in this module. Therefore, maximum power is also consumed by this module. The power consumed in this module is primarily dependent on the classification technique. The techniques have two general steps: (a) training and (b) classification. Some techniques, such as ANN and GA, have higher training times but are quick in classifying, whereas techniques such as KNN have low training time and high classification time. These general steps indicate the consumption of power during training as well as during classification phases. However, for a given technique only one step will be dominant.

Denoting the power consumed during training as p_{trg} and the power consumed during classification as $p_{classify}$, the

Table 3: Comparative matrix for power conservation techniques.

	Threshold-Based Conservation	Demand-Based Conservation	Cloud-Based Conservation	Context Change Detection-Based Conservation	Statistics-Based Conservation	Selective Sensor Access-Based Conservation	Source Code Optimization-Based Conservation
Intelligence	×	×	√	√	√	√	√
Adaptive Threshold	×	√	√	√	√	√	√
Network Energy Conservation	√	√	×	√	×	√	×
On Device Context Processing	√	√	×	√	√	√	√
Cut-off All Applications	√	√	×	×	×	×	×
Rule Based System	√	√	×	×	×	×	×
Load Balancing	×	×	√	×	×	×	×
Use of History	×	×	×	√	√	×	×

Table 4: Comparative overview of contemporary power conservation techniques.

Conservation Techniques	Advantages	Disadvantages	Requirements/Suitability	Technical Aspects
Threshold-Based Conservation	(i) Simple and easy to implement	(i) Lack of adaptability (ii) Device level conservation	(i) Battery operated devices that need to shut down in a stable manner	(i) Non-adaptive and hard threshold used for stable power-on and power-off procedures in battery operated devices
Demand-Based Conservation	(i) Adaptive threshold	(i) Device level conservation	(i) Battery operated devices that need to predict consumption based on current usage	(i) Rule based system (ii) Predictive analysis
Cloud-Based Conservation	(i) Use of cloud (ii) Adaptive threshold (iii) Reduction of power awareness overhead at device level (iv) Power conservation at application level	(i) High communication overhead	(i) Suitable for cloud assisted environment	(i) Cloud-based system (ii) Can allow classification techniques for adaptive thresholding
Context Change Detection-Based Conservation	(i) Adaptive threshold (ii) Context monitoring (iii) Proactive approach (iv) Power conservation at application level	(i) Context change detection is required a-priori	(i) Context aware systems	(i) Context awareness (ii) Context differential
Statistics-Based Conservation	(i) Adaptive threshold (ii) Predictive conservation (iii) Power conservation at application level	(i) History management overhead (ii) Statistical calculation overhead	(i) Low power consumption history management module	(i) Predictive analysis (ii) Regression
Selective Sensor Access-Based Conservation	(i) Selective sensor approach (ii) Power conservation at sensor level	(i) Minimal list of sensors for each activity classification (ii) Limited context classification scenarios	(i) Suitable for large number of sensors present in an environment	(i) Power conservation for energy hungry sensors
Source Code Optimization-Based Conservation	(i) Source code optimization keeping in view power constraint	(i) Cumbersome to implement (ii) Compile time conservation (iii) Non-adaptive thresholding	(i) Suitable for embedded systems (ii) Non-generic devices and applications	(i) Source code optimization (ii) Power consumption measurement for programming constructs and procedures

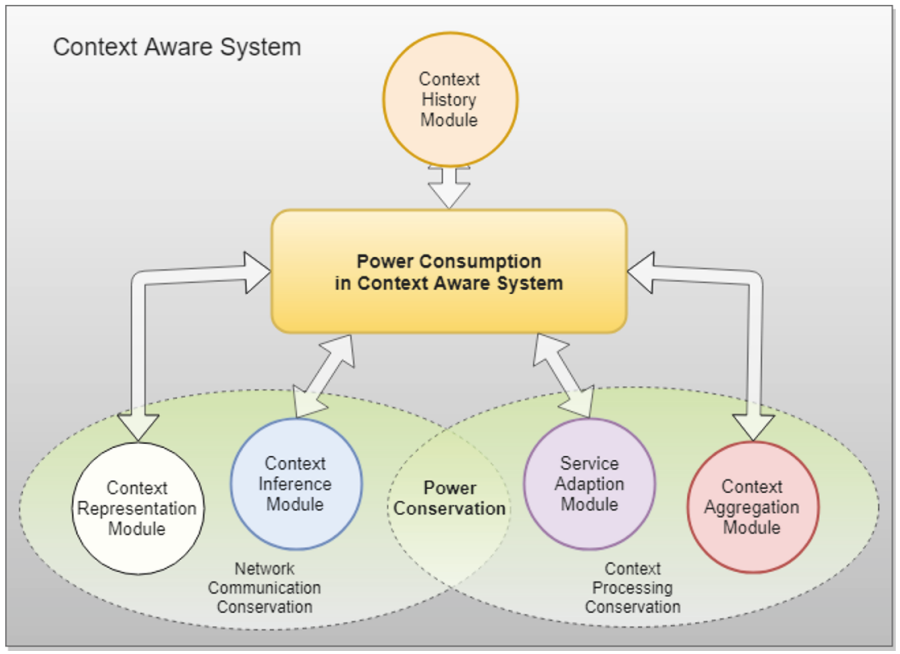

Figure 4: Framework for power awareness in context aware systems.

power consumed by context inference module P_{cim} is given by

$$P_{cim} = p_{trg} + p_{classify} \tag{4}$$

6.1.5. Power Consumption in Service Adaptation Module. This module adjusts both internal and external services present in the environment based on the recommendation from the context inference module. This module communicates with the external services over the network. Retries are required for successful transmission due to noisy and error-prone wireless network.

If the power consumed by the service is p_{svc}, the power consumed by the network for each external service is p_{nw}, the number of retries is t, and the number of services is v, the power consumed by the service adaptation module P_{sam} is given by

$$P_{sam} = \sum_{i=1}^{v} \left(p_{svc} + \left(p_{nw} \times t\right)\right) \tag{5}$$

6.2. Power Conservation Approach. To conserve power numerous methods and safeguards have been proposed. Based on the consumption characteristics, the power consumption centers within a context aware system are classified as network communication-based conservation and context processing-based conservation. A brief description of these is given below.

6.2.1. Network Communication-Based Conservation. The major usage of this method is in network communication, which is required in the context aggregation module and the service adaptation module. It is evident from (1) and (5) that the channel may result in a number of retries, which not only consumes energy but also takes more time. It should be pointed out that the cloud-based conservation techniques have more network utilization than the rest.

6.2.2. Context Processing-Based Conservation. The context representation module and the context inference module both consume power as part of the context processing. This is dependent on the hardware as well as the OS support. The cloud-based conservation techniques have low context processing as it is carried out over the cloud. The context processing is also algorithm dependent where the size of the data set increases the classification or training time and thus consumes more power.

7. Results and Discussions

In this section, we describe experimental setup used for evaluation of proposed framework then provide our observations as well as corresponding results. The tests are conducted on Dell Latitude E5430 and Samsung R450R4V machines. The tests simulates context data gathering, storage, and processing. The power consumption is measured as change in battery level. Object-oriented Java program using NetBeans version 8.2 is developed that measures the battery level through Java Native Access (JNA) library. The test results are summarized in Table 5, which shows the power consumption in percentage (%) and time elapsed in nanoseconds for context aware system. The power consumption is measured as the change in battery level of a test machine. The relationship between power consumption and time of execution is shown in

Table 5: Test results of power consumption and time elapsed for different machines.

Test No	Dell Latitude E5430		Samsung R450R4V	
	Battery Consumption Level (%)	Time Elapsed (nanoseconds)	Battery Consumption Level (%)	Time Elapsed (nanoseconds)
1	1	102871511678.00	1	107283040801.00
2	2	201146140580.00	2	202699872266.00
3	2	301197287625.00	2	305333596221.00
4	5	401126581201.00	3	402140583122.00
5	5	501171081067.00	4	505597135941.00
6	5	601158529665.00	5	604191542742.00
7	6	701601643796.00	6	70212151347.00
8	7	801798038474.00	6	802937759878.00
9	8	901387821639.00	7	908428301040.00
10	9	1005727418757.00	8	1002842365866.00

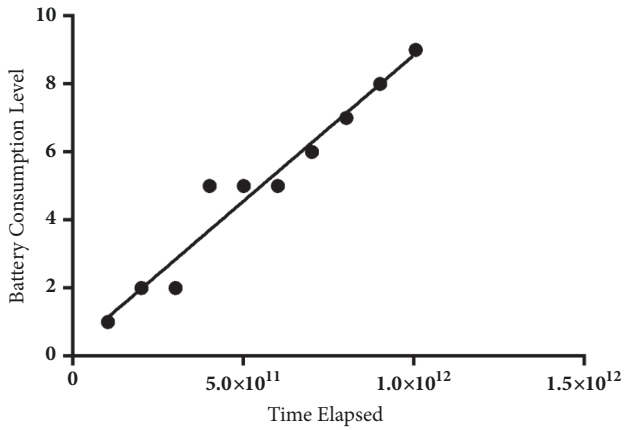

Figure 5: Linear least square line for battery consumption and time elapsed on machine no. 1.

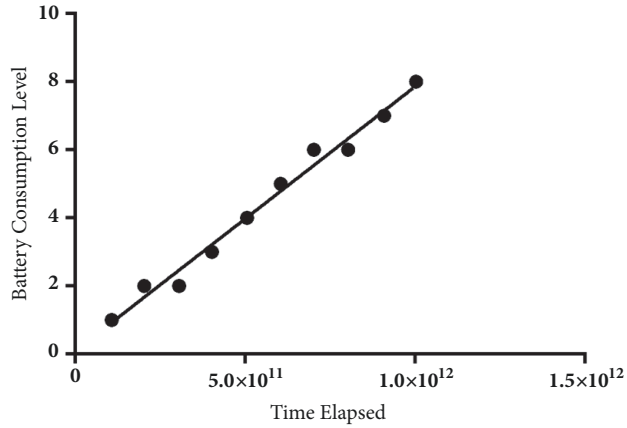

Figure 6: Linear least square line for battery consumption and time elapsed on machine no. 2.

Figures 5 and 6. The linear least square line has been calculated and is shown in

$$BatteryConsumptionLevel = 8.589^{-12} \times TimeElapsed + 0.25930 \quad (6)$$

$$BatteryConsumptionLevel = 7.762^{-12} \times TimeElapsed + 0.09720 \quad (7)$$

Both (6) and (7), as well as Figures 5 and 6, conform that the increase in activity that includes communication and processing causes a linear increase in power consumption. This observation conforms to the proposed framework in Section 6.1.

8. Conclusion

This paper presents a comparison of various contemporary power awareness techniques within the scope of context aware systems. Different techniques to achieve power awareness are classified in this work. The review also includes power consumption profiling to identify how power consumption can be measured. Several recommendations for power conservation are also provided that will help in identifying areas of improvement within a context aware system. A framework is also presented which models the power consumption at different stages of a context aware system. The results conform to the framework which shows that the power consumption is a linear function of execution time. This can then be optimized using network communication conservation and context processing conservation.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The work reported in this paper was supported by the National Natural Science Foundation of China (Grant no. 61672080).

References

[1] N. A. Malik, U. Mahmud, and M. Y. Javed, "Future challenges in context aware computing," in *WWW/Internet 2007*, pp. 306–310, 2007.

[2] T. Strang and C. Linnhoff-Popien, "A context modeling survey," in *Proceedings of the Workshop on Advanced Context Modelling, Reasoning and Management (UbiComp '04)*, pp. 34–41, 2004.

[3] P. Daponte, L. De Vito, F. Picariello, and M. Riccio, "State of the art and future developments of measurement applications on smartphones," *Measurement*, vol. 46, no. 9, pp. 3291–3307, 2013.

[4] M. Alaa, A. A. Zaidan, B. B. Zaidan, M. Talal, and M. L. M. Kiah, "A review of smart home applications based on Internet of Things," *Journal of Network and Computer Applications*, vol. 97, pp. 48–65, 2017.

[5] T. Cioara, I. Anghel, I. Salomie, G. Copil, B. Pernici, and D. Moldovan, "A context aware self-adapting algorithm for managing the energy efficiency of IT service centers," *Ubiquitous Computing and Communication Journal*, vol. 6, pp. 619–630, 2011.

[6] S. L. Kiani, A. Anjum, N. Antonopoulos, and M. Knappmeyer, "Context-aware service utilisation in the clouds and energy conservation," *Journal of Ambient Intelligence and Humanized Computing*, vol. 5, no. 1, pp. 111–131, 2014.

[7] O. Yuryur, *Energy Efficient Context-Aware Framework in Mobile Sensing [Ph.D. Thesis]*, University of South Florida, Florida, FL, USA, 2013.

[8] K. Naik, *A Survey of Software Based Energy Saving Methodologies for Handheld Wireless Communication Devices*, Department of Electrical and Computer Engineering, University of Waterloo, 2010.

[9] K. Ravindranath and K. R. S. Rao, "A survey on energy aware offloading techniques for mobile cloud computing," *International Journal of Computer Trends and Technology*, vol. 4, pp. 2081–2086, 2013.

[10] C. Arun and V. Jaiganesh, "Survey on minimizing energy consumption in mobile cloud computing," *International Journal of Computer Applications*, vol. 150, no. 3, pp. 5–8, 2016.

[11] Z. Li, S. Tesfatsion, S. Bastani et al., "A survey on modeling energy consumption of cloud applications: deconstruction, state of the art, and trade-off debates," *IEEE Transactions on Sustainable Computing*, vol. 2, no. 3, pp. 255–274, 2017.

[12] S. Mittal, "A survey of techniques for improving energy efficiency in embedded computing systems," *International Journal of Computer Aided Engineering and Technology*, vol. 6, no. 4, pp. 440–459, 2014.

[13] M. H. Jofri, M. F. M. Fudzee, and M. N. Ismail, "A survey on energy-aware profiler for mobile devices," in *Advances in Intelligent Systems and Computing*, vol. 331, pp. 295–305, Springer, 2015.

[14] J. F. M. Bernai, L. Ardito, M. Morisio, and P. Falcarin, "Towards an efficient context-aware system: Problems and suggestions to reduce energy consumption in mobile devices," in *Proceedings of the 9th International Conference on Mobile Business/9th Global Mobility Roundtable (ICMB and GMR '10)*, pp. 510–514, June 2010.

[15] T. Rault, A. Bouabdallah, Y. Challal, and F. Marin, "A survey of energy-efficient context recognition systems using wearable sensors for healthcare applications," *Pervasive and Mobile Computing*, vol. 37, pp. 23–44, 2017.

[16] M. N. Ismail, R. Ibrahim, and M. F. Md Fudzee, "A survey on content adaptation systems towards energy consumption awareness," *Advances in Multimedia*, vol. 2013, Article ID 871516, 8 pages, 2013.

[17] J. Wang, L. Feng, W. Xue, and Z. Song, "A survey on energy-efficient data management," *ACM SIGMOD Record*, vol. 40, no. 2, pp. 17–23, 2011.

[18] S. Brienza, S. E. Cebeci, S. S. Masoumzadeh, H. Hlavacs, Ö. Özkasap, and G. Anastasi, "A survey on energy efficiency in P2P systems: File distribution, content streaming, and epidemics," *ACM Computing Surveys*, vol. 48, no. 3, 2015.

[19] R. Bolla, R. Bruschi, F. Davoli, and F. Cucchietti, "Energy efficiency in the future internet: a survey of existing approaches and trends in energy-aware fixed network infrastructures," *IEEE Communications Surveys & Tutorials*, vol. 13, no. 2, pp. 223–244, 2011.

[20] M. R. Celenliôglu, D. Gözüpek, and H. A. Mantar, "A survey on the energy efficiency of vertical handover mechanisms," in *Proceedings of the International Conference on Wireless and Mobile Networks (WiMoN '13)*, 2013.

[21] U. Mahmud, N. Iltaf, A. Rehman, and F. Kamran, "Context-Aware Paradigm for a Pervasive Computing Environment (CAPP)," in *WWW/Internet 2007*, pp. 337–346, Villa Real, Portugal, 2007.

[22] M. Knappmeyer, S. L. Kiani, E. S. Reetz, N. Baker, and R. Tonjes, "Survey of context provisioning middleware," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 3, pp. 1492–1519, 2013.

[23] M. Hashemi and A. Sadeghi-Niaraki, "A theoretical framework for ubiquitous computing," *International Journal of Advanced Pervasive and Ubiquitous Computing*, vol. 8, no. 2, pp. 1–15, 2016.

[24] U. Alegre, J. C. Augusto, and T. Clark, "Engineering context-aware systems and applications: A survey," *The Journal of Systems and Software*, vol. 117, pp. 55–83, 2016.

[25] U. Mahmud and N. A. Malik, "Flow and threat modelling of a context aware system," *International Journal of Advanced Pervasive and Ubiquitous Computing*, vol. 6, no. 2, pp. 58–70, 2014.

[26] U. Mahmud, "UML based model of a context aware system," *International Journal of Advanced Pervasive and Ubiquitous Computing*, vol. 7, no. 1, pp. 1–16, 2015.

[27] U. Mahmud, "Organizing contextual data in context aware systems: A review," in *Handbook of Research on Human-Computer Interfaces, Developments, and Applications*, P. A. Hershey, Ed., pp. 273–303, IGI Global, 2016.

[28] G. W. Musumba and H. O. Nyongesa, "Context awareness in mobile computing: A review," *International Journal of Machine Learning and Applications*, vol. 2, no. 1, 2013.

[29] J.-Y. Hong, E.-H. Suh, and S.-J. Kim, "Context-aware systems: a literature review and classification," *Expert Systems with Applications*, vol. 36, no. 4, pp. 8509–8522, 2009.

[30] S. L. Kiani, A. Anjum, M. Knappmeyer, N. Bessis, and N. Antonopoulos, "Federated broker system for pervasive context provisioning," *The Journal of Systems and Software*, vol. 86, no. 4, pp. 1107–1123, 2013.

[31] U. Mahmud, U. Farooq, M. Y. Javed, and N. A. Malik, "Representing and organizing contextual data in context aware environments," *Journal of Computing*, vol. 4, pp. 61–67, March 2012.

[32] J. Grudin, "Desituating action: Digital representation of context," *Human-Computer Interaction*, vol. 16, no. 2-4, pp. 269–286, 2001.

[33] L. Feng, P. M. G. Apers, and W. Jonker, "Towards context-aware data management for ambient intelligence," *Lecture Notes in Computer Science*, vol. 3180, pp. 422–431, 2004.

[34] U. Mahmud, N. Iltaf, and F. Kamran, "Context congregator: gathering contextual information," in *Proceedings of the 5th Frontiers of Information Technology (FIT '07)*, pp. 134–141, Islamabad, Pakistan, 2007.

[35] N. A. Malik, M. Y. Javed, and U. Mahmud, "Estimating user preferences by managing contextual history in context aware systems," *Journal of Software* , vol. 4, no. 6, pp. 571–576, 2009.

[36] U. Mahmud and M. Y. Javed, "Context inference engine (CiE): inferring context," *International Journal of Advanced Pervasive and Ubiquitous Computing*, vol. 4, no. 3, pp. 13–41, 2012.

[37] U. Mahmud and M. Y. Javed, "Context Inference Engine (CiE): Classifying activity of context using minkowski distance and standard deviation-based ranks," in *Systems and Software Development, Modeling, and Analysis: New Perspectives and Methodologies*, pp. 65–112, IGI Global, 2014.

[38] S. Hussain, Z. Wang, and I. K. Toure, "An approach for QoS measurement and web service selection sureness," *High Technology Letters*, vol. 19, no. 3, pp. 283–289, 2013.

[39] S. Hussain, Z. S. Wang, and I. K. Toure, "Performance analysis of web services in different types of internet technologies," *Applied Mechanics and Materials*, vol. 513-517, pp. 1431–1436, 2014.

[40] O. Landsiedel, K. Wehrle, and S. Götz, "Accurate prediction of power consumption in sensor networks," in *Proceedings of the 2nd IEEE Workshop on Embedded Networked Sensors (EmNetS-II '05)*, pp. 37–44, May 2005.

[41] J. Flinn, *Extending Mobile Computer Battery Life Through Energy-Aware Adaptation [Ph.D. Thesis]*, Carnegie Melon University, Pittsburgh, PA, USA, 2001.

[42] A. Kansal, F. Zhao, J. Liu, N. Kothari, and A. A. Bhattacharya, "Virtual machine power metering and provisioning," in *Proceedings of the 1st ACM Symposium on Cloud Computing (SoCC '10)*, pp. 39–50, Indianapolis, IN, USA, June 2010.

[43] 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 ComputerSystems (EuroSys '12)*, pp. 29–42, Bern, Switzerland, April 2012.

[44] J. Kulk and J. Welsh, "A low power walk for the NAO robot," in *Proceedings of the Australasian Conference on Robotics and Automation (ACRA '08)*, pp. 1–7, December 2008.

[45] C. Seo, G. Edwards, S. Malek, and N. Medvidovic, "A framework for estimating the impact of a distributed software system's architectural style on its energy consumption," in *Proceedings of the 7th IEEE/IFIP Working Conference on Software Architecture (WICSA '08)*, pp. 277–280, February 2008.

[46] N. Vallina-Rodriguez, J. Shah, A. Finamore et al., "Breaking for commercials: Characterizing mobile advertising," in *Proceedings of the ACM Internet Measurement Conference (IMC '12)*, pp. 343–356, November 2012.

[47] Y. Li, H. Chen, and W. Shi, "Power behavior analysis of mobile applications using Bugu," *Sustainable Computing: Informatics and Systems*, vol. 4, no. 3, pp. 183–195, 2014.

[48] A. B. Abkenar, S. W. Loke, W. Rahayu, and A. Zaslavsky, "Energy considerations for continuous group activity recognition using mobile devices: The case of GroupSense," in *Proceedings of the 30th IEEE International Conference on Advanced Information Networking and Applications (AINA '16)*, pp. 479–486, Crans-Montana, Switzerland, March 2016.

[49] M. Marcu, "Powerthermal profiling of software applications," *Microelectronics Journal*, vol. 42, no. 4, pp. 601–608, 2011.

[50] L.-T. Duan, B. Guo, Y. Shen, Y. Wang, and W.-L. Zhang, "Energy analysis and prediction for applications on smartphones," *Journal of Systems Architecture*, vol. 59, no. 10, pp. 1375–1382, 2013.

[51] Y.-F. Chung, C.-Y. Lin, and C.-T. King, "ANEPROF: Energy profiling for android java virtual machine and applications," in *Proceedings of the 17th IEEE International Conference on Parallel and Distributed Systems (ICPADS '11)*, pp. 372–379, Tainan, Taiwan, December 2011.

[52] M. Tanabian, "Mobile device/application power profiling—testing and design considerations in mobile devices & apps for power performance and battery life," *Intuigence Group, White Paper*, 2011.

[53] Y.-D. Lin, E. Rattagan, Y.-C. Lai et al., "Calibrating parameters and formulas for process-level energy consumption profiling in smartphones," *Journal of Network and Computer Applications*, vol. 44, pp. 106–119, 2014.

[54] M. Lee, D.-K. Kim, and J.-W. Lee, "Analysis of characteristics of power consumption for context-aware mobile applications," *Information*, vol. 5, no. 4, pp. 612–621, 2014.

[55] T. Mitchell, *Machine Learning*, McGraw Hill, 1997.

[56] S. Elmalaki, M. Gottscho, P. Gupta, and M. Srivastava, "A case for battery charging-aware power management and deferrable task scheduling in smartphones," in *Proceedings of the 6th USENIX Conference on Power-Aware Computing and Systems (HotPower '14)*, pp. 1–4, Broomfield, CO, USA, 2014.

[57] G. Anastasi, S. Brienza, G. L. Re, and M. Ortolani, "Energy-efficient protocol design," in *Green Communications: Principles, Concepts and Practice*, K. Samdanis, P. Rost, A. Maeder, M. Meo, and C. Verikoukis, Eds., pp. 339–360, John Wiley & Sons, Ltd, Chichester, UK, 2015.

[58] J. Flinn and M. Satyanarayanan, "Energy-aware adaptation for mobile applications," in *Proceedings of the Seventeenth ACM Symposium on Operating Systems Principles*, pp. 48–63, Charleston, SC, USA, December 1999.

[59] S. Elmalaki, L. Wanner, and M. Srivastava, "CAreDroid: Adaptation framework for android context-aware applications," in *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking (MobiCom '15)*, pp. 386–399, Paris, France, September 2015.

[60] M. Moghimi, J. Venkatesh, P. Zappi, and T. Rosing, "Context-aware mobile power management using fuzzy inference as a service," in *Proceedings of the International Conference on Mobile Computing, Applications, and Services (MobiCASE '12)*, vol. 110, pp. 314–327, Seattle, WA, USA, 2012.

[61] Y. Fei, Z. Lin, and N. K. Jha, "An energy-aware framework for coordinated dynamic software management in mobile computers," in *Proceedings of the IEEE Computer Society's 12th Annual International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems (MASCOTS '04)*, pp. 306–317, Volendam, Netherlands, October 2004.

[62] J. Chen and G. Venkataramani, "EnDebug: A hardware-software framework for automated energy debugging," *Journal of Parallel and Distributed Computing*, vol. 96, pp. 121–133, 2016.

[63] P. Simoens, C. Mahieu, F. Ongenae et al., "Internet of robotic things: context-aware and personalized interventions of assistive social robots," in *Proceedings of the 5th IEEE International*

Conference on Cloud Networking (CloudNet '16), pp. 204–207, Pisa, Italy, October 2016.

[64] Q. Zhang, G. Metri, S. Raghavan, and W. Shi, "RESCUE: An energy-aware scheduler for cloud environments," *Sustainable Computing*, vol. 4, no. 4, pp. 215–224, 2014.

[65] N. Akhter and M. Othman, "Energy aware resource allocation of cloud data center: review and open issues," *Cluster Computing*, vol. 19, no. 3, pp. 1163–1182, 2016.

[66] S. Kang, J. Lee, H. Jang, Y. Lee, S. Park, and J. Song, "A scalable and energy-efficient context monitoring framework for mobile personal sensor networks," *IEEE Transactions on Mobile Computing*, vol. 9, no. 5, pp. 686–702, 2010.

[67] D. Sathan, A. Meetoo, and R. K. Subramaniam, "Context aware lightweight energy efficient framework," *World Academy of Science Engineering and Technology*, vol. 52, pp. 64–70, 2009.

[68] R. Hermann, P. Zappi, and T. S. Rosing, "Context aware power management of mobile systems for sensing applications," in *Information Processing in Sensor Networks*, pp. 1–5, 2012.

[69] T. Simunic, L. Benini, and G. De Micheli, "Energy-efficient design of battery-powered embedded systems," in *Proceedings of the International Symposium on Low Power Electronics and Design (ISLPED '99)*, pp. 212–217, San Diego, Calif, USA, August 1999.

[70] T. Ŝimunić, L. Benini, and G. De Micheli, "Energy-efficient design of battery-powered embedded systems," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 9, no. 1, pp. 15–28, 2001.

[71] V. Tiwari, S. Malik, A. Wolfe, and M. T.-C. Lee, "Instruction level power analysis and optimization of software," in *Technologies for Wireless Computing*, pp. 139–154, Springer, 1996.

[72] V. Tiwari, S. Malik, and A. Wolfe, "Power analysis of embedded software: a first step towards software power minimization," in *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*, pp. 437–445, San Jose, Calif, USA, 1994.

[73] H. Mehta, R. M. Owens, M. J. Irwin, R. Chen, and D. Ghosh, "Techniques for low energy software," in *Proceedings of the International Symposium on Low Power Electronics and Design (ISLPED '97)*, pp. 72–75, Monterey, Calif, USA, August 1997.

[74] E. Capra, C. Francalanci, and S. A. Slaughter, "Is software 'green'? Application development environments and energy efficiency in open source applications," *Information and Software Technology*, vol. 54, no. 1, pp. 60–71, 2012.

[75] S. Hao, D. Li, W. G. J. Halfond, and R. Govindan, "Estimating mobile application energy consumption using program analysis," in *Proceedings of the 35th International Conference on Software Engineering (ICSE '13)*, pp. 92–101, San Francisco, Calif, USA, May 2013.

[76] S.-L. Tsao, C.-K. Yu, and Y.-H. Chang, "Profiling energy consumption of I/O functions in embedded applications," in *Proceedings of the International Conference on Architecture of Computing Systems (ARCS '13)*, vol. 7767, pp. 195–206, Prague, Czech, 2013.